Colored Pencil Filter with Custom Colors

Shigefumi Yamamoto, Xiaoyang Mao, Atsumi Imamiya
University of Yamanashi, Japan
mao@yamanashi.ac.jp

Abstract

This paper presents a new technique for automatically converting digital images into colored pencil drawings. In real colored pencil drawing, artists use different colors for different regions, and add pure colors directly onto paper to build the target color through optical blending. To support such effect, our technique extends the existing technique for reproducing color image with custom inks to automatically select the best color set for individual regions in a source image. Then layers of stroke image for each color are generated and superimposed with the Kubelka-Munk optical compositing model. We also allow users to specify regions and to customize the color set for a specified region interactively. The proposed technique can be easily extended to simulate other artistic media featured with optical color blending, such as pastel and wax crayons.

1. Introduction

Colored pencil drawing is now gaining in appeal as an art form in its own right[1]. Its soft, harsh, and misty appearance gives us a gentle and charm feeling, while the line stroke based style provides us with an expressive and abstract representation of its subject. We propose a new colored pencil filter for automatically converting a 2D source image into a colored pencil drawing. Our major contribution is the support for color customization. In real colored pencil drawing, an artist usually presents a target color by mixing several colors. As usual, different color sets are used for different regions depending on the target color of the region. Such kind of color customization is a technique not just for producing more color variation from a given colored pencil set, but also for enriching the appearance of resulting works and for demonstrating the artist's personal style. Our new colored pencil generation technique supports both automatic and interactive color customization. The automatic color customization first segments the source image into different regions and automatically selects the best color set for each region, while the interactive customization allows the user to interactively specify regions and choose own color sets for the specified regions. After each region is assigned with a color set, the density of each color required for building a target color is automatically calculated. This density is then used to control the stroke density in generating the stroke image of each color. We employed a technique for reproducing color image with custom inks[2] for realizing the color customization.

Unlike those brush work based media such as oil painting or watercolor painting, where colors are usually mixed in palette before applied to surface, colored pencil drawing requires artists to directly apply pure colors to drawing paper, and to build the target color through optical blending. There are mainly two different ways to achieve the optical color blending effect in case of colored pencil drawing[1]. One known as glazing, is to overlay colors on previous layers so that light reflected from below the surface comes through presenting the blended color of multiple layers. The other is to juxtapose color strokes side by side close enough so that human eyes can "mix" them. The existence of different color strokes actually gives the subtle shade of color together with the textured appearance, which contributes to the unique rich and expressive representation of colored pencil drawings. We use the Kubelka Munk(KM) model[26] for modeling the optical color blending of two color layers and our composition model works in the same way both for glazing and juxtaposition. Our technique can be easily extended to other solid pigment-based media, such as pastel, wax rayon, and chalk, all share the common feature of color customization and optical color blending.

Instead of providing an accurate physical simulation to

the colored pencil drawing, we aim to develop a real time technique processing a 2D source image for an as realistic as possible colored pencil drawing effect. For this purpose, we employ the existing automatic pencil drawing generation technique based on Line Integral Convolution(LIC)[3] for creating the stroke image of each color layer.

In the remainder of this paper, we will first review the related work in Section 2. Section 3 briefly introduces the LIC based pencil drawing generation technique. Section 4 describes the algorithm of the proposed technique. Color customization and optical color blending are discussed in Section 5 and 6, respectively. Section 7 shows the implemented results and Section 8 concludes the paper with a discussion on the limitation of our approach and possible further extensions of the work.

2. Previous work

In the past decade, a number of techniques have been developed to simulate traditional artistic media and styles, such as pen and ink illustration[4-5], graphite and colored pencil drawing[6-8], woodcuts and engraving[9-10], impressionist styles[11], paintings of various materials including oil[12], water color[13], wax crayon[14] and so on. These techniques can be classified roughly into two different approaches as explained below. The first approach is to provide physical simulation to the materials and skills, and has been mainly combined with interactive painting systems or 3D non-photorealistic rendering systems for generating realistic painterly images. The second approach is the painterly filtering, which involves taking an image and applying some kind of image processing or filtering techniques to convert it into an image with a painterly look. Our technique falls into the second category, in particular, it converts a 2D source image into a colored pencil drawing with no or minimum user intervention. More general surveys on computer simulation to traditional media can be found in the literatures[15-16], we will briefly summarize the researches related to our work in the following four aspects: painterly filtering, colored pencil drawing, color customization and optical blending.

