CSE 528 Final Report

Professor: Hong Qin

Yishuo Wang

108533945

Abstract

In computer graphics, image texture filtering technology or texture smoothing technology is the method used to determine the texture color for a texture mapped pixel, using the colors (RGB elements of a pixel) of neighbor pixels. Due to this topic is not a brand new one, with the development of the technology in computer graphic area, people try and build many methods to filter the texture of the image. In this paper, we will introduce a new filtering-based method for decomposing an image into structures and textures[1]. The method smooths remove the texture by smoothing the image gradients, which is different from other filtering algorithms. Also, it has a new gradient created in the method, named "Interval Gradient", which plays a important part in the method for filtering the image.

Keywords: texture filtering, interval gradient, 1D filtering

I. Introduction

As we all know, one image can be decomposed to structure and texture. In the image processing area, people has built many different filtering method based on different theory. The method in this paper is motivated by the paper "Guided Image Filtering"[2], but the way that author does filtering is totally different, which is faster and created a new operator, named "Interval Gradient".

The method actually has two parts. First, we want to get the "Rescaling Gradient". To do this, we must define a new type of gradient operator, called "Interval Gradient", represented by $\nabla\Omega()$[1]. Our purpose is that creating a new method to suppress texture while keeping the structure using this new operator, Interval Gradient.

After that, we create a new image filtering method(gradient-based). As I mentioned before, the method is motivated by "the Guided Image Filter", so the second part, the method will have some similar parts with "the Guided Image Filter". According the data we get from first step, we will build a help image first to support us filtering image instead of inputting a help image just like the "the Guided Image Filter". As a result, we can filter the texture effectively and correctly while keeping the structure.

II. Get the Rescaling Gradient by Interval Gradient

In this method, we will filtering the image by doing with $1D$ dimension, which is the proposed method that composed of 1D local filters, similarly to the method in [3].In a $1D$ signal I, we have a $1D$ local window, and the size of the window is $(1, 2 * cell(3 * \sigma) + 1)$. During the whole paper, I will use σ as 3, so we can get the size of the window in this paper is $(1, 19)$. Also, we will use p to represent pixel.

II-1 Computing Normal Gradient

We will start with computing the normal gradient of each pixel. In a $1D$ discrete signal I, and for each pixel p, we can compute our normal gradient by measuring the difference between two adjacent pixel. Following is the equation

$$(\nabla I)_p = I_{p+1} - I_p \quad (1)$$

II-2 Computing Interval Gradient

On the other hand, for computing our interval gradient, we need to use both the left side and right side of the pixel. And the equation is

$$(\nabla_\Omega I)_p = \mathbf{g}^r_\sigma(I_p) - \mathbf{g}^l_\sigma(I_p) \quad (2)$$

where $\mathbf{g}^r$ and $\mathbf{g}^l$ respectively, represent left and right clipped $1D$ Gaussian filter functions defined by

$$\begin{aligned} \mathbf{g}^r_\sigma(I_p) &= \tfrac{1}{k_r} \textstyle\sum_{n\in\Omega(p)} \omega_\sigma(n-p-1) I_n \\ \mathbf{g}^l_\sigma(I_p) &= \tfrac{1}{k_l} \textstyle\sum_{n\in\Omega(p)} \omega_\sigma(p-n) I_n \qquad (3) \end{aligned}$$

ω_σ is the clipped exponential weighting function with a scale parameter σ:

$$\omega_\sigma(x) = \begin{cases} \exp(-\frac{x^2}{2\sigma^2}) & if\ x \geq 0 \\ 0 & otherwise \end{cases} \quad (4)$$

and k_r and k_l are normalizing coefficients defined as

$$\begin{aligned} k_r &= \textstyle\sum_{n\in\Omega(p)} \omega_\sigma(n-p-1) \\ k_l &= \textstyle\sum_{n\in\Omega(p)} \omega_\sigma(p-n) \qquad (5) \end{aligned}$$

From the equation above, we can see that we need to only care about the right side pixels of each pixel in the $\mathbf{g}_\sigma$, and only care about the left side pixels of each pixel in the $\mathbf{g}_\sigma^l$, due to other cases are all returning zero from the equation (4).

The difference between normal gradient and interval gradient is that interval gradient computes the difference between the weighted averages of the left and right parts of the signal around a pixel[1], not only the pixel value like normal gradient.

