Combining Sketch and Tone for Pencil Drawing Production

Cewu Lu Li Xu Jiaya Jia
Department of Computer Science and Engineering
The Chinese University of Hong Kong
{cwlu, xuli, leojia}@cse.cuhk.edu.hk

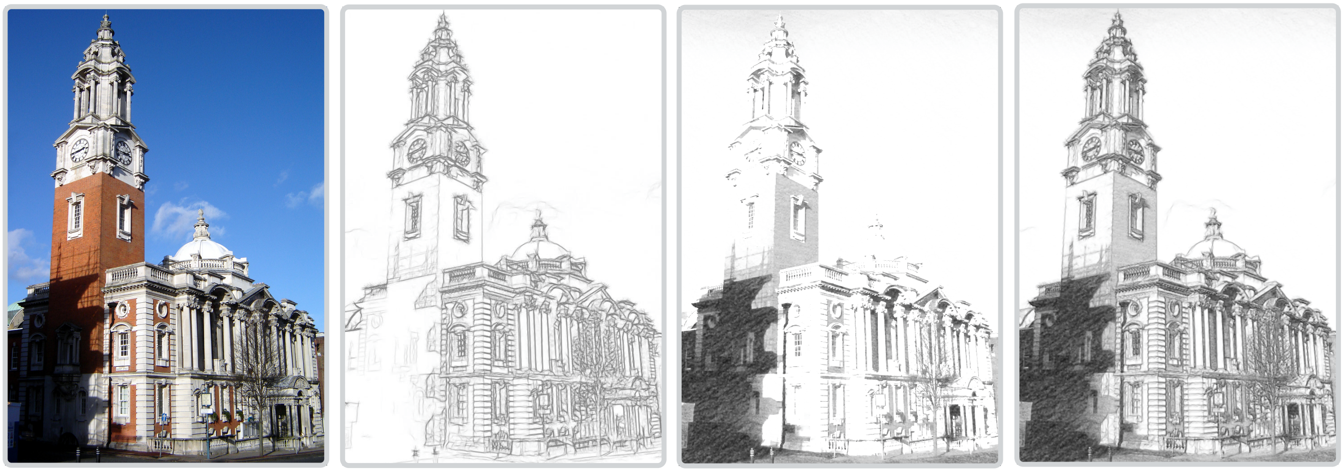


Figure 1: *Given a natural image (shown in (a)), our method has two main steps to produce the stroke layer and tonal texture, as shown in (b)-(c). They naturally represent respectively shape and shading. Our final result (d) is an algorithmic combination of these two maps, containing important structure information and mimicking a human drawing style.*

Abstract

We propose a new system to produce pencil drawing from natural images. The results contain various natural strokes and patterns, and are structurally representative. They are accomplished by novelly combining the tone and stroke structures, which complement each other in generating visually constrained results. Prior knowledge on pencil drawing is also incorporated, making the two basic functions robust against noise, strong texture, and significant illumination variation. In light of edge, shadow, and shading information conveyance, our pencil drawing system establishes a style that artists use to understand visual data and draw them. Meanwhile, it lets the results contain rich and well-ordered lines to vividly express the original scene.

Keywords: pencil drawing, non-photorealistic rendering, tonal drawing

1 Introduction

Pencil drawing is one of the most fundamental pictorial languages in visual arts to abstract human perception of natural scenes. It establishes the intimate link to artists' visual record. Pencil drawing synthesis methods generally fall into two categories: 2D image-based rendering and 3D model-based rendering. The majority of recent research along the line to mimic artistic drawing styles and produce perceptually expressive results resorts to using 3D models [Sousa and Buchanan 1999a; Lee et al. 2006]. With the popularity of digital cameras and internet sharing, obtaining high-quality pictures is much easier than constructing 3D models of a scene. Consequently, there has been a substantial increase in the demand of rendering pencil drawing from natural images.

In the literature, pencil drawing can be classified into a few styles. *Sketch* typically refers to a quickly finished work without a lot of details. Artists often use sketches to depict the global shape and main contours. *Hatching*, on the other hand, is used to depict tone or shading by drawing dark and parallel strokes in different regions.

While there has been work to produce line drawings [DeCarlo et al. 2003; Judd et al. 2007; Lee et al. 2007; Grabli et al. 2010], hatching [Hertzmann and Zorin 2000; Praun et al. 2001], and a combination of them [Lee et al. 2006], it is still difficult for many methods to produce comparable results to those with 3D model guidance. It is because accurately extracting and manipulating structures, which is almost effortless using 3D models with known boundary and geometry, becomes challenging due to the existence of texture, noise, and illumination variation. For example, many existing approaches simulate strokes based on gradients and edges. However, edge detectors often produce segments mistakenly broken in the middle or containing spurious noise-like structures, unlike the strokes drawn by artists. Further, without surface normal or lighting information, automatic hatching generation becomes difficult. Adding directional hatching patterns using closely placed parallel lines risks laying them across different surfaces and depth layers, causing unnatural drawing effects.

Based on the fact that the hatch and tone information is paramount in making automatic pencil drawing visually appealing, as shown in Fig. 1, and on the inherent difficulty to properly make use of it – as described above – we novelly propose a two-stage system with important stroke and tone management and make the following contributions.

Firstly, to capture the essential characteristics of pencil sketch and simulate rapid nib movement in drawing, we put stroke generation