Painterly filtering

A pioneer work on painterly filtering is Haebeli's interactive system for creating an impressionistic image from a source photograph[17]. His technique allows a user to interactively control the selection of sample position for placing brush strokes as well as the attributes of brushstrokes. Also for achieving impressionist effect, Litwinowicz developed an automatic technique which can be applied to both still images and video clips[11]. Hertzmann succeeded in achieving more realistic hand-painted appearance by using multiple layers of brushstrokes and decreasing the sizes of brushstrokes for the later added layers[12]. This technique has been further extended for processing video clips[18]. There are also several works on adapting the attribute of brushstrokes to the local statistics and features of the source image[19-20]. Hertzmann et al also came to a very smart idea of using recent texture synthesis technology for learning various painterly filtering effects from a pair of sample source-target images[21]. While many excellent works have been done on generating brushstroke based paintings, relatively few publications can be found on converting a source image into line stroke based drawings. In case of drawing, geometric information such as the outline of regions and the direction of strokes becomes much more sensitive to the visual quality of the result, and it is generally difficult to extract such information from 2D raster images automatically. As one of those successive works on pen-and-ink illustrations, Salisbury et al proposed a technique for generating pen-and-ink illustrations from an input reference image[5]. Their technique requires the user to specify stroke type and orientation for each region. Sousa and Buchanan demonstrated that their observation model for graphite pencil drawing can be used for converting a source image into a pencil drawing, if the parameter values of strokes are given[7]. Instead of modeling line strokes geometrically, we succeeded in generating line stroke like images with pixel-by-pixel image filtering technique based on LIC in our previous study[3]. The technique utilizes image segmentation and texture analysis technique to automatically detect outlines and decide stroke orientation. The work presented in this paper can be considered as an extension to the LIC pencil filter from the view point that we use it for generating the stroke image of individual color layer.

Colored pencil drawing

The only published work on colored pencil drawing is the volumetric modeling technique by Takagi et al [6]. Their technique uses 3D volume to model the microstructure of drawing paper as well as the deposition of pigment onto the paper and render the 3D volume with volume ray-tracing. This technique can generate reasonably realistic images, but it is computationally too expensive, to be used in any interactive painting system. Several commercial packages provide automatic colored pencil filters, but most of them use a similar approach as for generating brushstroke based paintings, such as by adding narrow long shaped brush strokes to a low-pass filtered version of the source image, and hence can not

well capture the feature of colored pencil drawing as a line stroke based medium. This drawback can be observed from Figure 6(c) which is generated using *colored pencil filter* in Adobe PhotoShop™ software[28]. Although some systems allow the user to combine the results from several filters, he/she usually needs to experiment with each image through trial and error for many times before obtaining a satisfactory image. Furthermore, none of those systems support the automatic color customization and optical color blending effect.

Color customization

A field closely related to the color customization problem in our study is the work on reproducing color images with customized ink sets. Usually the range of colors (called gamut) which can be reproduced by a fixed set of inks is limited when compared to the full range of colors visible to the human eye. Studies have been carried out trying to allow the selection of the best ink set for a particular image to be reproduced. Two representative approaches were developed by Power et al[2] and by Stollnitz et al[24]. Power et al presented a technique for *duotone* printing, in which only two inks are used, and demonstrated that an optimal choice of two inks for a particular image can result in remarkably good reproductions[2]. Stollnitz et al further generalized Power et al's work to allow the using of three or more custom inks[24]. Our current implementation only supports the color blending of two layers and we extend Power et al's technique for the color customization in colored pencil drawing. The idea of using different color sets for different regions in painterly image generation has also been explored by Curtis et al in applying their water color modeling technique for automatic water colorization of source images[13]. In their system, the color set for each region is specified by the user and a brute force algorithm is used to calculate the density of each color from all possible combinations.

Optical color blending

Optical blending of color pigments has been addressed in many works in the study of non-photo-realistic rendering. Haase and Meyer first introduced Kubelka-Munk(KM) model to the computer graphics field for modeling the appearance of a paint consisting of several pigments[26]. KM model has been used by Curtis et al for simulating the optical effect of the superimposed glazes in water color paintings[13]. More recently, Rudolf et al simulated the optical blending in wax crayons with a simplified KM model[14]. Our technique uses the KM model for modeling the effect of overlaying two layers of stroke images. We use an approach similar to Curtis et al.'s one to approximate the reflection and transmittance coefficients of each color required by the KM model.

3. Pencil drawing filter using LIC

Line Integral Convolution (LIC) is a texture based vector field visualization technique[22]. The idea of using LIC for pencil drawing generation was inspired by the visual similarity of LIC images and pencil drawings. An LIC image is obtained by low-pass filtering a white noise along the local streamlines of a vector field. Therefore we can observe the traces of streamlines along which intensity varies randomly on an LIC image. Such traces have a similar appearance of pencil strokes where the variance of intensity is caused by the interaction of lead material and the rough microtexture of paper surface. As shown in Figure 1, LIC pencil filter generates a pencil drawing from a 2D source image in the following seven steps:

