Animation with threshold textures

Oleg Veryovka†
Mainframe Entertainment

†Currently with Electronic Arts cluster
New Media Innovation Center (NewMIC),
Vancouver, BC

Abstract

We present a method for frame coherent texturing and hatching of 3D models with a discrete set of colors. Our technique is inspired by various artistic styles that use a limited set of colors to convey surface shape and texture. In previous research discrete color shading was produced by modifying smooth shading with a threshold function. We extend this approach and specify threshold values with an image or a procedural texture. Texture values and mapping coordinates are adapted to surface orientation and scale. Aliasing artifacts are eliminated by the modified filtering technique. The threshold texturing approach enables an animator to control local shading and to display surface roughness and curvature with a limited set of colors.

Key words: non-photorealistic rendering, texture mapping, filtering, animation

1 Introduction

Artistic rendering is the process of presenting visual information through the *" selection of significant and suppression of non-essential"*[26]. An artist emphasizes important features and minimizes extraneous details in the pursuit of visually effective representation. In cartoon style rendering, visual details are reduced in order to draw the audience into the story and to add humor and emotional appeal[15]. Cartoonists draw silhouette lines and provide visual cues of surface shading using minimal variations of colors. Hatching and texturing is an effective method of suggesting surface properties and materials. Unfortunately, drawing frame coherent animated hatching lines by hand is an extremely laborious process and thus hatching is often limited to rendering of static objects and backgrounds.

Frame coherent non-photorealistic rendering (NPR) is one of the most challenging problems of computer graphics research[2]. Current approaches enable digital animation in pen-and-ink, oil and water color painting styles using particle-based systems[17, 12, 5], graftals[14, 16], tonal art maps[22], and generalized mip-maps[8]. A texture mapping approach was used in the past to assist in cel animation[6] but was not applied to rendering with a limited set of colors.

Recent NPR research had successfully approximated the style of cartoon rendering. A silhouette detection problem was investigated by Buchanan and Sousa[3], Northrup and Markosian[18] and Gooch et. al. [11]. A system for real time cartoon shading was developed by Lake et. al. [15]. Commercial and public domain "toon" shaders are offered for most common animation systems[24]. Unfortunately, the issue of texturing and hatching with a limited set colors has not been sufficiently investigated.

The goal of our research is the development of a frame coherent rendering algorithm for the display of textures using a discrete set of colors. Our approach is based on threshold textures and is closely related to stylistic halftoning area of NPR. We review stylistic halftoning results in the following section.

1.1 Stylistic halftoning

Halftoning is the process of image approximation with a limited set colors. A texture is often introduced as a result of this approximation. Research in conventional halftoning attempts to hide this texture[28].

Non-photorealistic halftoning uses this texture to create a stylized rendering of an image and to convey additional information. Ostromoukhov and Hersch[19] modified ordered dithering and enabled artists to design single screen elements. This technique was later extended to multi-color halftoning[20]. Buchanan[4] introduced a variety of halftoning textures by altering various parameters in a clustered error diffusion method. Veryovka and Buchanan[29] generated dither matrices by processing a texture image with the clipped adaptive histogram equalization (CLAHE) algorithm. This technique enables designers to emboss a halftoned image with a desired texture or to imitate traditional illustration styles.

A number of algorithms adapt halftoning texture to the rendered image thus enhancing its display. Streit and Buchanan[27] developed an importance driven halftoning system and controlled the placement of graphics elements by an importance function. Ostromoukhov[21] presented an interactive digital facial engraving system based on halftoning. Durand et. al.[7] extended this halftoning approach to other styles of traditional drawing. Veryovka and Buchanan[30] investigated an algorithm for automatic control of halftoning texture using image buffers that store 2D and 3D information.

All the previous stylized halftoning techniques worked in image space and are not applicable to frame coherent animation. We extend stylized dither matrix approach to digital animation. Our method is based on a mapping of threshold textures onto 3D models. We introduce procedural and image based textures that enhance rendering, highlight important model features, suggest surface textures, and approximate traditional rendering styles. Our technique allows an animator to control the number of colors and their appearance depending on lighting and viewing conditions. Threshold textures evolve with changes in surface scale and orientation. We address aliasing problems by modifying previous texture filtering methods. In conclusion, we discuss advantages and limitation of the texture thresholding approach.

