Image Watercolorization effect

VR 2019 Final Project Report, B06902109 林仁傑

1. Introduction

For this project, I referred to the Color Intensity Alteration method[1] proposed by Fuh and Tung. In order to achieve watercolorization, first we apply a modified heat transfer differential equation to diffuse the image, then filter the image with a spatial mask algorithm.

2. Details

First, consider the standard heat transfer differential equation in 2D:

$$\frac{\partial u}{\partial t} = k(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2})$$

where denotes the temperature, t denotes time, and k denotes the diffusivity of the medium on which temperature is transferring.

We intend to simulate heat transfer by treating the value of each channel as temperature. So we alter the heat equation into discrete form:

$$u_{n+1} = u_n + k(\frac{\partial^2 u_n}{\partial x^2} + \frac{\partial^2 u_n}{\partial y^2})\Delta t$$

Where u_n denotes the temperature (or color intensity) of the nth iteration, and delta_t denotes the time step. The proposed value is

0.1; In my own experiment, a smaller value will blur the image too much.

However, we do not wish to blur the edge of significant regions, as the original heat equation evenly diffuses the image. In order to achieve this, we can make use of the diffusivity k:

$$k = e^{-(\frac{\sqrt{(\frac{\partial u_n}{\partial x})^2+(\frac{\partial u_n}{\partial y})^2}}{a})^2}$$

(a is just a pure scalar, and will be defined later)

You can see that k is no longer a constant, but a gaussian-distribution like equation – the numerator of the exponent is actually the Sobel operator, i.e. the edge of the image in each iteration. Since k goes to 0 as the Sobel value increases i.e. on the edge, we achieved the effect of reducing diffusion on the edge of the image.

Next step is the spatial mask algorithm. The proposed method is as follows:

For each pixel in the heat-transferred image:

For a k*k sized region at which the pixel is centered:

1. Create another k*k image where we quantize all pixels

into 20 levels with the following equation:

$$Level = \frac{R + G + B}{3 \times 255} \times 20$$

2. Calculate the level that has the highest occurrence.
3. For all pixels corresponding to the level with highest occurrence, calculate their average RGB values.
4. Replace the target pixel (the centered one) with the average RGB value.

The paper did not explain it very clearly such that it raises confusion: whether we should do it iteratively (such that previous pixels affect the pixels after) or simultaneously (all pixels calculate their replacement with the original image). From my own experiment, the iterative method creates obvious discontinued gradients that does not look like watercolor, and the simultaneous method creates a much softer gradient that looks much like watercolor, so I decided to adopt the latter.

Finally, the paper only provides the time step delta_t as 0.1, and did not provide any other. Through some experiments, I derived the optimal value for mask size, iterations, and the alpha in the diffusivity

equation k, which are mask size 9 for 512*512 images, 5 iterations for heat equation (independent of image size), and alpha = 1. The iteration number should not be modified at any time, for too large a number blurs the image too much, and the opposite basically blurs it too less. The value of alpha affects how much diffusion is applied to edges. However, the mask size should be adjusted according the the image size; Too large a mask simplifies the image too much such that it becomes some random color segments, and too small a mask does not quantize the color enough to show the watercolor effect. The final code and results using the same image in the paper can be found on Github[2].

3. Difficulties

As I mentioned, the paper is not exactly detailed: important notations are not very clear in meaning, and conflicting symbols occur (for example, Step 8. in the paper). The discrete form of heat equation also confuses me at first, because the number of iteration was superscripted as opposed to the common subscription, therefore I initially understood it as power of u, and could not continue.

4. Coding

A. Environment

Ubuntu 18.04

B. Required Packages

g++ 7.4.0, libopencv-dev cmake

C. How to compile

Run “make” under project directory

D. Explanation and usage

Refer to comments in the source files.

5. References

1. “Image Watercolorization Based on Color Intensity Alteration”, by C.S.Fuh and T.C.Tung, NTU - “https://www.csie.ntu.edu.tw/~fuh/personal/ImageWatercolorizationBasedonColorIntensityAlteration.pdf”
2. https://github.com/Agritite/VR2019_Term