Journal for Geometry and Graphics
Volume 9 (2005), No. 2, 191–200.

Texture Screening Method for Fast Pencil Rendering

Ruiko Yano, Yasushi Yamaguchi

Dept. of Graphics and Computer Sciences, Graduate School of Arts and Sciences
The University of Tokyo, 3-8-1 Komaba, Meguro-ku, Tokyo 153-8902, Japan
email: {lui,yama}@graco.c.u-tokyo.ac.jp

Abstract. Several techniques for rendering pencil drawing have been developed for the last ten years. Most of them are rather expensive in the sense of computation, because they simulate the physical interaction between a pencil and drawing paper for each stroke. This paper proposes a new rendering method for synthesizing pencil-drawing images in an inexpensive manner. This method calculates pencil graphite adherence statistically rather than using physically-based simulation. It is especially suited for generating soft-shaded images without sharp strokes.

First, we observe the real pencil drawing with microscope and examine the intensity histograms. Based on this observation, we suggest a method for fast pencil rendering using *texture screen*. A texture screen represents how much pencil graphite is likely to stick to the paper surface. The intensity histogram reflects real graphite distribution tendency on the paper surface. The intensity histogram varies according to the intensity of the input pixel. The intensity of the output image is determined with the relative position in the texture screen and the intensity histogram. As a result, the output image contains real drawing-like texture.

Key Words: rendering, pencil drawing, image processing

MSC 2000: 68U05

1. Introduction

This paper is aimed at providing a fast rendering method to generating pencil-drawing styled images with soft shading. Both input and output images are grayscale raster images. We suppose no 3D shape data. We do not handle stroke directions explicitly, but instead we focus on the paper texture caused by the asperity of paper surface. The main goal of our method is to reproduce paper texture in the output image.

There are many papers on image processing to add artistic effects such as pen-and-ink style [6, 11, 12] and painterly rendering [1, 3, 4, 5]. However, a limited number of papers have focused on pencil drawing, though it is one of the most fundamental techniques of fine art.

ISSN 1433-8157/$ 2.50 © 2005 Heldermann Verlag

There are mainly two approaches to pencil-drawing rendering, namely, the methods simulating physical interaction between pencil and paper, and those based on textures. Sousa and Buchanan's method [7] is categorized into the former one. Their approach is based on the physical properties of materials and the behaviors of their interactions. The system treats the material properties such as pencil's hardness, its sharpness, paper's roughness, and so on. When a line is drawn, the amount of graphite that adhere on a paper is controlled according to these parameter values.

Takagi et al. [8, 9] proposed an algorithm for colored pencil drawing using a physical model of pencil and paper. The model consists of three sub-models, which describe in a volumetric fashion, the microstructure of paper, pigment distribution on paper, and pigment redistribution, respectively. The model takes advantage of volumetric offset distance accessibility and line integral convolution, and was said to be highly controllable with a small number of parameters. Both methods introduce materials of real pencil drawing into computation and achieve the high quality results. However, they need a large amount of computation to simulate strokes.

The second approach of texture-based rendering is more favorable when processing speed is critical. Lake et al. [4] have proposed the technique for realtime 3D animation, enabling pencil sketch shading. They made a paper texture in advance, and placed strokes to the texture. Their method succeeds in representing various tone of drawings. However, this method may be difficult to represent soft tone drawing since the method controls darkness by changing the number of strokes. Wong [13] created a system for drawing portraits. He splits a face into five parts according to their features, and applies different effect to each part. One of the parts is facial tone shading, where he added paper texture by multiplying its intensity. It seems his method has succeeded in representing soft tones though the paper texture is hardly seen in the resulting images.

Durand et al.'s approach [2] is also categorized into the latter one. They proposed the real-time drawing system which allows users to produce drawings in a variety of traditional styles including sketch-like drawing. They generated tone of drawing such as paper texture using thresholding model. The thresholding model is basically a scheme of halftoning, which generates a binary image consisting of black dots. Durand et al. applied antialiasing to the output image for eliminating the texture roughness.

2. Observation of paper surface and pencil graphite

There appears a texture caused by the asperity of paper surface when a line is drawn with a pencil laid down to the paper surface. This texture is mainly determined by the asperity of the drawing paper, as observed in the real microscopic picture of pencil graphite on paper (Fig. 1, left).