2 Discrete color shading with a step function

A cartoon rendering style is associated with shading of surfaces with a limited set of colors. In computer graphics this shading effect can be approximated by augmenting a conventional illumination Equation 1 with a step function. Consider the following illumination equation:

$$I = m_a l_a + \sum_i m_d f_d(N, L_i) l_i + \sum_i m_s f_s(N, L_i, V) l_i. \quad (1)$$

where V is the eye vector, N is a surface normal, L_i is the direction to the i-th light of color l_i, m_a, m_d, m_s are ambient, diffuse and specular colors of the material; l_a is an ambient light of the scene. Functions $f_d(N, L_i)$ and $f_s(N, L_i, V)$ are continuous diffuse and specular components that produce continuous shading values. We modify the result of f_d and f_s by a step function T (Equation 2), thus limiting shading values to a set of discrete colors.

$$I = m_a l_a + \sum_i m_d T(f_d(N, L_i)) l_i + \sum_i m_s T(f_s(N, L_i, V)) l_i, \quad (2)$$

In the previous research[15, 24] a threshold function $T(x, t)$ was used as a step function:

$$T(x,t) = \begin{cases} 0 & \text{if } x < t \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

where t is a threshold. A single threshold of $t = 0.5$ was suggested. Diffuse shading with a single threshold will produce only 2 colors. Multi-color shading with $n + 1$ discrete colors can be achieved by modifying a step function as follows:

$$T_n(x,t) = \frac{1}{n}(\lfloor xn \rfloor + T(Frac(xn), t)), \quad (4)$$

where $Frac(x)$ denotes a fractional-part function $Frac(x) = x - \lfloor x \rfloor$. A modification of the diffuse shading component with a step function T_n, where $n = 2$ results in a discrete three color shading (Figure1).

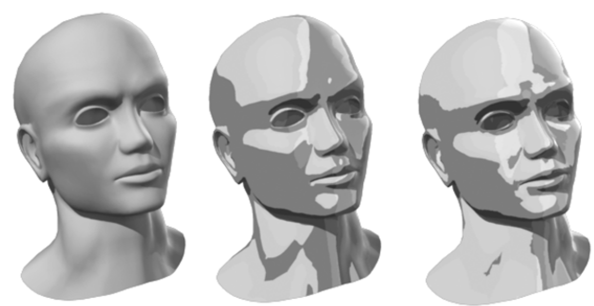

Figure 1: Thresholding components of the smooth shading model (left) introduces discrete tone shading (center). Threshold values vary across the surface controlling local shading and enhancing surface display (right).

The style of discrete color shading produced thus far approximates cartoon style rendering found in hand-drawn cel animation. We extend the thresholding technique and introduce a threshold texture to vary threshold values in the step function T (Equation 4).

3 Threshold textures

A variation of threshold values in the shading equation (Equation 2) modifies the resulting color locally. We use this property to control local shading highlighting model geometry and creating surface texturing.

Discrete color shading may hide important details of the geometry and exaggerate others. For example, the forehead surface of a head model (Figure 1, center) appear disconnected, muscles of the neck seem overly exaggerated. A threshold texture is constructed to highlight forehead bones, eyes, and lips (Figure 1, right).

In the rest of the section we present techniques for the creation of threshold textures that can be used to convey surface roughness and to imitate line hatching. The following considerations are important in the design of threshold textures.

Threshold textures modify surface shading and may significantly alter perception of an object shape. For ex-