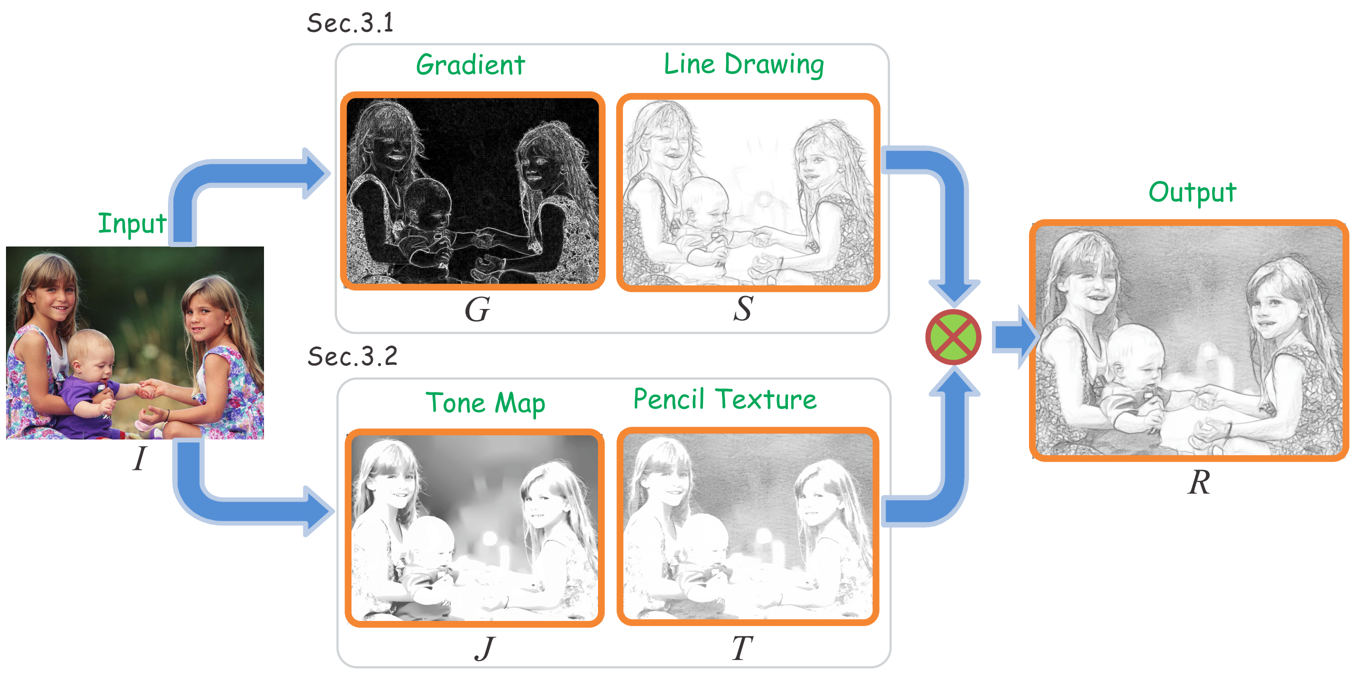


Figure 2: *Overview of our pencil drawing framework.*

into a convolution framework, which differs our method from existing approaches [Judd et al. 2007; Grabli et al. 2010]. Secondly, to avoid artifacts caused by hatching, we bring in tonal patterns consisting of dense pencil strokes without dominant directions. Parametric histogram models are proposed to adjust the tone, leveraging the statistics from a set of sketch examples. Finally, an exponential model with global optimization is advocated to perform tone adjustment, notably benefiting rendering in heavily textured regions and object contours. The resulting pencil drawing combines sketchy strokes and tonal drawings, which well characterize and approximate the common two-step human drawing process [Wang 2002]. All these steps in our framework play fundamental roles in constructing a simple and yet very effective drawing system with a single natural image as input.

Our method can be directly applied as well to the luminance channel of the color input to generate color pencil drawing. Effectiveness of the proposed method is borne out by extensive experiments on various types of natural inputs and comparison with other approaches and commercial software.

2 Related Work

Sizable research was conducted to approximate artistic styles, such as pen and ink illustration [Winkenbach and Salesin 1994; Winkenbach and Salesin 1996; Salisbury et al. 1997], graphite and colored pencil drawing [Takagi et al. 1999], oil painting [Hertzmann 1998; Zeng et al. 2009], and watercolor [Curtis et al. 1997; Bousseau et al. 2006]. We briefly review prior work on pencil sketching, hatching and line drawing, which are main components of pencil drawings. Other non-photorealistic rendering methods were reviewed in [Strothotte and Schlechtweg 2002].

Model-based Sketching Sousa and Buchanan [1999a] simulated pencil drawing by studying the physical properties. Outlines are drawn on every visible edge of the given 3D model followed by texture mapping onto the polygon to represent shading. Interaction is required in this process. Lee et al. [2006] detected contours from a 3D surface and perturbed them to simulate human drawing. Pencil textures with directional strokes are generated to express shading. It relies on lighting and material in the 3D models. For line drawing, majority of the methods also based their input on 3D models [DeCarlo et al. 2003; Judd et al. 2007; Lee et al. 2007; Grabli et al. 2010] and described how to extract crease and suggestive contours to depict shape. A comprehensive review of line drawing from 3D models is given in [Rusinkiewicz et al. 2005].

For expression of shape and shading, in [Lake et al. 2000], pencil shading rendering was proposed. Different hatching styles were defined and analyzed in [Hertzmann and Zorin 2000]. Praun et al. [2001] achieved hatching in real time. Visually very compelling results can be yielded as 3D models provide detailed geometry information. It is however still unknown how to directly apply these approaches to image-based rendering where structural edges and illumination are generally unavailable.

Image-based Sketching Sousa and Buchanan [1999b] developed an interactive system for pencil drawing from images or drawing in other styles. In [Chen et al. 2004], portrait sketch was proposed with the style of a particular artistic tradition or of an artist. Durand et al. [2001] presented an interactive system, which can simulate a class of styles including pencil drawing.

Mao et al. [2001] detected local structure orientation from the input image and incorporated Linear Integral Convolution to produce the shading effect. Yamamoto et al. [2004] divided a source image into intensity layers in different ranges. The region-based LIC [Chen et al. 2008] alternatively considers superposition of edges, unsharp masked image, and texture. Li and Huang [2003] used the feature geometric attributes obtained from local-region moments and textures to simulate pencil drawing. Another LIC-based method was introduced in [Gao et al. 2010].

For line extraction, Kang et al. [2007] proposed computing an edge tangent flow (ETF) and used an iterative directional difference-of-Gaussian (DoG). In [Son et al. 2007], line extraction is based

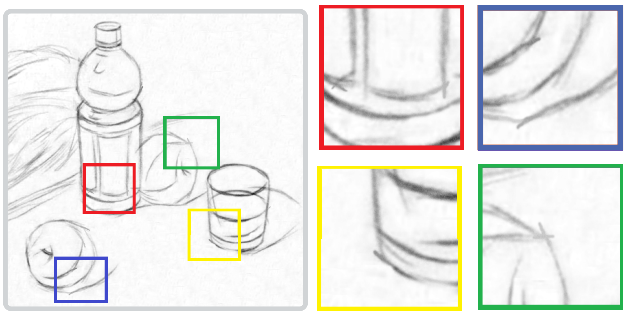

(a) artist-created pencil sketch (b) close-ups

Figure 3: *Artist-produced pencil sketch.*

on likelihood-function estimation with the consideration of feature scales and line blurriness. Gaussian scale space theory was employed in [Orzan et al. 2007] to compute a perceptually meaningful hierarchy of structures. Zhao et al. [2008] adopted mathematical morphology thinning and formed stroke paths from pixels. Level-of-detail of lines is controlled in rendering. Image analogies [Hertzmann et al. 2001] could also produce pencil sketch results from images. But extra training image pairs are required. Finally, Xu et al. [2011] advocated an $L0$ smoothing filter for suppressing noise as well as enhancing main structures. It is applicable to a few efforts including pencil stroke generation.

