Don’t Photoshop it!
Artistic Image Filtering

By
A'aeshah Alhakamy

Department of Computer and Information Science
Indiana University – Purdue University Indianapolis (IUPUI)

Image Processing and Computer Vision
Dr. Mihran Tuceryan

1 May 2016

Introduction
Non-photorealistic rendering is a method of image manipulation that modifies images for artistic purposes. The photographs can be stylized by simplifying or emphasizing the perceptual information. Most of the operations are done using image filtering by means of convolution between an image and a kernel. Other possible operations are: image adjustment, image arithmetic, color conversion, and color quantization. These operations are used to implement three artistic effects in this project: cartoon effect, sketch effect, and gilding effect. In addition, built-in Matlab functions are used to create three more effects: dithering operation to pointillize the image, dilation operation with a circular structuring element, and inversion operation.

Related work
There are many papers and web pages dedicated to non-photorealistic rendering and artistic image filtering. Coraor (2014) presents an example of complete implementation of cartoon-like algorithm. The color quantization is done directly in RGB space which results in cartoon images having false colors. Performing the color
quantization only in the luminance channel produces better results with preserved colors (Winnemöller et al., 2006). Pencil sketching is usually done using edge extraction and similar approaches (Xu, 2011). The 3D-like appearance of the gilding effect is done using the embossing operator, utilized by Kim et al. (2010) and Vandevenne (2007).

**Approach: Cartoon effect**
In order to achieve the required results, the following steps are performed to produce the cartoon effect:

- the original image is converted from RGB color space to Lab color space;
- the lightness channel is extracted from the Lab image;
- image blurring is done using a Gaussian filter;
- quantization of the lightness channel is done using a staircase function.

**Approach: Sketch effect**
In order to achieve the required results, the following steps are performed to produce the sketch drawing effect:

- the original image is converted from RGB color space to Lab color space;
- the lightness channel is extracted from the Lab image;
- edge detection is performed using the difference of Gaussians;
- the result is adjusted for the edges to be black and other areas to be white.

**Approach: Gilding effect**
In order to achieve the required results, the following steps are performed to produce the gilding effect:

- the original image is converted from RGB to grayscale;
• blurring is performed using the Gaussian filter;
• highlights and shadows are created using the embossing filter;
• the result is modulated by yellow color to achieve a golden plate look.

Implementation: Cartoon effect

The Matlab code for the cartoon effect is stored in cartoon.m. The original image is supplied as a function argument (Figure 1, left). The original image is converted from RGB color space to LAB color space (Wyszecki and Styles, 1982). First, a RGB to Lab color transformation is created using the makecform function. Then, the color transformation is performed using the applycform function. Finally, the lightness channel is extracted from the Lab image by taking the first component of each pixel (Figure 1, right).

A Gaussian filter is created using the fspecial function with the kernel size of 11x11 and sigma = 2.0. Figure 2 (left) shows the application of the Gaussian filter to the lightness image. The result is slightly blurred and contains less noise. The lightness quantization is done by dividing the levels to 7 bins using a staircase function (Figure 2, right).
The LAB color space is reconstructed using the updated lightness image. It is converted back to RGB color space using the inverse color transformation. The result is shown in Figure 3 (right) along with the original image for comparison. It is similar to the given reference example created in Photoshop.

**Implementation: Sketch effect**

The Matlab code for the sketch drawing effect is stored in sketch.m. The original image is supplied as a function argument (Figure 4, left). The original image is converted from RGB color space to LAB color space (Wyszecki and Styles, 1982). First, a RGB to Lab color transformation is created using the makecform function. Then, the color transformation is performed using the applycform function. Finally,
the lightness channel is extracted from the Lab image by taking the first component of each pixel (Figure 4, right).

![Figure 4: Left: The original image. Right: The lightness channel of the Lab color space.](image)

Gaussian filters are created using the fspecial function with the kernel size of 9x9. The values of sigma are 1 and sqrt(1.6), as suggested by Winnemöller et al. (2006). Figure 5 shows the application of both filters to the lightness image. There is not much difference, but a close inspection reveals the different suppression of noise which is mostly noticeable in the hair.