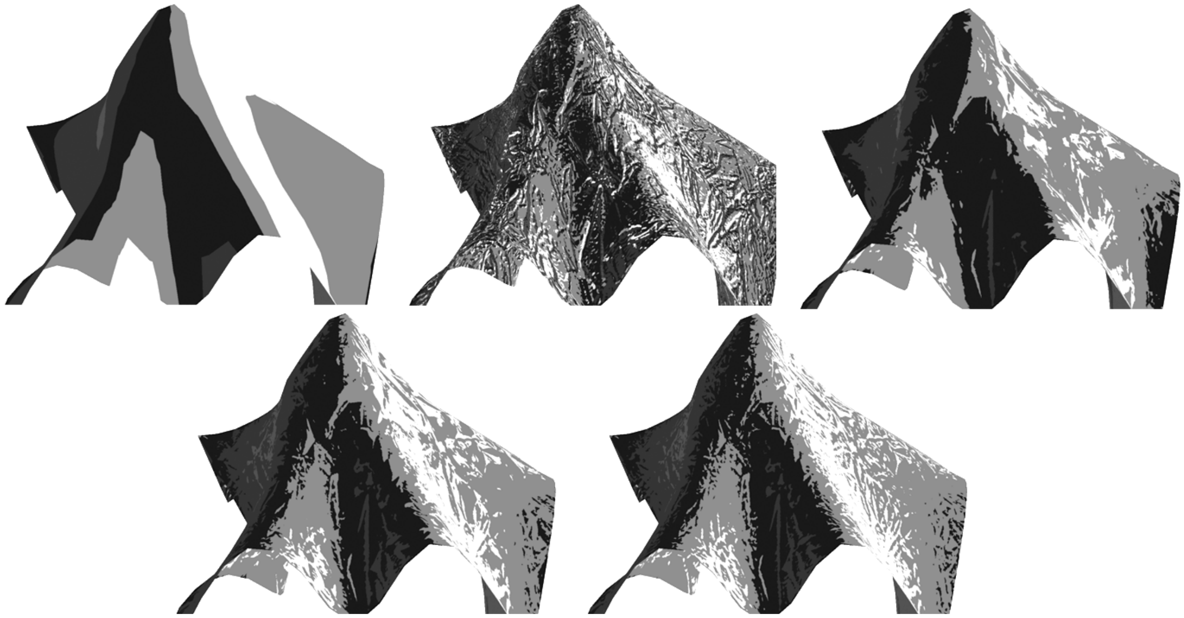

Figure 2: Discrete color shading with a single threshold (top-left) is enhanced by texture mapping. One approach is to use a texture as a bump map (top-center). When an unprocessed texture is used as thresholds its non-uniform distribution of values may distort display of the surface shape (top-left). We apply clipped histogram equalization algorithm (CLAHE) thus approximating uniform and homogeneous distribution of thresholds. The scale of texture features is controlled by CLAHE window parameter w. Smaller w produce softer textures ($w = 16$ pixels, bottom-left; $w = 8$ pixels bottom-right).

ample, the wrinkled paper texture distorts shading of the patch hiding surface folds ((Figure 2, top-right). Thus, texturing should approximate the overall shading of the model. Previous research suggests the use of threshold textures with uniform and homogeneous distribution of values[29]. These properties enable similar discrete approximation of continuous shading values across a 3D surface.

Artists vary surface details depending on surface scale, orientation, distance from the viewer, etc. Thus, it is important to control the level of texture details. Moreover, texture details should gradually evolve with changes in the scene. We found that texture features “attach” to the surface at multiple scales only if local minima and maxima points at lower resolution correspond to the minima and maxima at the higher resolution level.

3.1 Surface texture

We use threshold texturing to convey roughness of a 3D surface. Color texturing and bump mapping is traditionally used for this purpose. However, these methods may be hard to control or not be suitable for rendering with a limited set of colors (Figure 2).

We construct threshold textures from texture images using the adaptive histogram equalization algorithm with clipping of values (CLAHE). Veryovka and Buchanan[29] demonstrated that CLAHE processed images approximate uniform and homogeneous distribution of threshold values. The level of texture detail is controlled by window size and clipping parameters of the algorithm. When processed with a window $w = 8$ the wrinkled paper texture produced softer shading (Figure 2, bottom row). However, static control of the details is not sufficient for the use in animation. The same texture will appear at different scales and orientations and has to be controlled locally (Figure 3, right wall).

One approach is to compute threshold values dynamically and vary histogram equalization window according to the surface normal and distance from the camera. In this case histogram equalization will be recomputed for every texture sample. We approximate dynamic computation of threshold values using a mip-map filtering approach. In a pre-process step, we construct the mip-map texture pyramid and apply the CLAHE algorithm to every level of the pyramid. The resulting rendering smoothly adapts surface texturing depending on surface orientation and scale (Figure 3, floor and left wall).