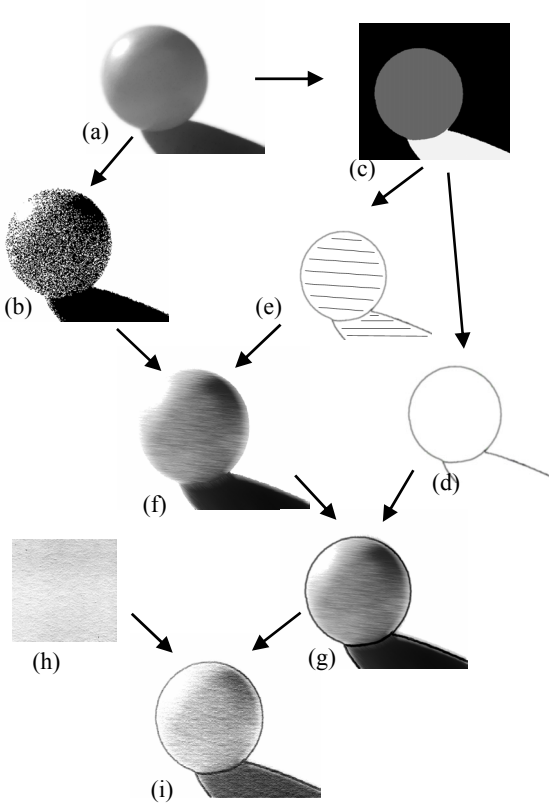

Figure 1 Pencil drawing generation using LIC[3]

1. Generate a white noise(Figure 1(b)) from the source image(Figure 1(a)).
2. Segment the input image(Figure 1(a)) into different regions(Figure 1(c)).
3. Extract region boundary(Figure 1(d)).
4. Generate the vector field (Figure 1(e)) representing the orientation of strokes.
5. Generate stroke image (Figure 1(f)) by applying LIC to the white noise(Figure 1(b)) and the vector field(Figure 1(e)).
6. Add the boundary (Figure 1(d)) to obtain the drawing with outlines(Figure 1(g)).
7. Composite the resulting image (Figure 1(g)) with the paper sample (Figure 1(h)) to obtain the finished pencil drawing(Figure 1(i)).

To match the tone between the source image and the resulting pencil drawing, the white noise image is generated in a way that the probability of a white value being set for a pixel is proportional to the intensity level of the corresponding pixel in the source image. A Fourier analysis technique is employed to extract the directions of textures. If a region does not consist of directional textures, then a randomly chosen stroke direction is assigned to it. We can easily change the appearance of the pencil drawing through adjusting some parameters of LIC. For example, varying the length of the line integral convolution kernel visually changes the length of strokes and adjusting the granularity of white noise changes the width of strokes.

4. Algorithm overview

As shown in Figure 2, the proposed technique generates a colored pencil drawing from a source image in the following seven steps:

1. Segment the source image(Figure 2(a)) into different regions and decide the two best colors for each region(Figure 2(b)).
2. For each of the two colors chosen in Step 1, calculate its density required for building the target color. The density is calculated in a pixel-by-pixel manner, the result consists of two density values(one for each color) for each pixel. Subsequently, we generate two layers of gray scale images with the density of each color (Figure 2(c,d)).
3. Generate noise images(Figure 2(e,f)) for the two layers(Figure 2(c,d)), respectively. Each region of the resulting noise images is a white noise with its pixels being either white or the color of the region with a probability proportional to the density value.
4. For each layer, define a vector field representing the stroke directions.
5. Apply LIC to the vector fields and noise images to generate two layers of stroke images (Figure 2(g,h)).
6. Modify the stroke images with the given paper model to obtain two improved stroke images(Figure 2(i,j)).
7. Blend the two layers with the KM model to finally produce the finished colored pencil drawing (Figure 2(o)).

Due to our assumption that an artist chooses color set for a region according to the overall color of the region, a color-based split and merge segmentation algorithm[27] is used in Step 1. The technique first transforms a given image into CIE L*a*b color space, which is known to be more compatible with colors sensed by human eyes. Then pixels with small L*2* norm distance to each other in L*a*b space are merged to form regions. The user is also allowed to specify regions interactively. In this case, the best color set for the specified region can either be specified by the user or automatically decided by the system. But note that in either case, the required density of each color for building the target color of the region is calculated automatically. The details of color customization can be found in the Section 5 and the optical color blending at Step 7 will be described in Section 6.

At step 2, we assign the color of higher density to the bottom layer and the color of lower density to the top layer. We use this to simulate the fact that an artist usually draw the dominant color first and then add other colors on top of it decoratively. Therefore, for each region we calculate the total density of each color. The color having higher total density goes to the bottom layer(Figure 2(c) and the other goes to the top layer (Figure 2(d)). Step 4 uses different methods for deciding the stroke direction of the bottom and the top layers. For the bottom layer, each pixel is subjected to be detected if it is a part of a directional texture. If so, the detected texture direction is set to the pixel, otherwise a direction which is randomly assigned to the region enclosing the pixel is used. The detailed description on texture direction detection can be found in the literature[3]. The strokes on the top layer are aligned to have a user specified angle with the strokes on the bottom layer. For example, if a user specifies the angle