A pencil drawing consists of black and white areas, with graphite and without graphite respectively in the microscopic viewpoint. In other words, graphite sticks to the convex areas and the concave areas remain white. However, the scale of graphite particles is far smaller than the scale of a pixel when the entire drawing is represented as a digital image with a lower resolution. It is not necessary to handle graphite by a grain. It would be sufficient to handle a tendency of graphite adherence to paper statistically. Therefore, each pixel's intensity will be in proportion to the amount of graphite stuck onto the paper region corresponding to the pixel, as shown in Fig. 1 (right).

We scanned some drawing papers drawn in different darkness at 8-bit gray levels in 300 dpi

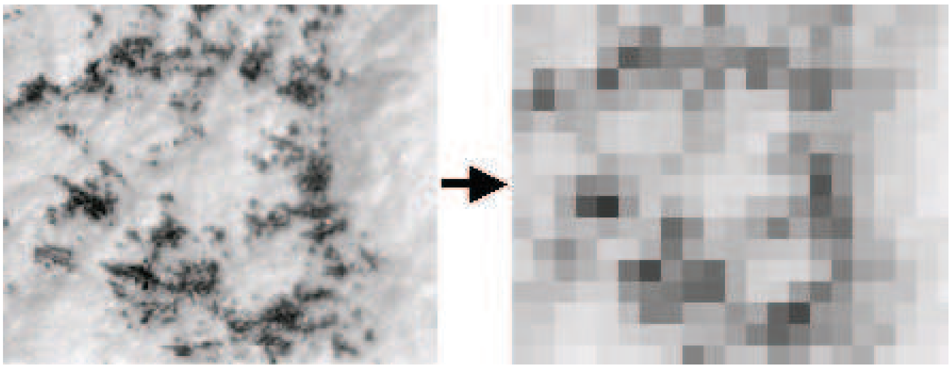

Figure 1: Raster expression of a pencil drawing

and calculated the histograms. Fig. 2 shows some of the scanned images and their histograms. We noticed the following characteristics:

1. There are no perfect black pixels, since any region corresponding to a pixel contains white parts where the pencil graphite cannot reach due to the paper asperity.
2. The shape of the histogram varies according to the darkness of drawing.

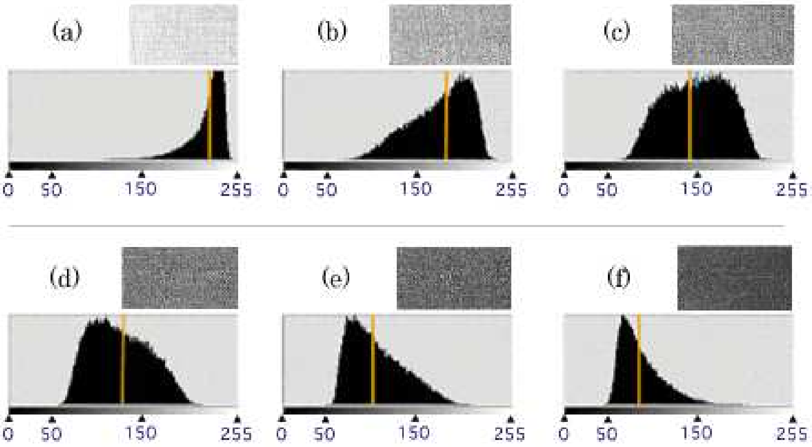

Figure 2: Intensity histograms

3. Experiments

According to the observation explained in the previous section, the texture of a pencil drawing reflects the asperity of the paper surface, when it is shaded by a soft touch. In this section, we evaluate the effects of some simple image filtering techniques. We used Fig. 3 as the original image. The method we experimented are as follows:

1. arithmetic mean,
2. multiplication of intensities,
3. multiplication of darknesses.