The smooth transition between texture scales is achieved by tri-linear interpolating between pixel values

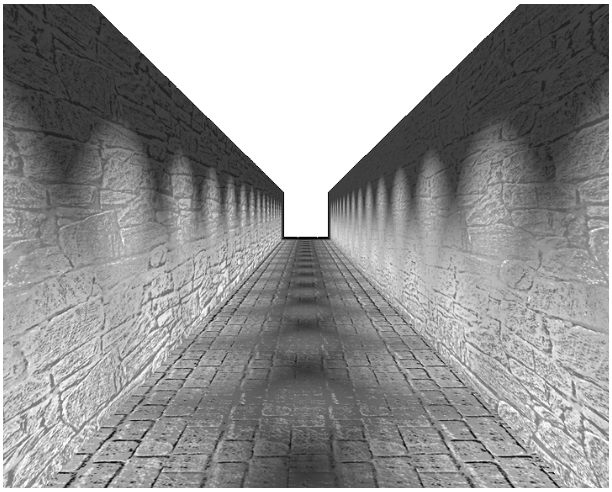

Figure 3: The mip-map filtering reduces the range of threshold values resulting in reduced texture definition (right wall). We apply CLAHE algorithm to every mip-map level to maintain sharp texturing across multiple scales (floor and left wall).

of the closest two mip-map levels. The CLAHE algorithm ensures that dark pixels at one level are mapped to the locally dark pixels at the other level.

Overall, our algorithm constructs threshold textures with the necessary distribution of thresholds and gradually adapts to local texture scale. We believe that discrete shading with image based threshold textures introduces texturing effects and conveys shape and roughness of the surface.

3.2 Line hatching

Line hatching is a widely used method of illustrating surface curvature. Hatching often appears in conjunction with discrete color shading in printed images but is rare in hand-drawn animation. Previous research[9] concludes that the use of textures and line hatching aligned with surface curvature enhances perception of 3D shapes.

We generate procedural line textures and introduce hatching effects of line drawing to digital animation. A similar approach was suggested before[1], however, the proposed textures did not satisfy threshold distribution properties necessary for uniform shading approximation. Procedural line textures with uniform and homogeneous value distribution were developed for comprehensive halftoning of rendered scenes by Veryovka and Buchanan[30]. We extend this previous work to account for multiple scales.

In order to guarantee the approximately constant size of lines we re-map texture coordinates according to the diadic scale of the projected texture[1]. Let us assume that procedural lines are generated by a function of the v texture coordinate and therefore are parallel to the u coordinate. We compute diadic texture scale using the following equation:

$$s = \log_2 \frac{d}{w}, \tag{5}$$

where d is the screen distance between u and $u+1$ texture iso-lines, and w is the desired scale of the procedural texture. Thus, the v texture coordinates are re-mapped as follows:

$$\bar{v} = Frac(v\lfloor s \rfloor) \tag{6}$$

Unfortunately, the use of the integer part of the diadic scale $\lfloor s \rfloor$ in coordinate re-mapping (Equation 6) introduces texture discontinuities. We achieve a smooth transition between texture scales by accounting for the fractional-part of the diadic scale $Frac(s)$. Let $F(v, Frac(s))$ be the multi-scale texture function, and $F(v,0) = f(v)$ represents a texture at an integer diadic scale $\lfloor s \rfloor$. Then, $F(v,1)$ denotes texture at the next scale $\lfloor s \rfloor + 1$ and is constructed as follows:

$$F(v,1) = \begin{cases} F(2v,0) & \text{if } v < 0.5 \\ F(2v-1,0) & \text{otherwise} \end{cases} \tag{7}$$

We generate texture at the intermediate scale $t = Frac(s)$ as a linear combination of $F(v,0)$ and $F(v,1)$:

$$F(v,t) = (1-t)F(v,0) + tF(v,1). \tag{8}$$

Similar to the previous research[30], a base function $f(v) = v$ is used. This function provides uniform distribution of values at all intermediate scales and its local minima and maxima match at consecutive levels. Note, that the $sin(v)$ suggested in [1] does not satisfy this property. The multi-scale textures can be modified by a noise

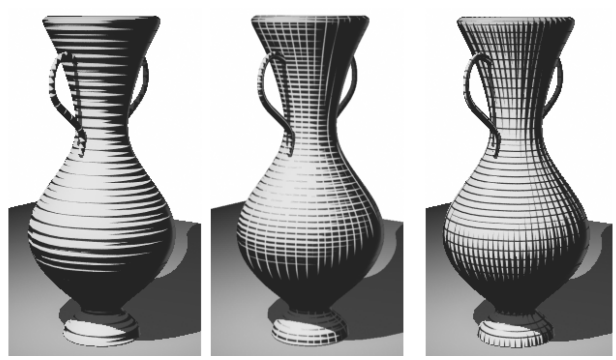

Figure 4: Procedural line textures adapt to surface curvature and orientation. Composition of multiple line textures approximates various styles of cross-hatching.

