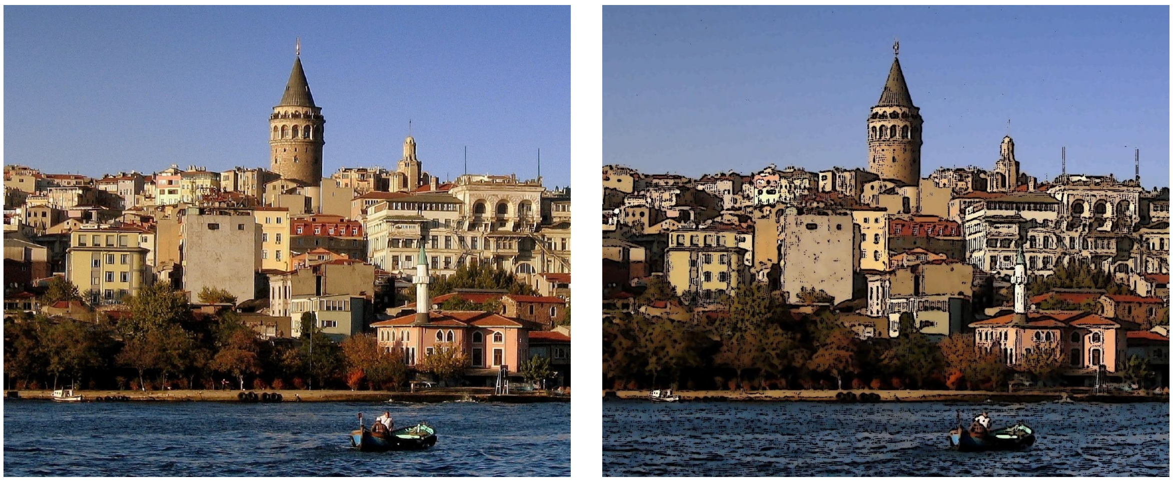
Assignment 1

Due on November 14, 2023 (23:59:59)



a) Input b) Result

Figure 1: Input image and cartoon result of it

Giving Cartoon Effect to Colorful Images

Background

Image filtering is one of the most fundamental tasks in Image Processing. It has been used for very different purposes such as image smoothing and edge detection. The main aim of image smoothing is to remove the high frequency component of the given image and to obtain the low frequency component. Basically, smoothing is carried out by convolving an image with a low-pass filter like the Gaussian filter kernel. The goal of edge detection is to determine the pixels where the brightness values are changed abruptly. Sobel and Prewitt filters are some of the examples to edge detection filters.

Many image editing tools like Photoshop let the user to perform some special filters on the images for various artistic effects. For example, one can obtain cartoon like images, pencil drawing images, etc. Real-Time Video Abstraction by Winnemoller et al.[1] is one such study where the cartoon-like outputs are obtained by as follows: A color image is first smoothed by a non-linear image filter, and then the edges are extracted. After that the smoothed image is quantized to represent the image with less number of colors for the cartoon effect. Lastly, the edges and the quantized image are combined. You may refer to the article for the details and to understand how those processes are performed.

Overview

The goal of this assignment is to implement a simplified version of Real-Time Video Abstraction[1] by using simple image filters. A sample image showing a possible result is given above (Figure 1).

Details

Your program will take a color image as input and produce a cartoon like version of the input image as output. Specifically, you should carry out the following steps:

1. **Image Smoothing**
   You will convolve the input image with a Gaussian filter or a median filter for this purpose. You should play with the filter parameters and comment on the results you obtained (the effects of the sigma parameter for the Gaussian filter, kernel width (size), etc. In Python, you can use the functions such as: (”scipy.ndimage.gaussian_filter”, ”scipy.ndimage.convolve”, ”scipy.ndimage.median_filter”)
2. **Edge Detection**

   (a) You can use thresholded Difference of Gaussian filter to obtain pencil sketching effect. To do this you will define DoG kernel as the differences of two Gaussian functions(DoG) as shown in Equation 1 and convolute it with original image(Equation 2). Lastly you will threshold filtered image with small number ϵ(Equation 3).

$$D_{\sigma,k} = G_\sigma(x) - G_{k\sigma}(x) : k > 1 \tag{1}$$

$$I_f = D_{\sigma,k} * I \tag{2}$$

$$T_\epsilon(n) = \begin{cases} 1 & \text{if } I_f \geq \epsilon \\ 0 & \text{else} \end{cases} \tag{3}$$

3. **Image Quantization**
   Purpose of quantization is to simplify the color values for a cartoon like effect. For better results you should convert RGB values to Lab or HSV color spaces and quantize (L)uminance or (V)alue channels and again convert them to RGB values. You can also quantize the R,G and B channels separately.
4. **Combining Edge and Quantized Image**
   After you extract the image edges (from the smoothed image) and quantize the smoothed image, you should combine edges and quantized image. For this step,

you should take the inverse of the estimated edges values and multiply it with the quantized image for each channel.

What to Hand In

You are required to submit all your report along with PDF format. For that purpose, prepare a folder containing:

- README.txt *(text file containing details about your assignment)*
- code/ *(directory containing all your code)*
- report/ *(directory containing all your documents, including your images)*
- report/data/ *(including your data images)*
- report/result/ *(including your result images)*
- report/pset1.pdf *(PDF report)*

Archive this folder as **studentid_pset1.zip** and send to **submit system**.

Your report should contain a brief overview of the problem, the details of your approach, and the results of your algorithm on at least 5 images (especially including different scene details) with your comments. Show the results of all of the main steps (smoothing, edge detection, quantization, final cartoon-like result) and with different filter selections or parameters for your implementation (such as ϵ, σ or k). If your algorithm failed to give a satisfactory result on a particular image, provide a brief explanation of the reason(s).

References

[1] Holger Winnemöller, Sven C Olsen, and Bruce Gooch. Real-time video abstraction. In ACM Transactions On Graphics (TOG), volume 25, pages 1221-1226. ACM, 2006.