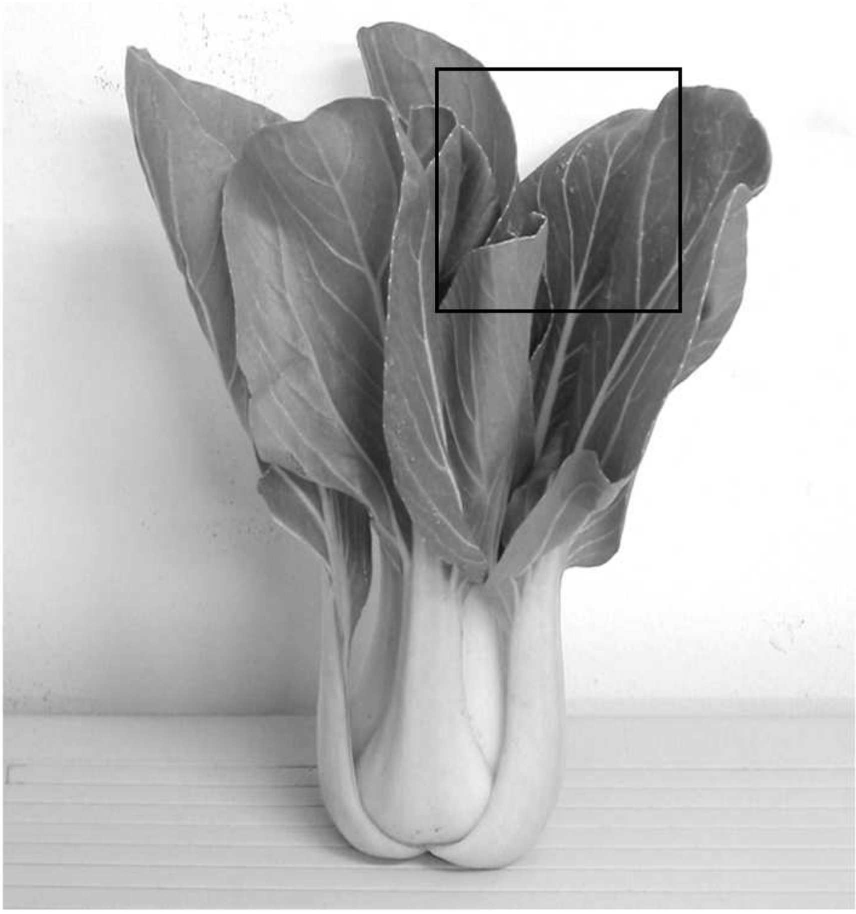

Figure 3: Original image

3.1. Arithmetic mean

The arithmetic mean is calculated by

$$\mathit{Result}(x,y) = \left(\mathit{Input}(x,y) + \mathit{Texture}(x,y)\right)/2,$$

where $\mathit{Result}(x,y)$, $\mathit{Input}(x,y)$ and $\mathit{Texture}(x,y)$ stand for the pixel intensities of the resulting image, input image, and texture respectively. Fig. 4 shows an example of a resulting image. Both the original and texture images are still observed in the resulting image because this method simply overlaps the two images. This leads to a somehow artificial impression. The artificiality is caused by the remaining details of the original image. That is, the resulting image is too photo-realistic because it is far more detailed than what people can draw with a pencil. The arithmetic mean cannot simulate a white area caused by the asperity of paper surface, which also results in the artificiality.

3.2. Multiplication of intensities

The multiplication of intensities is calculated by

$$\mathit{Result}(x,y) = \mathit{Input}(x,y) \cdot \mathit{Texture}(x,y),$$

where $\mathit{Result}(x,y)$, $\mathit{Input}(x,y)$ and $\mathit{Texture}(x,y)$ are normalized to $[0,1]$. Following this equation, a resulting pixel becomes black if one of the corresponding pixels of the original and the texture image is black. The example of this multiplication is shown in Fig. 5.

Figure 4: An experimental result of the arithmetic mean

It is obvious that the resulting image tends to be too dark, which spoils the entire tone. Generally, people focus on the object and omit the background when they draw a still life. However, the background of this resulting image is drawn too much and this causes unnaturalness. Moreover, it changes a dark area of the original image so black that the paper texture cannot be observed any more.

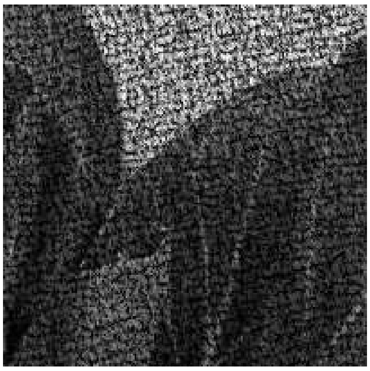

Figure 5: An experimental result of multiplication of intensities

3.3. Multiplication of darkness

Lastly, the multiplication of darknesses is calculated by

$$Result(x, y) = 1 - (1 - Input(x, y)) \cdot (1 - Texture(x, y)) \,,$$

where each pixel value is normalized to $[0, 1]$. An example of a resulting image is shown in Fig. 6. A resulting pixel becomes white if one of the corresponding pixels of the original and the texture image is white.