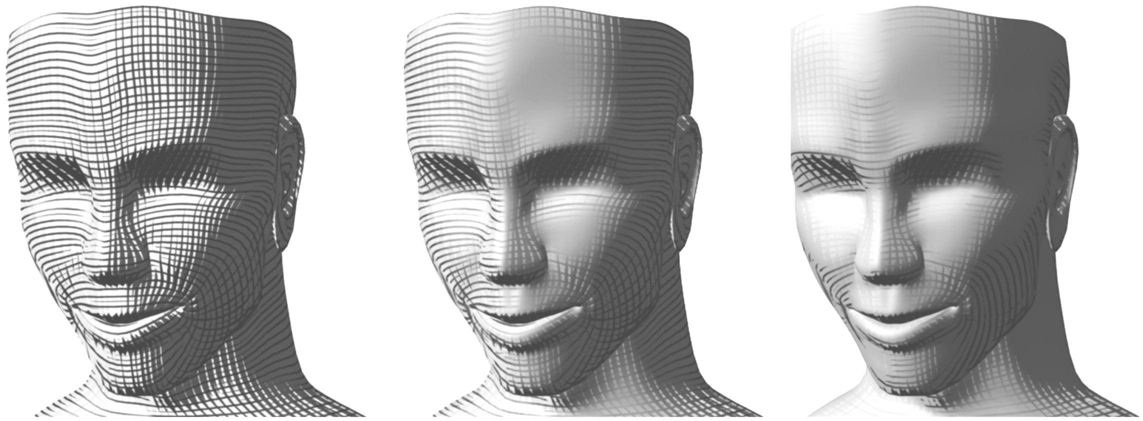

Figure 5: The strength of threshold texturing effects (left) vary depending on surface orientation (center) and illumination (right).

function or composed together to approximate various styles of line drawings: single line, cross-hatching, stippling (Figure 4).

4 Texture filtering

A thresholding process creates sharp discontinuities of the shading intensities. If not accounted for properly these discontinuities create strong aliasing artifacts. It turns out that filtering threshold values does not eliminate these artifacts. This is explained by the fact that smooth variations in threshold texture do not prevent large shading variations.

The conventional approach is filtering the final result with multiple samples per pixel. However, due to the large number of shading discontinuities, a large number of rays is needed making this approach very computationally expensive.

Durand et. al. [7] use "soft thresholding" to eliminate sharp shading transitions. We found that soft thresholding can be used to modify the overall look of the shading but is not sufficient for anti-aliasing in animation.

Our technique is similar to the percentage-closer filtering[23] solution for antialiased shadow rendering with depth maps. We threshold smooth diffuse and specular components with multiple texture samples. Thus, we filter not the texture itself but the result of thresholding. The sampling method depends on the texture used. Analytical filtering is possible for the procedural line textures. In the case of image textures we compute a discrete shading value as a weighted sum of thresholding results with threshold values from two levels of the mip-map pyramid. The resulting filtering combined with multiple rays per pixel is sufficient for elimination of most aliasing artifacts for both procedural and image based textures.

5 Implementation

The discrete shading with threshold texture mapping approach is implemented as a collection of shaders in the Softimage/MentalRay environment.

Texture shaders compute a set of threshold values and their filtering weights. An image texture shader applies CLAHE algorithm and samples mip-map pyramid levels. A procedural line shader re-maps texture coordinates depending on the diadic scale and samples texture in pseudo-random fashion.

A material shader computes soft illumination values first and then thresholds them using texture samples. Multiple samples are filtered according to their precomputed weights. We control the strength of texturing effects by blending between the original and thresholded illumination values. An artist can vary the amount of blending depending on surface orientation and/or illumination level. For example, in Figure 5, the intensity of hatching lines is reduced on the surfaces facing the camera (center). The lines disappear in darkest and brightest areas of the model (right) when the intensity level control is added.

An artist can vary textures for the display of the same model. Wide cross hatching texture was used to display general shading of a vase (Figure 6), while thin lines suggested specular highlights.

Our threshold texturing approach was combined with a contour rendering software implemented in MentalRay (Figure 7). The contours are computed through sampling of geometry and are not affected by the discrete color shading. The use of contours highlighted silhouette edges of the model and complemented the cross-hatching texture (Figure 7).