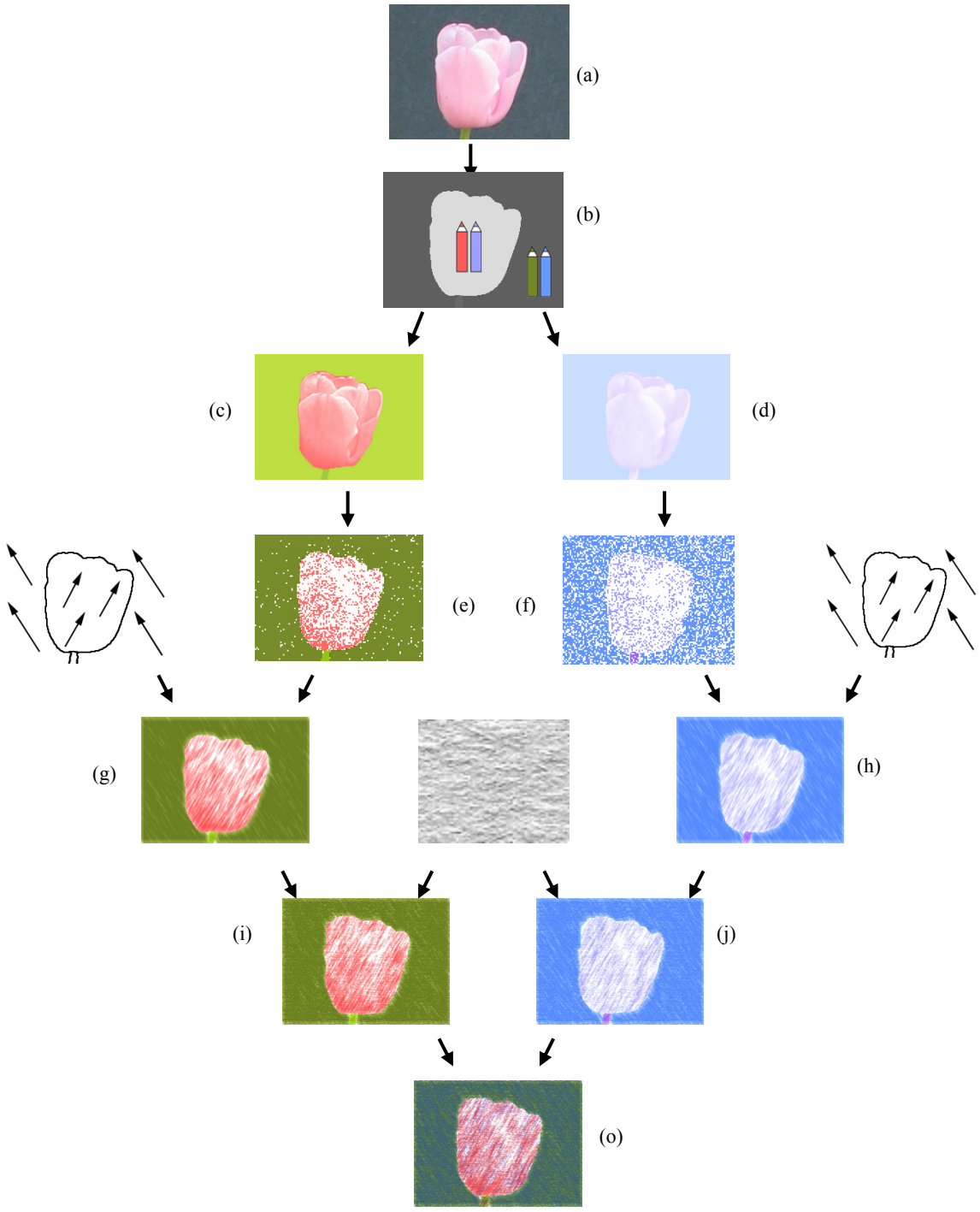

Figure 2 Colored pencil drawing generation process

to be 0°, then the strokes on the top layer are in the same directions as the strokes on the bottom layer. If the angle is 90°, then an orthogonal cross-hatching effect can be obtained. The user is allowed to adjust the angle interactively to achieve different cross-hatching effect.

The drawing paper is modeled as a height field calculated from the intensity of the scanned image of a paper. Low and high intensities on the scanned image correspond to a valley and peak on the height field of the paper structure, respectively. At Step 6, we use the intensity of the scanned paper image to adjust the intensity of the stroke images obtained at the previous step. We first calculate the average intensity of the paper image. As it is observed that lead materials are more likely being deposited near the peak area[6], for those pixels with its intensity above the average we increase the intensity in stroke image and for those pixels with its intensity lower than the average we reduce its intensity in stroke images with an amount proportional to their deviations from the average intensity. This algorithm is an improved version of the method used in our previously developed LIC based pencil filter, the latter simply subtracts the paper image from the stroke image[3]. The new method preserves the overall intensity of the stroke images better and can achieve more realistic effect of paper texture.