The result looks better than the previous ones. However, the entire image tends to be too bright. Bright parts are rendered almost white and dark parts are not rendered dark enough. This bright tone rendering ruins the style of pencil drawing.

The three methods above have each shortcomings respectively. To overcome those issues, we propose a method which controls a local pattern of pencil graphite distribution according to the intensity of the original pixel.

Figure 6: An experimental result of the multiplication of darknesses

4. Texture screening method

4.1. Intensity conversion algorithm

Our method calculates the output intensities pixel by pixel, adding textural effect to the original image. This process is similar to that of the halftoning with image-based dither screens developed by Verevka et al. [10]. Ordered dithering generates the halftoned image $P'(x,y)$ by adopting a dither screen $O(x,y)$ to the original image $P(x,y)$. This algorithm is denoted as below:

$$P'(x,y) = \begin{cases} 0 & \text{if} \;\; P(x,y) < O(x,y) \\ 1 & \text{if} \;\; P(x,y) \geq O(x,y) \end{cases} .$$

Usually, a uniformly distributed pattern is used as a dither matrix to avoid an irregularity on output images. However, by controlling threshold values of the dither matrix, we can add the intended texture to the input image. The image-based dithering is one of the application, which generates a textured binary image using arbitrary texture as the dither matrix. The thresholding model proposed by Durand et al. performs just the same as this image-based dithering. However, as discussed in Section 2, a rasterized pencil-drawing image scanned at the lower resolution is a gray-scale image rather than a binary image. Therefore, to achieve a realistic pencil-drawing effect, it is necessary to make a gray-scale output with paper texture from the input original image.

In our method, we use the intensity of a dither screen $O(x,y)$ and the intensity histogram, namely, the relative frequency of intensity $f(l)$, to generate the output image. As discussed in Section 2, the intensity histogram varies according to the target darkness which is the expected darkness of the area. The outline of our intensity conversion algorithm is shown in Fig. 7.

In the normal dithering method, the intensity of the dither screen $O(x,y)$ can be regarded as a rank representing how the resulting pixel is likely to be black. $O(x,y)$ is obtained by applying the histogram equalization to a grayscale image with paper texture. However, our method assigns a little different role to $O(x,y)$. We use it as the relative rank how much pencil graphite will stick to that pixel. Since our method is not for dithering anymore, we call $O(x,y)$ *texture screen* from now on.

The algorithm converts the image in the pixelwise manner. A pixel of the original image, $P(x,y)$, is regarded as the target darkness at the point (x,y). The output intensity can be computed with the intensity histogram $f(l)$ whose expectation is $P(x,y)$. A relative rank of intensity l is given by the accumulated relative frequency $g(l) = \int_0^l f(t)dt$.

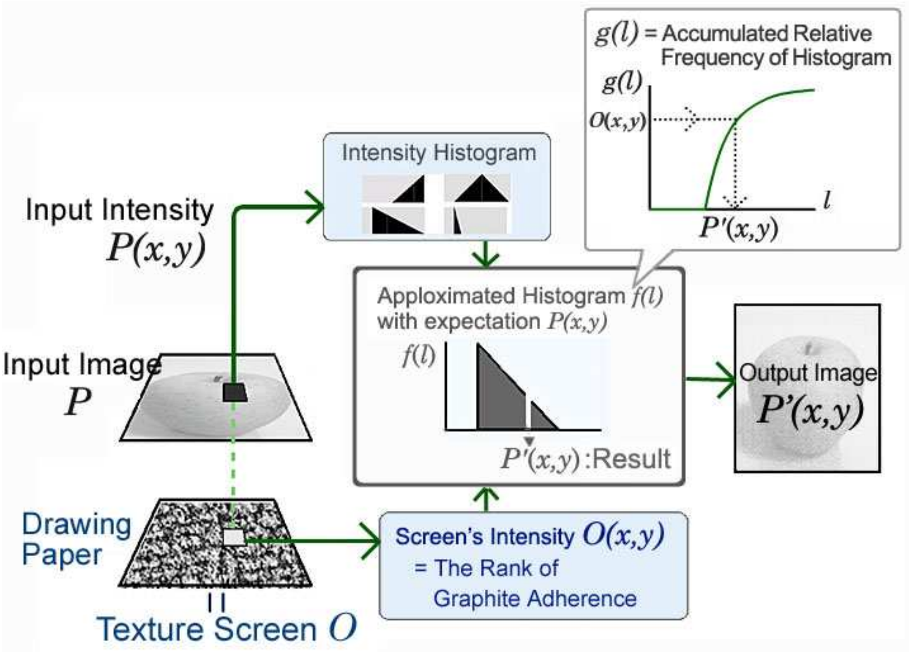
g(l) = Accumulated Relative Frequency of Histogram
g(l)
O(x,y)
l
P'(x,y)
Intensity Histogram
Input Intensity
P(x,y)
Apploximated Histogram f(l) with expectation P(x,y)
f(l)
P'(x,y) :Result
Input Image
P
Output Image
P'(x,y)
Drawing Paper
Texture Screen O
Screen's Intensity O(x,y)
= The Rank of Graphite Adherence