Note previous work in general determines orientation of strokes and hatches based on local structures, which might be unstable for highly textured or noisy regions. Winnemoeller et al. [Winnemöller et al. 2006] applied line extraction on bilaterally smoothed images to reduce noise. DeCarlo and Santella [DeCarlo and Santella 2002] adopted region simplification to eliminate superfluous details.

3 Our Approach

Our image-based pencil sketching approach consists of two main steps, i.e., pencil stroke generation and pencil tone drawing. Their effects complement each other. Specifically, stroke drawing aims at expressing general structures of the scene, while the tone drawing focuses more on shapes, shadow, and shading than on the use of lines. The latter procedure is effective to perceptually depict global illumination and accentuate shading and dark regions. The framework is illustrated in Fig. 2.

It is noteworthy that there are various styles in pencil drawing. Our framework generates one that includes fundamental and important components for natural and perceptually expressive representation.

3.1 Line Drawing with Strokes

Strokes are the vital elements in line drawing. Artists use strokes with varying thickness, wiggliness, or brightness [Hsu and Lee 1994; Curtis 1998]. An important observation is that artists cannot always draw very long curves without any break. Strokes end often at points of curvature and junctions. Consequently, there might be crosses at the junction of two lines in human drawing, caused by rapid hand movement. These are critical evidences for human-drawn strokes. Fig. 3(a) shows artist-created pencil sketch. In the close-ups (b), broken strokes with cross-like effect exist at junctions.

Based on this fact, unlike previous methods [Lee et al. 2006; Judd et al. 2007; Grabli et al. 2010] using continuous functions, e.g., Sinusoid function, to bend strokes, we take another approach to put a set of less lengthy edges to approximate sketchy lines. Intuitively, when drawing strokes at a point, we determine the direction, length, and width in a pixel classification and link process based on a unified convolution framework.

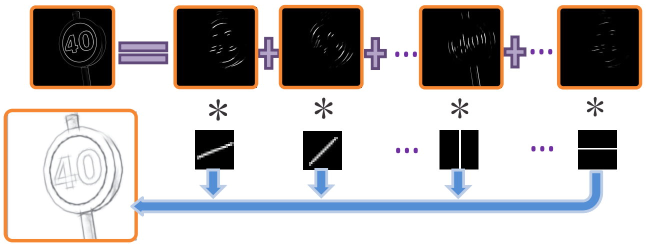


Figure 4: *Line drawing with strokes.*

Classification We compute gradients on the grayscale version of the input, yielding magnitude

$$G = \left((\partial_x I)^2 + (\partial_y I)^2\right)^{\frac{1}{2}}, \tag{1}$$

where I is the grayscale input. ∂_x and ∂_y are gradient operators in two directions, implemented by forward difference. We note that the gradient maps for natural images are typically noisy and do not contain continuous edges immediately ready for stroke generation.

One issue of generating short lines for sketch is the estimation of line direction for each pixel. Naive classification to this end uses the gradient direction, which is sensitive to noise and thus is unstable. We propose a more robust strategy using local information.

In the first place, we choose eight reference directions at 45 degrees apart and denote the line segments as $\{\mathcal{L}_i\}$, $i \in \{1 \ldots 8\}$. The response map for a certain direction is computed as

$$G_i = \mathcal{L}_i * G, \tag{2}$$

where $\mathcal{L}_i$ is a line segment at the i^{th} direction, which is deemed as a convolution kernel. The length of $\mathcal{L}_i$ is set to $1/30$ of the image height or width, empirically. $*$ is the convolution operator, which groups gradient magnitudes along direction i to form the filter response map G_i. The classification is performed by selecting the maximum value among the responses in all directions and is written as

$$C_i(p) = \begin{cases} G(p) & \text{if } \operatorname{argmin}_i\{G_i(p)\} = i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where p indexes pixels and C_i is the magnitude map for direction i. Fig. 4 shows the C maps. Their relationship with G is expressed as $\sum_{i=1}^{8} C_i = G$ and is illustrated in the top row of Fig. 4. We note this classification strategy is very robust against different types of noise. A pixel can be correctly classified into a group C given the support of neighboring gradients and structures. It does *no matter* whether the gradient of the current pixel is noise contaminated or not.

Line Shaping Given the map set $\{C_i\}$, we generate lines at each pixel also by convolution. It is expressed as

$$S' = \sum_{i=1}^{8} (\mathcal{L}_i \otimes C_i).$$

Convolution aggregates nearby pixels along direction $\mathcal{L}_i$, which links edge pixels that are even *not* connected in the original gradient map. This process is also very robust to noise and other image visual artifacts. The final pencil stroke map S is obtained by

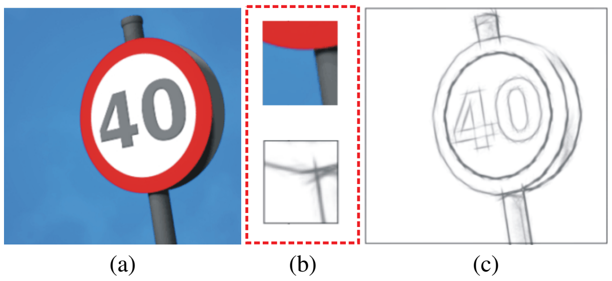


Figure 6: *Pencil sketch line generation. (a) Input color image. (c) Output S. Close-ups are shown in (b).*

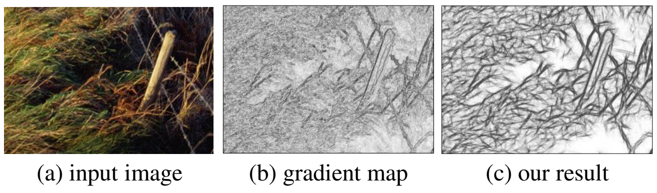


Figure 7: *Our method can handle texture very well. It does not produce noise-like texture patterns contained in the gradient map.*

inverting pixel values and mapping them to $[0,1]$. This process is demonstrated in Fig. 4. An automatically generated stroke map S is shown in Fig. 6. The close-ups indicate that our method captures well the characteristics of hand-drawn sketchy lines shown in Fig. 3.

Discussion Compared with other line drawing methods, our approach contributes in generating strokes taking special consideration of edge shape and neighborhood pixel support, capturing the sketch nature.

- Using lines to simulate sketch is surprisingly effective. The convolution step extends the ends of two lines at the junction point. Note that only long straight lines in the original edge map would be significantly extended as directional convolution only aggregates pixels along strictly straight lines.
- Pixels at the center of a long line would receive pixel values from both sides, which make the line center darker than the ends, desirable in pencil drawing.
- Our strategy helps link pixels that are not necessarily connected in the original edge map when nearby pixels are mostly aligned along the straight line, imitating human line drawing process and resisting noise.

Fig. 5 shows one comparison with the method of Son et al. [2007], where strokes are generated based on local gradients. Our result, shown in (c), demonstrates a different style more like human pencil drawing.