5. Color customization

We employ Power et al's technique for reproducing color image from duotone color to select the best color set for each region and to calculate the required density of each color[2]. Power et al's work assumes that color halftone printing is used, in which an image is reproduced by printing several layers on top of one another, with each layer consisting of dots of a single ink. Color halftone printing differs from color dithering on monitors in that both subtractive effects and additive effects contribute to the reproduced color. The subtractive effect of superimposing dots of different color produces the set of printing primaries. For example, for cyan, magenta, and yellow inks printed on white paper, the set of printing primaries is cyan, magenta, yellow, blue (cyan + magenta), green (cyan + yellow), red (magenta +yellow), black (cyan + magenta + yellow), and white (no ink). The additive effect of juxtaposing dots of primary colors produces the entire gamut achieved with the particular set of inks. One interesting observation is that pencil drawing actually also has this property, that is, both subtractive effect in superimposition as well as additive effect in juxtaposition plays a role in building colors except for that strokes of much rougher resolutions are used instead of the high resolution ink dots. And such observation actually gives us the motivation for using the color reproduction technique for realizing the color customization in colored pencil drawing.

In halftone printing, the amount of ink needed to represent a given color can be represented as an equation given by Neugebauer[25]. The Neugebauer equation for duotone printing is given as

$$c = \begin{bmatrix} g_0 & g_1 & g_2 & g_{12} \end{bmatrix} \begin{bmatrix} (1-\alpha_1)(1-\alpha_2) \\ \alpha_1(1-\alpha_2) \\ (1-\alpha_1)\alpha_2 \\ \alpha_1\alpha_2 \end{bmatrix}$$

Here $\boldsymbol{g}_0$ is the color of the paper, $\boldsymbol{g}_1$, $\boldsymbol{g}_2$ the color of ink_1 and ink_2 on the paper, $\boldsymbol{g}_{1,2}$ the color of ink_1 and ink_2 superimposed on the paper, α_i is the amount of ink_i (between 0 and 1) which corresponds to the relative area the ink covers. The above equation describes a color $\boldsymbol{c}$ in the printing gamut in terms of the four printing primaries and the amounts of two inks(α_1, α_2) required to achieve $\boldsymbol{c}$. If we represent colors in a 3D additive color space such as the CIE *XYZ* color space, then the duotone gamut given by above equation is a bilinear surface shown in Figure 3.

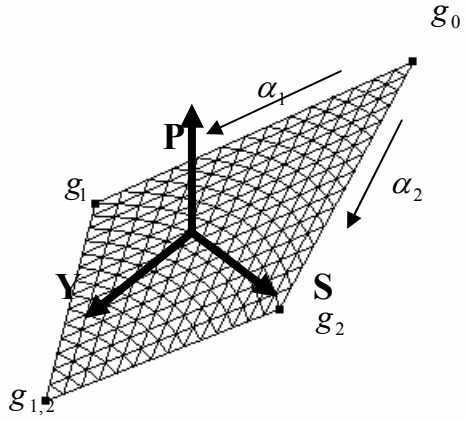

Figure 3 Duotone gamut as a bilinear surface in CIE XYZ color space

To decide the gamut, the four primary $\boldsymbol{g}_0$, $\boldsymbol{g}_1$, $\boldsymbol{g}_2$, $\boldsymbol{g}_{1,2}$, should be decided. This requires estimating the color of inks printed on the selected paper and the color of two inks superimposed on the selected paper. In our current implementation, we assume that the color of paper is pure

white and superimposing a color on the white paper results in the color itself. The result of superimposing two colors on the white paper is calculated with the KM model using the method described in Section 6.

After the gamut surface of the two given colors are obtained, the problem of calculating the density of each color for a pixel is equivalent to find a transformation to map the pixel color in the 3D XYZ color space on to the gamut surface. The mapping is realized in 3 steps:

1. Choose the fist axis of the duotone gamut to be the same as the Y axis of the XYZ color space so as to preserve the relative luminance. For each pixel, linearly transform its y value into the new range of Y formed by four primaries $\boldsymbol{g}_0$, $\boldsymbol{g}_1$, $\boldsymbol{g}_2$, $\boldsymbol{g}_{1,2}$.
2. Orthogonalize $\boldsymbol{g}_2$ - $\boldsymbol{g}_1$ with respect to Y and choose it as the second axis of the duotone gamut so as to preserve the separation in the direction of widest color variation on the duotone gamut. For each pixel, linearly transform its color separation into the separation range on gamut surface.
3. For each point resulted from Steps 1 and 2, project it onto the gamut surface along the direction of $\boldsymbol{Y}$x$\boldsymbol{S}$ to obtain color amount α_1, α_2.

See the literature[2] for more details on the color mapping procedure.

To find the best colors for a region, we generate the colored pencil drawings for the region with all possible pair of colors from the giving color set, while comparing the distance between the regions in the low-pass filtered colored pencil drawing and in the source image. The pair produces a colored pencil drawing region with its low-pass filtered version closest to the region in the source image is selected as the best color set for the region. The distance between two regions is measured as the sum of pixel-wise *L*2 norm in XYZ space. Same as in Power et al's technique, we use simulated annealing method for accelerating the optimization process[2].