![Figure 5: Applying Gaussian filters to the lightness image.](image)

Edge detection is done by computing the difference of Gaussians (Winnemöller et al., 2006). The Gaussian filters are created using the fspecial function. Then the edges are created by applying the filters to the lightness image using the imfilter function. The amplified edges have the appearance of a negative image (Figure 6, left). However, applying a simple inversion would look wrong. A better adjustment
using a smoothed step function is implemented by Winnemöller et al. (2006). Using their adjustment in this project resulted in mostly white images, no matter what parameters were chosen. Instead, a simplified processing was implemented by shifting the edge image by 1 for non-edge areas to become white. The pixel values are constrained to be in the valid color range using min and max functions. The final result is shown in Figure 6 (right). It is similar to the given reference example created in Photoshop.

![Figure 6: Edge detection.](image)

*Left: The edges as the difference of Gaussians (amplified). Right: The value of 1 is added for non-edge areas to become white.*

**Implementation: Gilding effect**

The Matlab code for the gilding effect is stored in gilding.m. The original image is supplied as a function argument (Figure 7, left). The input image is converted to grayscale using the rgb2gray function (Figure 7, right).

![Figure 7: Left: The original image. Right: The grayscale image.](image)
A Gaussian filter is created using the 'fspecial' function with the kernel size of 9x9. The value of sigma is 1.0 which was selected using a trial and error method until a satisfied end result was achieved. Figure 8 (left) shows the blurred image after applying the Gaussian filter which contains less tiny details and noise. Then the embossing filter (Vandevenne, 2007) is applied to the blurred image which creates highlights and shadows at brightness boundaries (Figure 8, right).

The embossed image in Figure 8 (right) is too dark. Scaling and shifting is used for pixel values to obtain a more clear image in Figure 9 (left). The result is colorized by modulating pixel values with yellow color components (Figure 9, right).

**Implementation: Other effects using built-in Matlab functions**

Built-in Matlab functions are used to create additional effects. Dithering operation uses the dither function to pointillize the image. Dilation operation uses the imdilate
function with a circular structuring element. Inversion operation uses imcomplement function. The corresponding results are shown in Figure 10.

![Image of dithering effect, dilation effect, and inversion effect](image.png)

*Figure 10: Left: The dithering effect. Middle: The dilation effect. Right: The inversion effect.*

**Implementation: Matlab GUI**

A GUI was designed to provide a simple and convenient way to use the filtering functions. The image and button controls were created (and edited) using the 'guide' command in the MATLAB Command Window and stored in the file gui.fig. The button names were modified using the Property Inspector context menu. The file gui.m was created automatically with event handlers to manage mouse clicks. The following functions were customized in gui.m:

- loadButton_Callback – loads the input image and shows it at the top;
- cartoonButton_Callback – applies the cartoon effect and shows it at the bottom;
- sketchButton_Callback – applies the sketch effect and shows it at the bottom;
- gildingButton_Callback – applies the gilding effect and shows it at the bottom;
- ditherButton_Callback – applies the dithering and shows it at the bottom;
- dilateButton_Callback – applies the dilation and shows it at the bottom;
- invertButton_Callback – applies the inversion and shows it at the bottom.
Figure 11: Graphic User Interface (GUI)
More Results

Figure 12: results on my third child

Figure 13: more results on my second daughter
Discussion

The implementation of artistic image filtering produced acceptable results in this project. The major problem with the cartoon effect was the appearance of false colors after color quantization. Limiting quantization to the lightness channel worked much better at preserving original colors. Some authors add black edges after quantization to outline uniform color areas in the image. However, the stippled look of sharp edges does not work well in all cases and was not considered in this project. The sketch effect is based on edge detection in the original image and worked well with minor adjustments. The gilding effect is based on the embossing operation and resulted in a typical 3D look right away. For other effects, several built-in Matlab functions were used to achieve artistic results. These include pointillizing the image, dilation operation, and inversion operation.

Conclusion

Matlab is a powerful package which is suitable for more than just pure computation. It has a number of image manipulation functions for basic filtering operations. It also allows to create specialized scripts for advanced processing of images. Using image filtering as well as image adjustment, image arithmetic, color conversion, and color quantization allows to create modified photographs with various effects applied. The results presented in this project can be improved in the future by incorporating more complex algorithms like bilateral abstraction, image based warping, edge thinning, etc. In addition, different kinds of effects can be devised to diversify the available operations for image manipulation.
References