Another notable benefit of the proposed stroke generation is its strong ability to handle texture. Natural images contain many textured surfaces with fine details, such as leaves and grass. Their gradients are usually noisy, as shown in Fig. 7(b). It is difficult to connect these noise-like texture into continuous lines. Also it is not practical to find one dominant direction to generate strokes using the method of Son et al. [2007]. Our result in Fig. 7(c), thanks to the classification and link process, contains reliable strokes from the originally noisy gradient field.

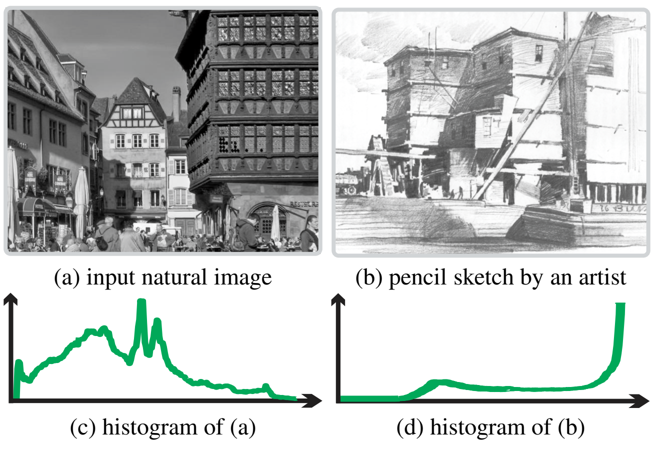


Figure 8: *Tone distributions in a natural image and in a pencil sketch are quite different. Although they are not about the same scene, similar edge structures can be observed.*

3.2 Tone Drawing

Artists also use dense strokes, such as hatching, to emphasize shadow, shading, and dark objects. Besides line strokes, our system is also equipped with a new scheme to render pencil texture

Tone Map Generation We first determine the tone value for each pixel leveraging information in the original grayscale input. Previous methods [Li and Huang 2003; Yamamoto et al. 2004] used image tones to generate hatching patterns. We found this process, for many cases, is not optimal because the tone distribution of a grayscale image generally differs significantly from that of pencil sketch. One example is shown in Fig. 8(a)-(b). The image tone thus should not be directly used and further manipulation is critically needed.

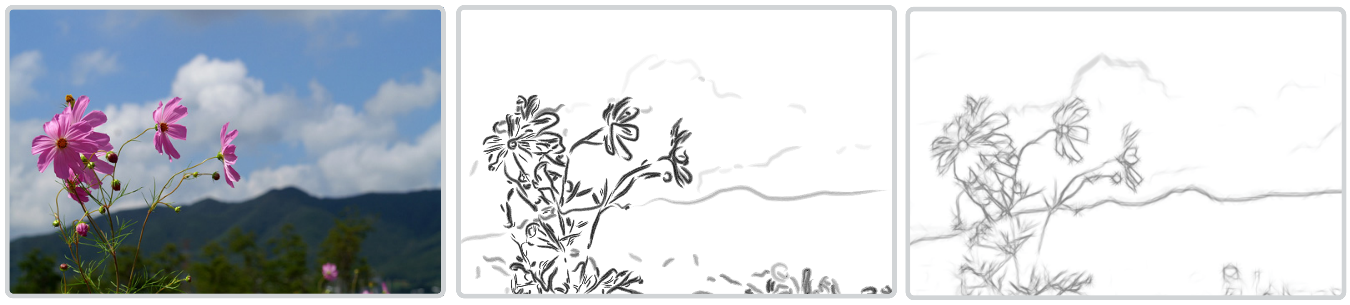
We propose a parametric model to fit tone distributions of pencil sketch. Unlike the highly variable tones in natural images, a sketch tone histogram usually follows certain patterns. It is because pencil drawings are the results of interaction of paper and graphite, which mainly consist of two basic tones. For very bright regions, artists do not draw anything to show white paper. Heavy strokes, contrarily, are used to accentuate boundaries and highlight dark regions. In between these two tones, mild tone strokes are produced to enrich the layer information.

Model-based Tone Transfer With the above findings, we propose a parametric model to represent the target tone distribution, written as

$$p(v) = \frac{1}{Z}\sum_{i=1}^{3} \omega_i p_i(v), \qquad (4)$$

where v is the tone value and $p(v)$ gives the probability that a pixel in a pencil drawing is with value v. Z is the normalization factor to make $\int_0^1 p(v)dv = 1$. The three components $p_i(v)$ account for the three tonal layers in pencil drawing, and ωs are the weights coarsely corresponding to the number of pixels in each tonal layer. We scale the values into the range $[0,1]$ to cancel out illumination difference in computing the distributions.

Fig. 9 shows the layers and respective tone distributions. Given an artist-drawn pencil sketch shown in (a), we partition pixels into three layer, according to their values (details are given later when



(a) input image (b) [Son et al. 2007] (c) our result

Figure 5: *Comparison of stroke generation.*

discussing parameter learning). They are highlighted in green, orange, and blue in (b). (c) gives the tone distributions of the three layers, in accordance with our observation. Specifically, the bright and dark layers have obvious peaks while the mild tone layer does not correspond to a unique peak due to the nature of contact between black graphite and white papers.

Consequently, a major difference between natural images and pencil drawings is that the latter consists of more brighter regions, which is the color of the paper. To model the tone distribution, we use the Laplacian distribution with its peak at the brightest value (255 in our experiments) because pixels concentrate at the peak and number decreases sharply. The small variation is typically caused by the using of eraser or slight illumination variation. We thus define

$$p_1(v) = \begin{cases} \frac{1}{\sigma_b} e^{-\frac{1-v}{\sigma_b}} & \text{if } v \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where σ_b is the scale of the distribution.

Unlike the bright layer, the mild tone layer does not necessarily peak at a specific gray level. Artists usually use many strokes with different pressure to express depth and different levels of details. We simply use the uniform distribution and encourage full utilization of different gray levels to enrich pencil drawing. It is expressed as

$$p_2(v) = \begin{cases} \frac{1}{u_b - u_a} & \text{if } u_a \le v \le u_b \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

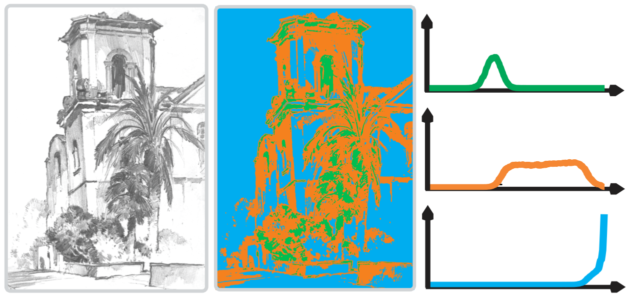
where u_a and u_b are two controlling parameters defining the range of the distribution.