Figure 4 shows several images resulting from using different color sets for the same source image(Figure 4(a)). Color sets are shown with the icons under each image. The left icons are the colors for the bottom layers and the right icons are for the top layers. Figure 4(b) is generated with the automatically selected color set and Figure 4(c) and (d) are generated with user specified color sets. Although automatically selected color set produces the image closest to the source image, Figure 4(c) and (d) also demonstrate that customized color sets can be used for producing some special color effects.

6. Optical color blending

We use the Kubelka-Munk (KM) model for performing the optical blending of the overlaid two stroke layers. Given a color layer of thickness *d*, the reflectance *R* and transmittance *T* through the layer can be calculated with the following equations:

$$R = \frac{\sinh bSd}{a \sinh bSd + b \cosh bSd}$$

$$T = \frac{b}{a \sinh bSd + b \cosh bSd}$$

Here

$$a = \frac{(S+K)}{S} \qquad b = \sqrt{a^2 - 1}$$

The total reflectance and transmittance from two layers can be calculated as

$$R = R_1 + \frac{T_1^2 R_2}{1 - R_1 R_2} \qquad T = \frac{T_1 T_2}{1 - R_1 R_2}$$

To use the KM model, we need to know the absorption coefficient *K* and the scattering coefficient *S* for each color, which represent the fraction of energy absorbed and scattered back, respectively, per unit distance in the color layer. Curtis et al. proposed a perception based method to approximate these coefficients[13]. They allow a user to specify the coefficients *K* and *S* interactively by choosing the desired appearance of a "unit thickness" of the pigment over both white and black backgrounds. Given these two user-selected *RGB* colors, respectively, they compute *K* and *S* values for each RGB components with the inversion of the KM equation. We adopted a similar approach in our system, but as the default we assume that a color does not change its appearance with a white background and place it on a black background will reduce its intensity. The rate of intensity reduction is

decided by a user given parameter which provides the control to the transparency of the color. When in the interactive mode, the user is allowed to change these two colors interactively. When running the automatic color customization, for each color, all possible appearances obtained by varying this transparent parameter are tested.

In our case, each layer is a gray-scale stroke image of the corresponding color with the intensity value of each pixel standing for the color density of a stroke passing through that pixel. We perform KM composition of two stroke images pixel-by-pixel by treating the intensity of each pixel as the thickness of the color at the pixel. In this way, we can make our composition model work in the same way both for glazing and juxtaposition effects.

7. Results

We have implemented our technique as a Windows application using Java. The prototype system provides two processing modes: automatic and interactive. With the automatic mode, a colored pencil drawing is generated in a completely automatic way after a user specifies an input image. A user is allowed to change the result by adjusting the threshold for segmentation, and the parameter for controlling the transparency of colors. In the interactive mode, the user is allowed to specify individual regions. We provided two different interfaces for specifying regions, one is similar to the *Magic Wand* tool of Adobe PhotoShop™, allowing users to use mouse to specify a seed and a threshold of color difference. Then a region is obtained by expanding from the seed to include all pixels with their color difference from the seed not exceeding the threshold. Another interface allows a user to specify the boundary of a region, and thus makes it possible to intentionally concatenate regions of different colors. A user does not need to specify all regions but can just selectively specify those regions he/she wants to use customized colors. After each region is specified, the best color set can either be automatically decided or again specified by the user. Then the colored pencil drawing is generated and added to the previously generated part. Figure 5 shows a pencil drawing (right) automatically generated from a picture of still objects (left). The input picture is segmented into the regions of different fruits and a best color set is automatically selected for each regions. We can see the colors in the input picture were well reproduced in the resulting pencil drawing. Depending on artists' style, the boundaries of regions are usually not explicitly drawn in colored pencil drawing but left to be perceived naturally through the changing of color at the boundary. Some artists prefer not to mix strokes of different colors at boundary so as to present sharp boundaries, while others like to have fuzzy and muddy boundaries bring softer and charmer mood to the work. Our system allows users to choose either of these two styles, the sharp boundaries are obtained by setting the vector magnitude to be 0 at the region boundaries so that convolution kernel of LIC never across the boundaries. Figure 6 shows an image with soft boundaries. In case of this example, region segmentation and color selection are specified by a user interactively.

The time required for automatically generating an image of 640*480 on a Pentium(R)4 PC is about 10 minutes. The cost is mainly due to the optimization process for finding the best color set and depends on the number of regions rather than the size of the image. Other parts of the algorithm can be executed in real time.