6 Discussion

The main advantage of the threshold texture mapping approach is its ability to produce frame-coherent rendering in a variety of styles (Figure 6). Textures and lines attach to 3D surfaces and gradually evolve with changes in scale and orientation (Figure 7). Unlike previous techniques[13, 22], the texture display always remains sharp and is not blurred at intermediate mip-map levels. The use of textures with a uniform and homogeneous distribution of values allows a fine control of the overall shading of the model. The threshold texture mapping algorithm can be combined with various illumination models [10, 25] and contour detection algorithms [3, 18].

Our technique has a number of limitations. Unlike tonal art maps [22] the same texture is used for various shading intensities. Particle-based algorithms [17] separately place every brush stroke or a hatch line. We have limited control over line placement through texture coordinates.

7 Conclusions

Artists often use a limited number of colors to represent shading and texturing of surfaces. The main contribution of this research is the threshold texture mapping technique that enables approximation of these artistic styles in digital animation (Figure 6).

Our approach is based on the thresholding of a conventional illumination model. Unlike previous research, threshold values are specified by a procedural or an image-based texture. We found that uniform and homogeneous distribution of thresholds is necessary for a good shading reproduction. Thus, image-based textures were processed with the clipped adaptive histogram equalization algorithm (CLAHE). We modified the mip-map filtering to maintain uniform texture effects regardless of surface orientation and scale. Similarly, procedural line textures are constructed with the necessary distribution of values. In order to maintain constant spacing between the lines, our algorithm recomputes coordinates of procedural textures depending on surface position. The aliasing artifacts were addressed by filtering shading values produced with multiple threshold samples.

The use of threshold texture mapping approach enhanced stylistic rendering in animation. Artists applied image based textures to highlight important features and to suggest surface roughness. Application of procedural textures approximated artistic hatching and helped to convey surface curvature.

Acknowledgements

We thank Chris Welman, Dave Fracchia, Tim Belsher and the entire crew at Mainframe Entertainment for the insightful comments of technical and artistic nature, and for making this publication possible.

References

[1] Larry Gritz Anthony A. Apodaca. *Advanced RenderMan: Creating CGI for Motion Pictures*. Morgan Kaufmann Publishers, 1999.

[2] Amy Ashurst Gooch Bruce Gooch. *Non-Photorealistic Rendering*. A K Peters Ltd, 2001.

[3] John W. Buchanan and Mario C. Sousa. The edge buffer: A data structure for easy silhouette rendering. In *Non-Photorealistic Animation and Rendering 2000 (NPAR '00)*, Annecy, France, June 5-7,2000.

[4] John W. Buchanan. Special effects with half-toning. *Computer Graphics Forum*, 15(3):97–108, August 1996. ISSN 1067-7055.

[5] Derek Cornish, Andrea Rowan, and David Luebke. View-dependent particles for interactive non-photorealistic rendering. *Graphics Interface 2001*, pages 151–158, June 2001. ISBN 0-96888-080-0.

[6] Wagner Toledo Corrêa, Robert J. Jensen, Craig E. Thayer, and Adam Finkelstein. Texture mapping for cel animation. *Proceedings of SIGGRAPH 98*, pages 435–446, July 1998. ISBN 0-89791-999-8. Held in Orlando, Florida.

[7] Fredo Durand, Victor Ostromoukhov, Mathieu Miller, Francois Duranleau, and Julie Dorsey. Decoupling strokes and high-level attributes for interactive traditional drawing. *Eurographics Workshop on Rendering*, 2001.

[8] Bert Freudenberg, Maic Masuch, and Thomas Strothotte. Walk-through illustrations: Frame-coherent pen-and-ink style in a game engine. *Computer Graphics Forum*, 20(3), 2001. ISSN 1067-7055.

[9] Ahna Girshick, Victoria Interrante, Steven Haker, and Todd Lemoine. Line direction matters: An argument for the use of principal directions in 3d line drawings. In *Non-Photorealistic Animation and Rendering 2000 (NPAR '00)*, Annecy, France, June 5-7,2000.

[10] Amy Gooch, Bruce Gooch, Peter Shirley, and Elaine Cohen. A non-photorealistic lighting model for automatic technical illustration. *Proceedings of SIGGRAPH 98*, pages 447–452, July 1998. ISBN 0-89791-999-8. Held in Orlando, Florida.

[11] Bruce Gooch, Peter-Pike Sloan, Amy Gooch, Peter Shirley, and Richard Riesenfeld. Interactive Technical Illustration. *Interactive 3D Conference Proceedings*, April 1999.