Finally, dark strokes emphasize depth change and geometric contours of objects. We model them as

$$p_3(v) = \frac{1}{\sqrt{2\pi}\sigma_d} e^{-\frac{(v-\mu_d)^2}{2\sigma_d^2}}, \tag{7}$$

where μ_d is the mean value of the dark strokes and σ_d is the scale parameter. The variation of pixel values in the dark layer is in general much larger than that in the bright layer.

Parameter Learning The controlling parameters in Eqs. (5)-(7) actually determine the shape of the tone histogram. For different pencil drawings, they may differ largely. It is also why the extracted edges do not simply give a pencil-sketch feeling. In practice, we collect a number of pencil sketches with similar drawing styles to fit the parameters. Examples are in Fig. 10. Using parametric models is greatly advantageous in resisting different types of noise and visual artifacts.



(a) (b) (c)

Figure 9: *Illustration of composition of three tone layers and their value distributions taking an artist-drawn pencil sketch as an example. The green, orange, and blue pixels (and plots) belong to the dark, mild-tone, and bright layers.*

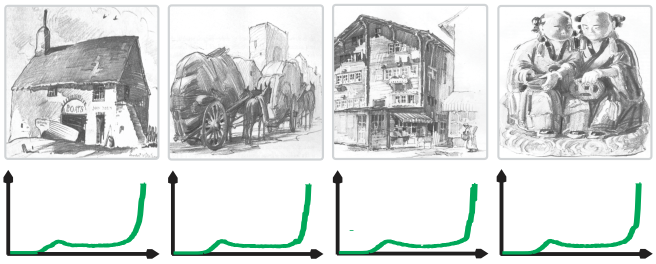


Figure 10: *Four examples and their corresponding tone distributions.*

Initially, the grayscale input I is slightly Gaussian smoothed. Then the parameter estimation process follows, in which the bright and dark tone layers are manually indicated with threshold of intensities and the remaining pixels are in the mild-tone layer. The number of pixels in each layer decides weights ω. For each layer, the parameters are estimated using Maximum Likelihood Estimation (MLE). Denoting the mean and standard deviation of the pixel values as m and s in each layer, the parameters have the closed-form representation, written as

$$\sigma_b = \frac{1}{N}\sum_{i=1}^{N}|x_i - 1|, \quad u_a = m_m - \sqrt{3}s_m, \quad u_b = m_m + \sqrt{3}s_m,$$
$$\mu_d = m_d, \quad \sigma_d = s_d,$$

where x_i is the pixel value and N is the number of pixels in the layer.

ω_1	ω_2	ω_3	σ_b	u_a	u_b	μ_d	σ_d
11	37	52	9	105	225	90	11

Table 1: *Learned parameters.*

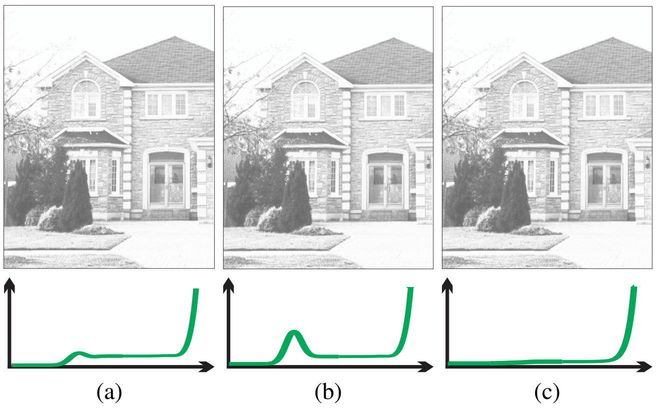


Figure 11: *Three results with (a) $\omega_1 : \omega_2 : \omega_3 = 11 : 37 : 52$, (b) $\omega_1 : \omega_2 : \omega_3 = 29 : 29 : 42$, and (c) $\omega_1 : \omega_2 : \omega_3 = 2 : 22 : 76$. The corresponding value distributions are plotted in the second row.*

The finally estimated parameters are listed in Table 1. Based on the parametric p_1, p_2, and p_3, for each new input image, we adjust the tone maps using simple histogram matching in all the three layers and superpose them again. We denote by J the finally tone adjusted image of input I. Note that the weights ω_1, ω_2, and ω_3 in Table 1 can vary to produce results in dark and light styles. One example is shown in Fig. 11.

Pencil Texture Rendering Generating suitable pencil textures for images is difficult. Tonal texture refers to pencil patterns without obvious direction, which reveal only the tone information. We utilize the computed tone maps in our framework and learn human-drawn tonal patterns, as illustrated in Fig. 12.

We collect a total of 20 tonal textures from examples in rendering. One input image only needs one pattern. In human drawing, tonal pencil texture is generated by repeatedly drawing at the same place. We simulate the process using multiplication of strokes, which results in an exponential combination $H(x)^{\beta(x)} \approx J(x)$ (or in the logarithm domain $\beta(x)\ln H(x) \approx \ln J(x)$), corresponding to drawing pattern H β times to approximate the local tone in J. Large β would darken more the image, as shown in Fig. 12. We also require β to be locally smooth. It is thus computed by solving

$$\beta^* = \arg\min_{\beta} \|\beta \ln H - \ln J\|_2^2 + \lambda \|\nabla \beta\|_2^2, \qquad (8)$$

where λ is the weight with value 0.2 in our experiments. Eq. (8) transforms to a standard linear equation, which can be solved using conjugate gradient. The final pencil texture map T is computed through an exponential operation

$$T = H^{\beta^*}. \qquad (9)$$

Fig. 13 shows a comparison with the LIC-based method [Yamamoto et al. 2004], which uses directional hatching patterns. Ours, contrarily, has tonal textures.

3.3 Overall Framework

We combine the pencil stroke S and tonal texture T by multiplying the stroke and texture values for each pixel to accentuate important

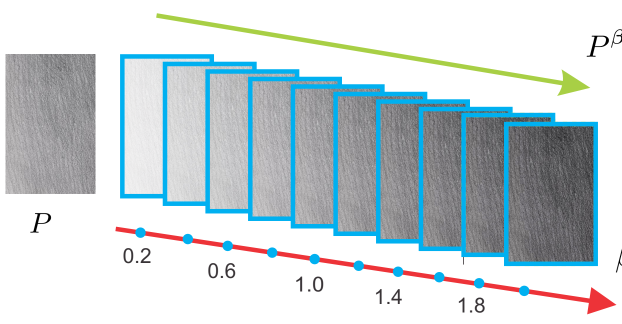


Figure 12: *Human drawn tonal pattern p. It does not have a dominant direction. Varying β changes density in our method.*

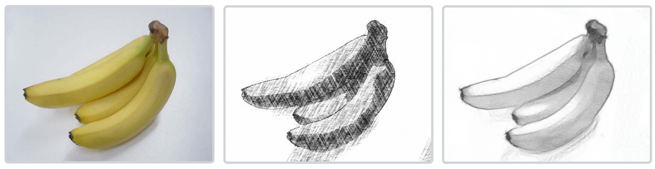


Figure 13: *Result comparison with the LIC method.*

contours, expressed as

$$R = S \cdot T. \qquad (10)$$

Note that the tonal texture is instrumental to make the drawing perceptually acceptable, by adding shading to originally textureless regions. One example is shown in Fig. 1.