8. Concluding Remarks

We have presented a new colored pencil drawing filter featuring the color customization function. In fact, other solid pigment-based media, such as pastel, wax rayon, and chalk, all share the common feature of optical color blending. Therefore, the proposed technique can also be easily extended for these media. One of the major future works is to provide a more accurate approach for color customization. Our current implementation is based on the assumption that overlaying color stroke layers is similar to overlaying ink dot layers in halftone printing. But this is not true. The distribution of colored pencil strokes is usually non-uniform and anisotropic, unlike the distribution of ink dots in halftone printing. Furthermore, the resulting image color can look very different if we overlay the strokes on two layers along different directions. For example, orthogonal cross-hatching must have less overlapped area compared with overlaying the strokes on two layers in parallel direction. A more accurate approach requires taking into account these properties of colored pencil drawing. Although our current implementation can only deal with two color layers, extension to arbitrary number of layers can be realized by using Stollnitz's technique[24]. Developing a more perceptually effective image segmentation technique and a new stroke direction specification technique conveying object shapes more effectively is also one of our important future work..

Acknowledgement

The authors would like to thank Ryutaro Ohbuchi from University of Yamanashi for his helpful comments. We are deeply grateful to Issei Fujishiro of Ochanomizu University for his valuable advices. This project is supported by the National Science Foundation China (NSFC) Grant no. 60340440422.

References

1. I. Hutton-Jamieson. *Colored Pencil Drawing Techniques*.North Light Books, September 1986.
2. J. L. Power, B. S.West, E. J. Stollnitz, and D. H. Salesin. Reproducing Color Images as Duotones. In *Proceedings of SIGGRAPH 96*, pages 237–248, 1996.
3. X. Mao, Y. Nagasaka and A. Imamiya", Automatic Generation of Pencil Drawing from 2D Images Using Line Integral Convolution", *Proceedings of the Senventh International Conference on Computer Aided Design and Computer Graphics CAD/GRAPHICS2001,* PP. 240-248, 2001.
4. G. Winkerbach, D. H. Salesin, "Computer-Generated Pen-and-Ink Illustration", *SIGGRAPH94 Conference Proceedings*, pp. 91-100, 1994.
5. M. P. Salisbury, M. T. Wong, J. F. Hughes, and D. H. Salesin, "Orientable Textures for Image-Based Pen-and-Ink Illustration", *SIGGRAPH 97 Conference Proceedin*gs, pages 401–406, 1997.
6. S. Takagi, I. Fujishiro and M. Nakajima, "Volumetric modeling of colored pencil drawing," *Pacific Graphics '99 conference proceedings,* page 250-258, 1999.
7. M.C.Sousa and J.W.Buchanan, "Observational Model of Blenders and Erasers in Computer-Generated Pencil Rendering", *Graphics Interface '99 conference proceedings*, pages 157-166, 1999.
8. M.C.Sousa and J.W.Buchanan, "Computer-Generated Graphite Pencil Rendering of 3D Polygonal Models", *EUROGRAPHICS '99 conference proceedings*, pages 195-207, 1999.
9. S. Mizuno, M. Okada and J. Toriwaki, "An Interactive Designing System with Virtual Sculping and Virtual Woodcut Printing", *EUROGRAPHICS '99 Conference Proceedings*, pp. 183-193, 1999.
10. V. Ostromoukhov, "Digital Facial Engraving", *SIGGRAPH '99 Conference Proceedings,* pp. 417-424, 1999.
11. P. Litwinowicz, "Proceeding Images and Video for An Impressionist Effect", *SIGGRAPH97 Conference Proceedings*, pp.407-414, 1997.
12. A. Hertzmann, "Painterly Rendering with Curved Brush Strokes of Multiple Sizes", *SIGGRAPH98 Conference Proceedin*gs, pp. 453–460, 1998.
13. C. J. Curtis, S. E. Anderson, J. E. Seims, Kurt W. Fleische, and D. H. Salesin, "Computer-Generated Watercolor", *SIGGRAPH 97 Conference Proceedin*gs, pp. 421–430, 1997.
14. D. Rudolf, D. Mould, E. Neufeld, "Simulating Wax Crayons", *Pacific Graphics '03 conference proceedings,* page 163-174, 2003.
15. Gooch & Gooch, *Non-Photorealistic Rendering,* AK Peters Ltd, 2001,
16. T. Strothotte and S. Schilchtweg, *Non-Photorealistic Computer Graphics. Modelling, Rendering, and Animation,* Morgan Kaufmann, San Francisco, 2002.
17. P. E. Haeberli. Paint By Numbers: Abstract Image Representations. Computer Graphics (SIGGRAPH '90 Proceedings), volume 24, pages 207-214, 1990.
18. A. Hertzmann and K. Perlin. Painterly Rendering for Video and Interaction. In Proceedings of the First Annual Symposium on Non-Photorealistic Animation and Rendering, June 2000.
19. M. Shiraishi and Y. Yamaguchi. An algorithm for automatic painterly rendering based on local source image approximation. NPAR 2000: First Interna-tional Symposium on Non Photorealistic Animation and Rendering, pages 53-58, June 2000.
20. S. M. F. Treavett, M. Chen Statistical Techniques for the Automatic Generation of Non-Photorealistic Images, Proceedings of 15th Eurographics UK Conference, March 1997.
21. A Hertzmann, C. E. Jacobs, N. Oliver, B. Curless. and H. D. Salesin, Image Analogies. *Proc. of SIGGRAPH '01,* pp. 327-340.
22. B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proc. ACM SIGGRAPH 93(COMPUTER GRAPHICS Annual Conference Series)*, pages 263–272, August 1993.
23. Joanna L. Power, Brad S. West, Eric J. Stollnitz, and David H. Salesin. Reproducing Color Images as Duotones. In *Proceedings of SIGGRAPH96*, pages 237-248, 1996.
24. Eric J. Stollnitz, Victor Ostromoukhov, and David H. Salesin. Reproducing Color Images Using Custom Inks. In *roceedings of SIGGRAPH 98*, pages 267-274. ACM, New York, 1998.
25. Kazuo Sayanagi, editor. *Neugebauer Memorial Seminar on Color Reproduction*, volume 1184 of *Proceedings of the SPIE*. SPIE, Bellingham, WA, 1990.
26. C. S. Haase and G. W. Meyer. Modeling pigmented materials for realistic image synthesis. *ACM Transactions on Graphics (TOG)*, 11(4):305 335, 1992.
27. Y. Hamasaki, K. Kondo, Image Generation Method using Synthesis and Control of Rendering Region, *Proceedings of the 1st annual conference of Asia Digital Art and Design Association*, pp.70-71, 2003.
28. Adobe Systems Incorporated, *AdobePhotoshop 6 User's Manual*, 2001.