[12] Matthew Kaplan, Bruce Gooch, and Elaine Cohen. Interactive artistic rendering. In *Non-Photorealistic Animation and Rendering 2000 (NPAR '00)*, Annecy, France, June 5-7,2000.

[13] Allison W. Klein, Wilmot Li, Misha Kazhdan, Wagner Toledo Corra, Adam Finkelstein, and Thomas Funkhouser. Non-photorealistic virtual environments. *Proceedings of SIGGRAPH 2000*, July 2000. Held in New Orleans, Louisianna.

[14] Michael A. Kowalski, Lee Markosian, J. D. Northrup, Lubomir Bourdev, Ronen Barzel, Loring S. Holden, and John Hughes. Art-based rendering of fur, grass, and trees. *Proceedings of SIGGRAPH 99*, pages 433–438, August 1999. ISBN 0-20148-560-5. Held in Los Angeles, California.

[15] Adam Lake, Carl Marshall, Mark Harris, and Marc Blackstein. Stylized rendering techniques for scalable real-time 3d animation. *NPAR 2000 : First International Symposium on Non Photorealistic Animation and Rendering*, pages 13–20, June 2000.

[16] Lee Markosian, Barbara J. Meier, Michael A. Kowalski, Loring S. Holden, J.D. Northrup, and John F. Hughes. Art-based rendering with continuous levels of detail. In *Non-Photorealistic Animation and Rendering 2000 (NPAR '00)*, Annecy, France, June 5-7,2000.

[17] Barbara J. Meier. Painterly rendering for animation. *Proceedings of SIGGRAPH 96*, pages 477–484, August 1996.

[18] J.D. Northrup and Lee Markosian. Artistic silhouettes: A hybrid approach. In *Non-Photorealistic Animation and Rendering 2000 (NPAR '00)*, Annecy, France, June 5-7,2000.

[19] Victor Ostromoukhov and Roger D. Hersch. Artistic screening. *Proceedings of SIGGRAPH 95*, pages 219–228, August 1995. ISBN 0-201-84776-0. Held in Los Angeles, California.

[20] Victor Ostromoukhov and Roger D. Hersch. Multicolor and artistic dithering. *Proceedings of SIGGRAPH 99*, pages 425–432, August 1999. ISBN 0-20148-560-5. Held in Los Angeles, California.

[21] Victor Ostromoukhov. Digital facial engraving. *Proceedings of SIGGRAPH 99*, pages 417–424, August 1999. ISBN 0-20148-560-5. Held in Los Angeles, California.

[22] Emil Praun, Hugues Hoppe, Matthew Webb, and Adam Finkelstein. Real-time hatching. *Proceedings of SIGGRAPH 2001*, pages 579–584, August 2001. ISBN 1-58113-292-1.

[23] William T. Reeves, David H. Salesin, and Robert L. Cook. Rendering antialiased shadows with depth maps. In *Computer Graphics (Proceedings of SIGGRAPH 87)*, volume 21, pages 283–291, Anaheim, California, July 1987.

[24] Craig Reynolds. Stylized depiction in computer graphics. http://www.red3d.com/cwr/npr/.

[25] Peter-Pike Sloan, William Martin, Amy Gooch, and Bruce Gooch. The lit sphere: A model for capturing npr shading from art. *Graphics Interface 2001*, pages 143–150, June 2001. ISBN 0-96888-080-0.

[26] Harold Speed. *The Practice and Science of Drawing*. Steeley, Service and Co. Ltd., London, 1917.

[27] Lisa Streit and John Buchanan. Importance driven halftoning. *Computer Graphics Forum*, 17(3):207–218, 1998. ISSN 1067-7055.

[28] Robert Ulichney. *Digital Halftoning*. MIT Press, 1987.

[29] Oleg Veryovka and John Buchanan. Texture-based dither matrices. *Computer Graphics Forum*, 19(1):51–64, March 2000. ISSN 1067-7055.

[30] Oleg Veryovka and John Buchanan. Comprehensive halftoning of 3D scenes. *Computer Graphics Forum*, 18(3):13–22, September 1999. ISSN 1067-7055.