3.4 Color Pencil Drawing

We extend our method to color pencil drawing by taking the generated grayscale pencil sketch R as the brightness layer, i.e., the Y channel in the YUV color space, and re-maping YUV back to the *rgb* space. Figs. 1 and 14 show two examples. As our tone adjustment is effective to produce pencil drawing like appearance, the final color drawing also presents proper tones.

4 Results and Comparison

We compare our results with those produced by several representative image-based pencil sketching approaches and commercial software. Fig. 15 gives the comparison with the method of Sousa and Buchanan [1999b]. The input image (a) and result (b) are provided in the original paper. Our result, shown in (c), presents a different drawing style – not only the contours are highlighted by sketchy

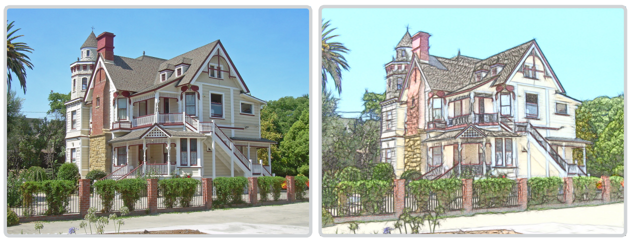


Figure 14: *Color pencil drawing example.*

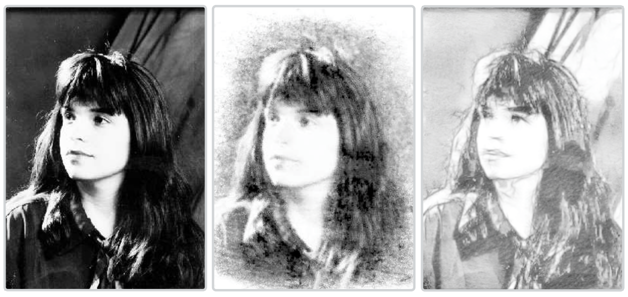
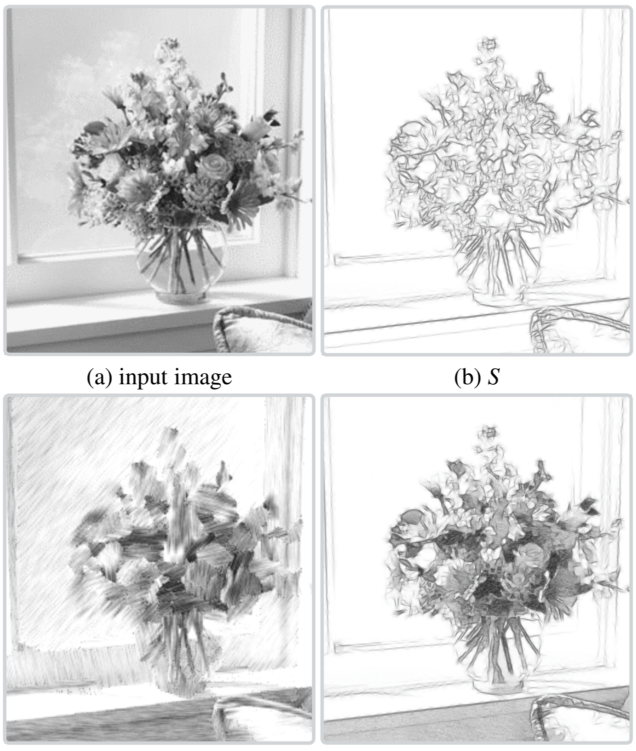

(a) input image (b) [1999b] (c) our method

Figure 15: *Comparison with Sousa and Buchanan [1999b].*


(a) input image (b) S

(c) [Li and Huang 2003] (d) our final R

Figure 16: *Comparison with an LIC-based approach.*

lines, but also the illumination and shadow are well preserved by our tonal map.

We also compare in Fig. 16 our method with a LIC-based approach [Li and Huang 2003]. Our stroke map and the output are respectively shown in (b) and (d). We note that determining hatching direction based on local gradients is not always reliable, considering the ubiquity of highly textured regions in natural images.

The method presented in [Sousa and Buchanan 1999b] can generate results from drawing in other styles. We also apply our system to pencil drawing produced by the method of [Lee et al. 2006] to generate a result of a different style. We smooth the pencil drawing in (a) using bilateral filtering to get a grayscale image shown in (b), which is taken as the input of our system. Our final result is shown in (c). The close-ups show difference in strokes and hatching.

Fig. 18 shows a comparison with drawing produced by commercial software. (b) and (c) are results generated by PhototoSketch v3.5

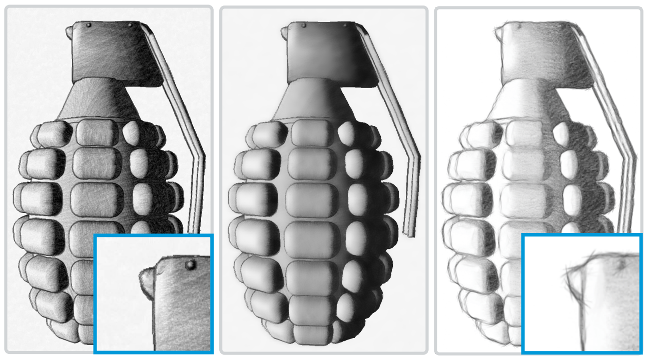

(a) [Lee et al. 2006] (b) BLF result of (a) (c) our result

Figure 17: *(a) is a result presented in [Lee et al. 2006]. (b) is the the bilateral filtering result of (a). (c) is our result taking (b) as input.*

[Photo to Sketch Inc. 2007] and the sketch filter of Adobe Photoshop CS5 [Adobe System Inc. 2011], respectively. Our result is shown in Fig. 18(d), which not only contains more detailed strokes but is visually appealing as well.

Finally, we show in Figs. 19 and 20 more results. Currently, the running time of our un-optimized Matlab implementation is about 2s for a 600×600 image on a PC with an Intel i3 CPU.