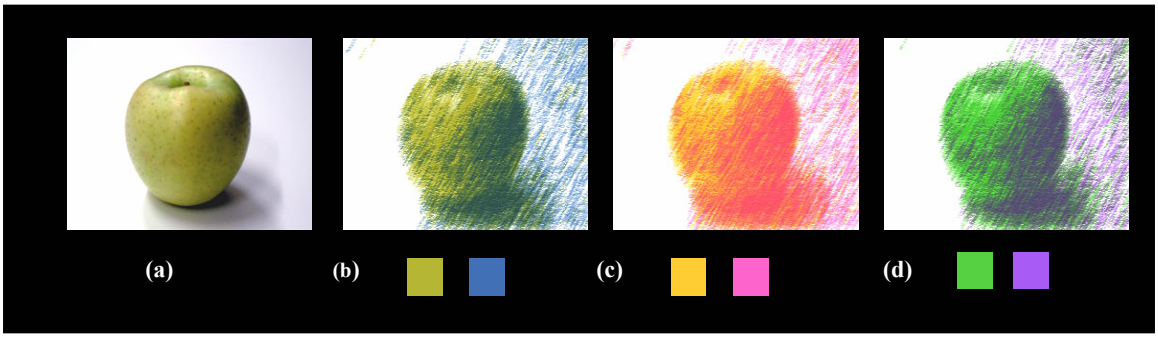

Figure 4 Colored pencil drawing generated with different color sets. (a) source image; (b) automatically selected color set; (c),(d) user specified color sets.

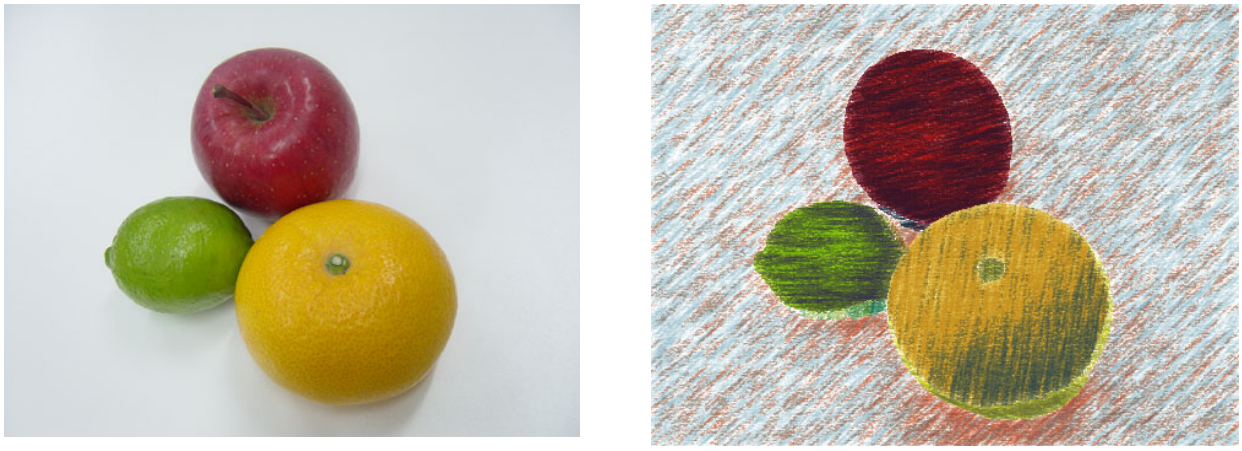

Figure 5 An automatically generated colored pencil drawing of still objects.

Figure 6 An colored pencil drawing with the soft boundary effect.