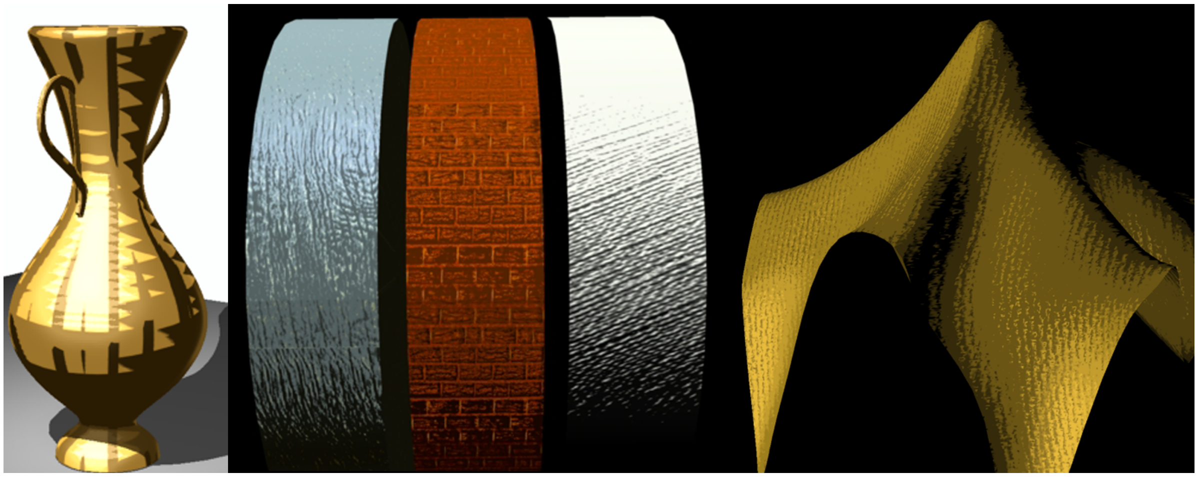

Figure 6: Image-based and procedural textures modify shading suggesting surface materials (wood, brick, textile) or traditional art media (brush strokes, pencil drawing).

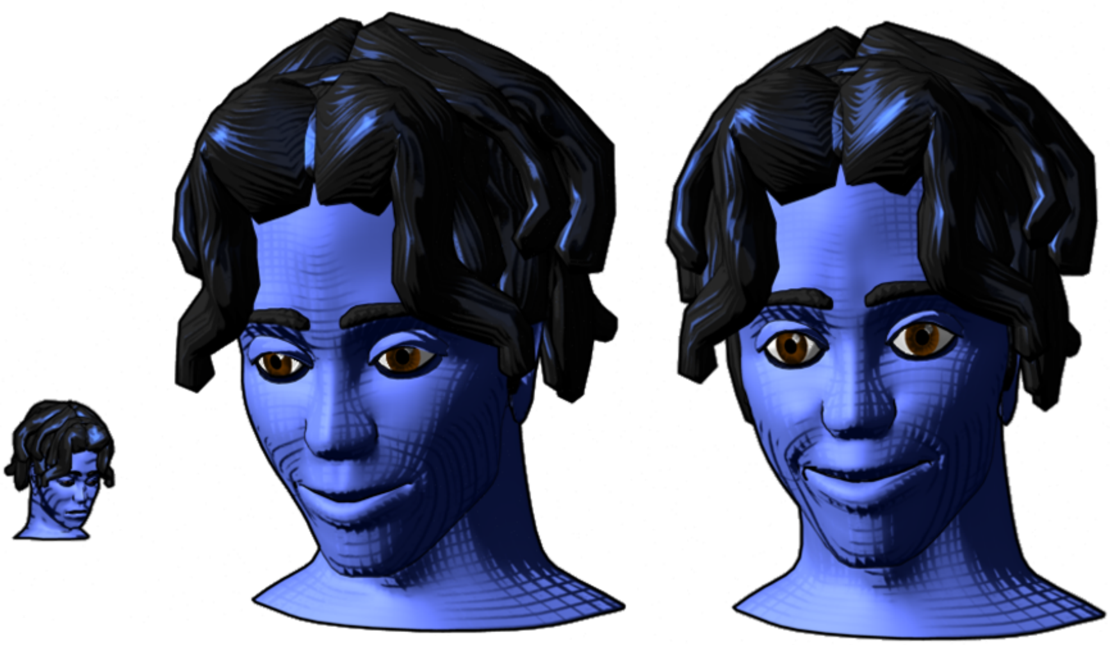

Figure 7: Procedural line hatching is used to convey discontinuities of the hair surface. The spacing of hatching lines adapts to orientation, scale, and deformation of the face model.