5 Concluding Remarks

Pencil drawing from natural images is an inherently difficult problem because not only structural information should be selectively preserved but as well the appearance should approximate that of real drawing using pencils. Thus a general algorithm that works for many types of natural images and can produce visually compelling results is hard to develop. Without 3D models, complex texture and spatially varying illumination further complicate the process. Directly generating strokes and hatching based on local gradients generally does not work well, due preliminary to variation of noise, texture and region boundaries. We address these issues and propose a two-stage system combining both line and tonal drawing. The main contribution includes a novel sketchy line generation scheme and a tonal pencil texture rendering step. They are effective and robust.

Limitation and Future Work As global pencil texture patterns are adopted in tonal image production, our method cannot be directly applied to video sequences. This is because the pencil patterns are nearly static among frames, inconsistent with possibly moving foreground and background layers. Our future work includes extending this framework to video sequences with temporal constraints and implementing the system on mobile devices.

6 Acknowledgments

The work described in the paper is supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region (Project No. 413110).

References

ADOBE SYSTEM INC., 2011. Photoshop CS5. `http://www.adobe.com/products/photoshop.html`.

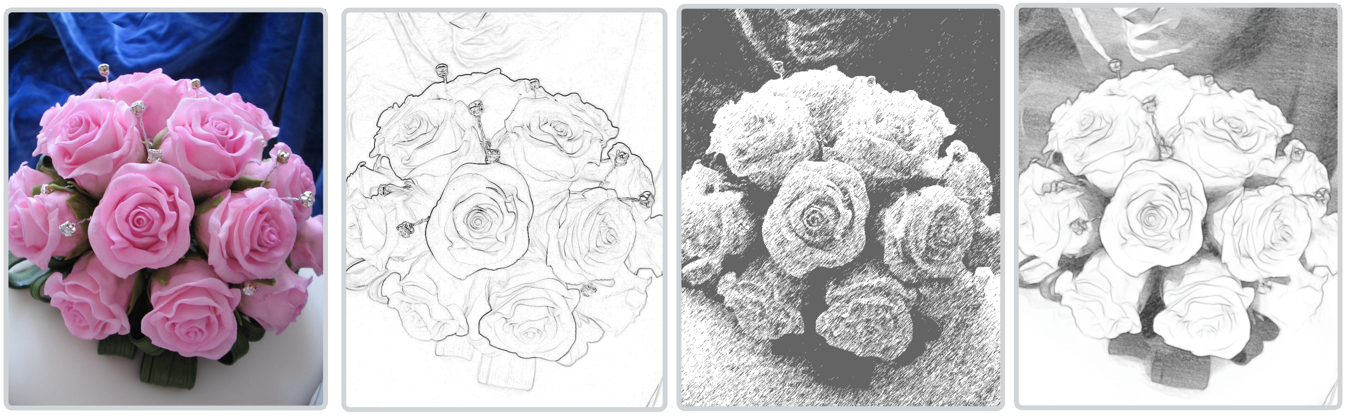

(a) input (b) PhototoSketch v3.5 (c) Photoshop CS5 (d) our result

Figure 18: *Comparison with commercial software.*

BOUSSEAU, A., KAPLAN, M., THOLLOT, J., AND SILLION, F. X. 2006. Interactive watercolor rendering with temporal coherence and abstraction. In *NPAR*, 141–149.

CHEN, H., LIU, Z., ROSE, C., XU, Y., SHUM, H.-Y., AND SALESIN, D. 2004. Example-based composite sketching of human portraits. In *NPAR*, 95–153.

CHEN, Z., ZHOU, J., GAO, X., LI, L., AND LIU, J. 2008. A novel method for pencil drawing generation in non-photo-realistic rendering. In *PCM*, 931–934.

CURTIS, C. J., ANDERSON, S. E., SEIMS, J. E., FLEISCHER, K. W., AND SALESIN, D. 1997. Computer-generated watercolor. In *SIGGRAPH*, 421–430.

CURTIS, C. J. 1998. Loose and sketchy animation. In *ACM SIGGRAPH '98 Electronic art and animation catalog*.

DECARLO, D., AND SANTELLA, A. 2002. Stylization and abstraction of photographs. *ACM Transactions on Graphics 21*, 3, 769–776.

DECARLO, D., FINKELSTEIN, A., RUSINKIEWICZ, S., AND SANTELLA, A. 2003. Suggestive contours for conveying shape. *ACM Transactions on Graphics 22*, 3, 848–855.

DURAND, F., OSTROMOUKHOV, V., MILLER, M., DURANLEAU, F., AND DORSEY, J. 2001. Decoupling strokes and high-level attributes for interactive traditional drawing. In *Rendering Techniques*, 71–82.

GAO, X., ZHOU, J., CHEN, Z., AND CHEN, Y. 2010. Automatic generation of pencil sketch for 2d images. In *ICASSP*, 1018–1021.

GRABLI, S., TURQUIN, E., DURAND, F., AND SILLION, F. X. 2010. Programmable rendering of line drawing from 3d scenes. *ACM Transactions on Graphics 29*, 2.

HERTZMANN, A., AND ZORIN, D. 2000. Illustrating smooth surfaces. In *SIGGRAPH*, 517–526.

HERTZMANN, A., JACOBS, C., OLIVER, N., CURLESS, B., AND SALESIN, D. 2001. Image analogies. In *Computer graphics and interactive techniques*, ACM, 327–340.

HERTZMANN, A. 1998. Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH*, 453–460.

HSU, S., AND LEE, I. 1994. Drawing and animation using skeletal strokes. In *Computer graphics and interactive techniques*, 109–118.

JUDD, T., DURAND, F., AND ADELSON, E. H. 2007. Apparent ridges for line drawing. *ACM Transactions on Graphics 26*, 3.

KANG, H., LEE, S., AND CHUI, C. K. 2007. Coherent line drawing. In *NPAR*, 43–50.

LAKE, A., MARSHALL, C., HARRIS, M., AND BLACKSTEIN, M. 2000. Stylized rendering techniques for scalable real-time 3d animation. In *NPAR*, 13–20.

LEE, H., KWON, S., AND LEE, S. 2006. Real-time pencil rendering. In *NPAR*, 37–45.

LEE, Y., MARKOSIAN, L., LEE, S., AND HUGHES, J. F. 2007. Line drawings via abstracted shading. *ACM Transactions on Graphics 26*, 3.

LI, N., AND HUANG, Z. 2003. A feature-based pencil drawing method. In *GRAPHITE*, 135–140.

MAO, X., NAGASAKA, Y., AND IMAMIYA, A. 2001. Automatic generation of pencil drawing from 2d images using line integral convolution. In *Proc. CAD/Graphics*, 240–248.

ORZAN, A., BOUSSEAU, A., BARLA, P., AND THOLLOT, J. 2007. Structure-preserving manipulation of photographs. In *NPAR*, 103–110.