Figure 7: Outline of the conversion algorithm

On the other hand, the intensity of the corresponding pixel in $O(x, y)$ stands for the relative rank of the point (x, y) representing how much the point is likely to get dark due to the paper asperity. The output intensity of the point $P(x, y)$ fulfills the following equation:

$$g\left(P'(x, y)\right) = O(x, y).$$

Thus $P'(x, y)$ is determined as below:

$$P'(x, y) = g^{-1}\left(O(x, y)\right).$$

4.2. Creating texture and histogram

In our method, the intensity histogram $f(l)$ is necessary in advance. We introduce a model which approximates some experimental results because it is difficult to make all the histograms of entire intensities by experiments only. We scanned paper surfaces drawn at some darkness and checked their intensity histogram as in Fig. 2. We approximated them by the piecewise linear function. The approximated histogram has three patterns of form according to its expectation, namely, (a)–(c), (c)–(g), and (g)–(i), in Fig. 8.

A texture screen must be correctly ranked by the graphite adherence. We spread the graphite all over using the pencil laid down on the drawing paper. The resolution is set to 200dpi and the quantization level is 16bit-split. The left side of Fig. 9 is the scanned image. Then we applied the histogram equalization to the scanned image. The right side of Fig. 9 is the histogram equalized image which we used as a texture screen.

5. Result

We implemented the algorithm and generated images with drawing-like texture. Fig. 10 is the resulting image of a vegetable generated from Fig. 3. The size of the image is 1500 × 1500