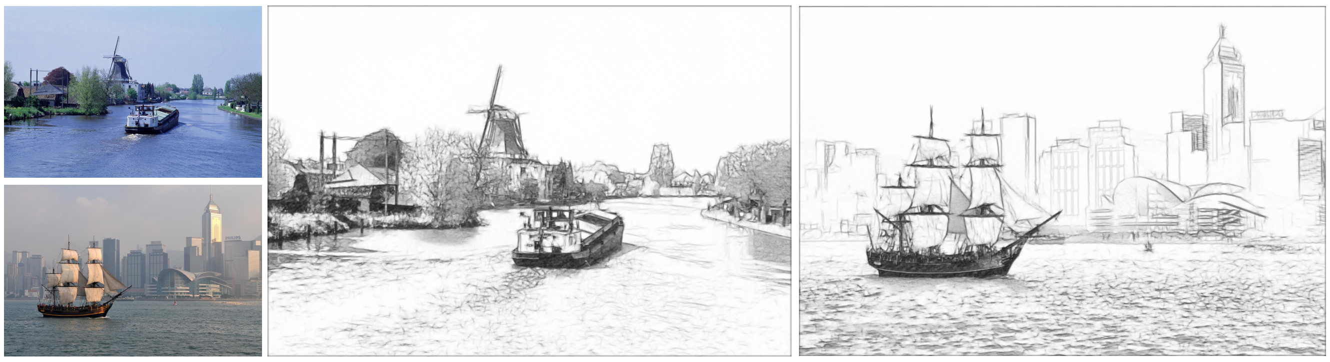
PHOTO TO SKETCH INC., 2007. PhototoSketch 3.5. `http://download.cnet.com/Photo-to-Sketch/`.

PRAUN, E., HOPPE, H., WEBB, M., AND FINKELSTEIN, A. 2001. Real-time hatching. In *SIGGRAPH*, 581.

RUSINKIEWICZ, S., DECARLO, D., AND FINKELSTEIN, A. 2005. Line drawings from 3d models. In *ACM SIGGRAPH 2005 Courses*.

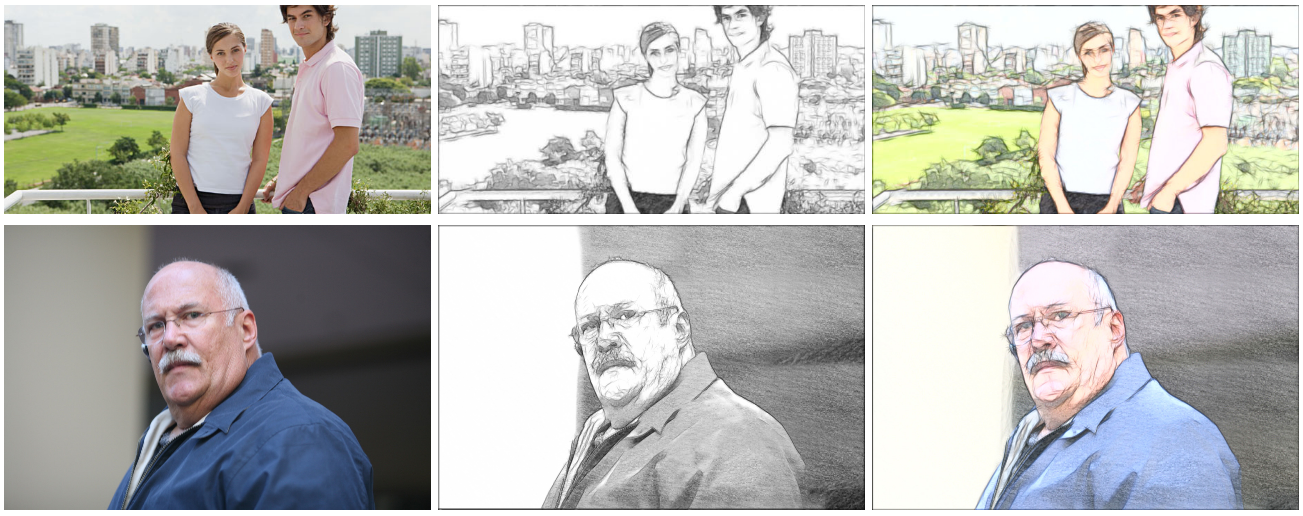
SALISBURY, M., WONG, M. T., HUGHES, J. F., AND SALESIN, D. 1997. Orientable textures for image-based pen-and-ink illustration. In *SIGGRAPH*, 401–406.

SON, M., KANG, H., LEE, Y., AND LEE, S. 2007. Abstract line drawings from 2d images. In *Pacific Conference on Computer Graphics and Applications*, 333–342.



(a) input (b) our result (c) our result

Figure 19: *More grayscale pencil drawing results.*



(c) input (b) our grayscale result (c) Our color pencil drawing result

Figure 20: *More grayscale and color pencil drawing results.*

SOUSA, M. C., AND BUCHANAN, J. W. 1999. Computer-generated graphite pencil rendering of 3d polygonal models. *Comput. Graph. Forum 18*, 3, 195–208.

SOUSA, M. C., AND BUCHANAN, J. W. 1999. Observational model of blenders and erasers in computer-generated pencil rendering. In *Graphics Interface*, 157–166.

STROTHOTTE, T., AND SCHLECHTWEG, S. 2002. *Non-photorealistic computer graphics: modeling, rendering, and animation*. Morgan Kaufmann Publishers Inc.

TAKAGI, S., NAKAJIMA, M., AND FUJISHIRO, I. 1999. Volumetric modeling of colored pencil drawing. In *Pacific Conference on Computer Graphics and Applications*, 250–258.

WANG, T. C. 2002. *Pencil sketching*. John Wiley & Sons, Inc.

WINKENBACH, G., AND SALESIN, D. 1994. Computer-generated pen-and-ink illustration. In *SIGGRAPH*, 91–100.

WINKENBACH, G., AND SALESIN, D. 1996. Rendering parametric surfaces in pen and ink. In *SIGGRAPH*, 469–476.

WINNEMÖLLER, H., OLSEN, S. C., AND GOOCH, B. 2006. Real-time video abstraction. *ACM Transactions on Graphics 25*, 3, 1221–1226.

XU, L., LU, C., XU, Y., AND JIA, J. 2011. Image smoothing via l0 gradient minimization. *ACM Transactions on Graphics 30*, 6.

YAMAMOTO, S., MAO, X., AND IMAMIYA, A. 2004. Enhanced lic pencil filter. In *CGIV*, 251–256.

ZENG, K., ZHAO, M., XIONG, C., AND ZHU, S. C. 2009. From image parsing to painterly rendering. *ACM Transactions on Graphics 29*, 1.

ZHAO, J., LI, X., AND CHONG, F. 2008. Abstract line drawings from 2d images based on thinning. In *Image and Signal Processing*, 466–470.