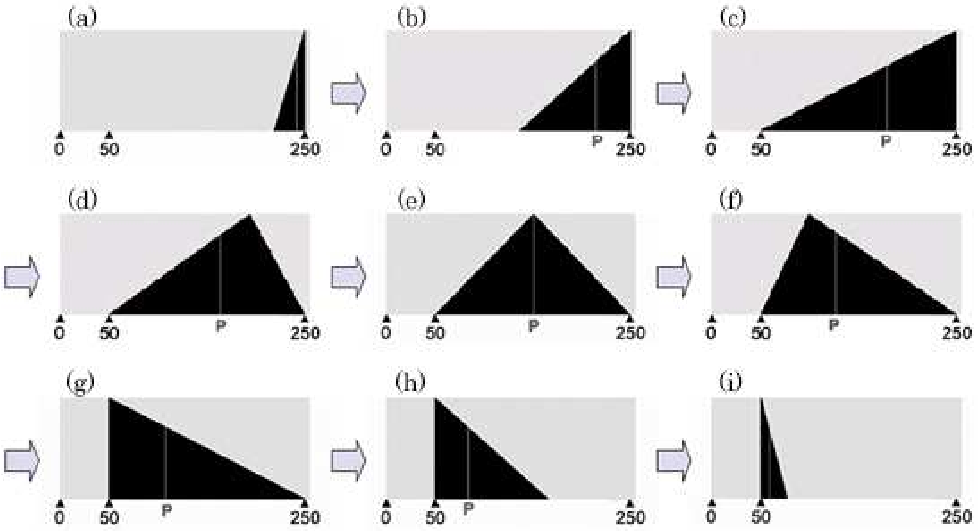

Figure 8: Examples of approximated histograms

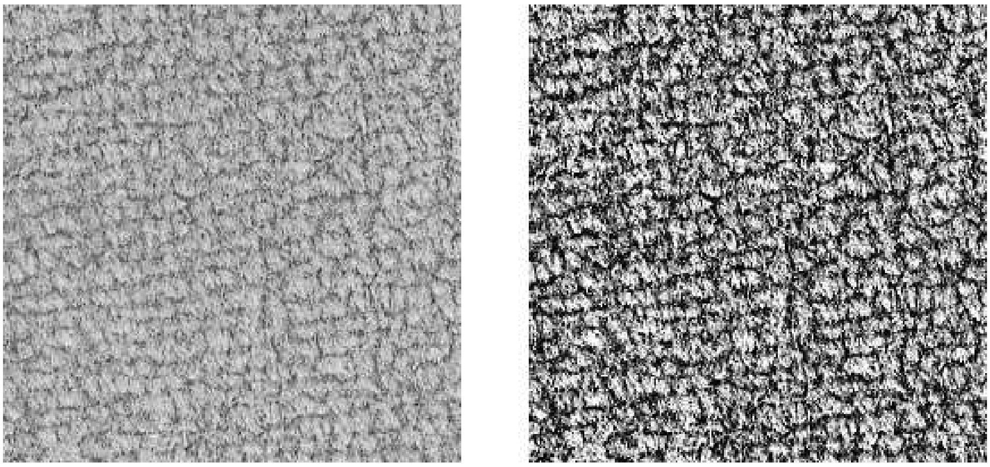

Figure 9: A scanned image (left) and a texture screen (right)

pixel. We implemented the algorithm by Java and the calculation time was 0.53 seconds on Pentium4 1.4 GHz (about 23 seconds including file input/output). For comparison with the experiments explained in Section 3, a magnified image of the same area is depicted in Fig. 11.

The resulting image appears to be successful in making soft mood of pencil shading. Especially, the left part of the image in Fig. 10 is well rendered. It looks that the three patterns of histograms sufficiently reflect the relative rank of intensities for target darknesses.

However, the right edges of the vegetable's leaves are too sharp. The right part of the image looks artificial because of this sharpness. Focusing on the bottom side, stems of the vegetable are rendered too soft. This is caused by a characteristic of our method that the same histogram is used for the same input intensity. In the case of the actual drawing, it would be possible to make an object and background distinctive by adding edges or changing directions of strokes.

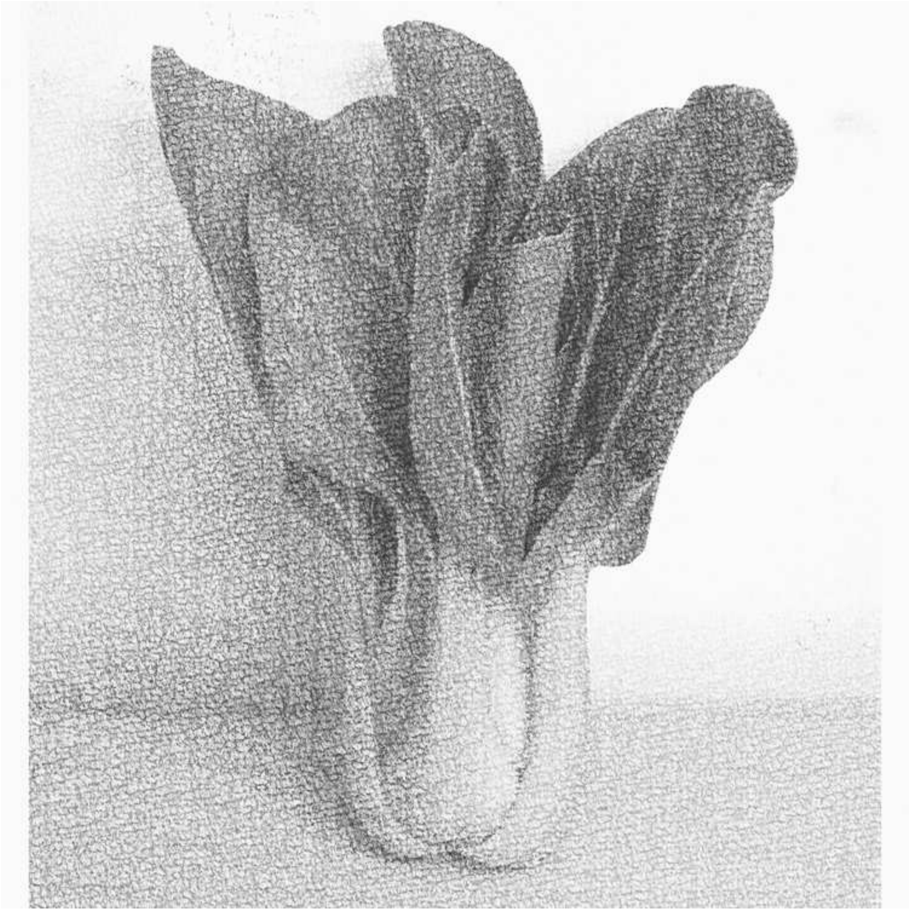

Figure 10: An example of results

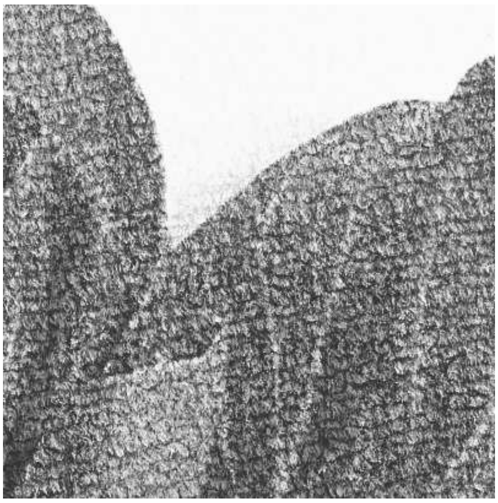

Figure 11: The magnified resulting image

6. Conclusion and future work

This paper proposes a fast algorithm for the pencil-drawing rendering. Our method uses a texture screen which is based on the scanned asperity of the paper surface. The distribution of graphite adherence is simulated with the intensity histogram which is unique to the intensity of the input image. The results are very pleasing, especially for the soft-shaded pencil drawing. Since our method enables pencil drawing rendering at rather fast computation speed, it may be applied to animations with many frames.

References

[1] C.J. Curtis, S. E. Anderson, J.E. Seims, K.W. Fleischer, D.H. Salesin: *Computer-generated watercolor.* Computer Graphics **31** (Proceedings of SIGGRAPH '97), 421–430 (1997).

[2] F. Durand, V. Ostromoukhov, M. Miller, F. Duranleau, J. Dorsey: *Decoupling strokes and high-level attributes for interactive traditional drawing.* Proceedings of Eurographics Workshop on Rendering, 2001.

[3] A. Hertzmann: *Painterly rendering with curved brush strokes of multiple sizes.* Computer Graphics **32** (Proceedings of SIGGRAPH '98), 453–460 (1998).

[4] A. Lake, C. Marshall, M. Harris, M. Blackstein: *Stylized rendering techniques for scalable real-time 3d animation.* Proceedings of NPAR 2000, 13–20 (2000).

[5] B.J. Meier: *Painterly rendering for animation.* Computer Graphics **30** (Proceedings of SIGGRAPH '96), 477–484 (1996).

[6] M.P. Salisbury, M.T. Wong, J. F. Hughes, D.H. Salesin: *Orientable textures for image-based pen-and-ink illustration.* Computer Graphics **31** (Proceedings of SIGGRAPH '97), 401–406 (1997).

[7] M.C. Sousa, J.W. Buchanan: *Computer-generated graphite pencil rendering of 3D polygonal models.* Computer Graphics Forum **18**, no. 3, 195–207, 1999).

[8] S. Takagi, I. Fujishiro, M. Nakajima: *Volumetric modeling of artistic techniques in colored pencil drawing.* Conference Abstracts and Applications of ACM SIGGRAPH '99, 283 (1999).

[9] S. Takagi, I. Fujishiro, M. Nakajima: *Volumetric modeling of colored pencil drawing.* Proceedings of Pacific Graphics '99, 250–258 (1999).

[10] O. Verevka, J. Buchanan: *Halftoning with image-based dither screens.* Proceedings of Graphics Interface '99, 167–174 (1999).

[11] G. Winkenbach, D.H. Salesin: *Computer-generated penandink illustration.* Computer Graphics **28** (Proceedings of SIGGRAPH '94), 91–100 (1994).

[12] G. Winkenbach, D.H. Salesin: *Rendering parametric surfaces in pen and ink.* Computer Graphics **30** (Proceedings of SIGGRAPH '96), 469–476 (1996).

[13] E.C. Wong: *Artistic Rendering of Portrait Photographs.* Master's thesis, Cornell University 1999.

Received August 1, 2004; final form August 3, 2005