II-3 Computing Rescaling Gradient

We have got the normal gradient and the interval gradient of the input image. For the black-white (1 channel), we just simply compare the sign of the normal gradient and the interval gradient in order to get the Rescaling gradient, and we use $(\nabla' I)_p$ to represent it. The equation is

$$(\nabla' I)_p = \begin{cases} (\nabla I)_p \cdot w_p & if\ sign((\nabla I)_p) = sign((\nabla_\Omega I)_p) \\ 0 & otherwise \end{cases} \quad (6)$$

with the ω_p is the rescaling weight,

$$w_p = \min\left(1, \frac{|(\nabla_\Omega I)_p| + \epsilon_s}{|(\nabla I)_p| + \epsilon_s}\right) \quad (7)$$

However, for filtering the color images, we need to use the gradient of the color channels to compute the rescaling weight. Definitely we will use the color filtering method to do the project. So the rescaling weight of the color image is

$$w_p = \min\left(1, \frac{\sum_{c\in\{r,g,b\}} |(\nabla_\Omega I^c)_p| + \epsilon_s}{\sum_{c\in\{r,g,b\}} |(\nabla I^c)_p| + \epsilon_s}\right) \quad (8)$$

For the border case, we just copy the border most pixel (left most and right most) to do the filtering. Due to the method is $1D$, we need to do all of these steps two times with one is x direction (row by row) and another one is y direction (column by column).

In order to see the difference intuitively, we will show the oscillations of different situations in the graph.

In figure 1, we can see the changes by various types of signals and gradients.

III. Get the help image and get the filtering result

As I mentioned before, this method is motivated by the "Guided Image Filtering"[2]. "Guided Image Filtering"[2] uses a help image as one of the input, while we will build the

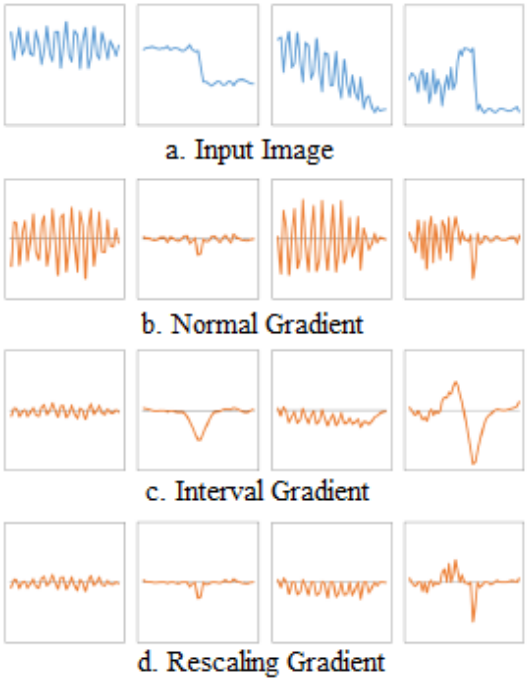


Figure 1: From left to right, they are texture, structure edge, texture with shading and noisy valley near a structure edge.

help image by ourselves in the method. The data we need to use is the input image and the rescaling gradient. So, let's construct a temporary signal R (help image) through the accumulation:

$$R_p = I_0 + \sum_{k=0}^{p-1} (\nabla' I)_k, p \in \{0, 1, \ldots, N_p\} \quad (8)$$

I_0 is the leftmost pixel value of each row in I, or the topmost pixel value of each column in I. Then, each time after we build the R_p, we will want to find the best linear transform coefficients a_p and b_p that minimize:

$$\arg\min_{a_p, b_p} \sum_{n \in N(p)} w_n \Big((a_p R_n + b_p - I_n)^2 + \epsilon a_p^2 \Big) \quad (9)$$

and the w_n is the Gaussian weight with scale parameter $\sigma = 3$ in our paper, which is

$$w_n = \exp(-\frac{(n-p)^2}{a\sigma^2}) \quad (10)$$

Then the original paper gives up the equation for linear transform coefficients a_p and b_p,

$$a_p = \frac{\mathbf{g}_\sigma((RI)_p) - \mathbf{g}_\sigma(R_p)\mathbf{g}_\sigma(I_p)}{\mathbf{g}_\sigma(R_p^2) - \mathbf{g}_\sigma(R_p)^2 + \epsilon}$$

$$b_p = \mathbf{g}_\sigma(I_p) - a_p \cdot \mathbf{g}_\sigma(R_p) \quad (11)$$

In the end, we will use a_p and b_p to build the new image,

$$S_p = \mathbf{g}_\sigma(a_p)R_p + \mathbf{g}_\sigma(b_p) \quad (12)$$

Certainly, just filtering one time is not enough for the purpose. When we finish one time filtering, the program will go to do the next time filtering automatically, until the current result meets our request. Therefore, the iteration will stop when the difference is less than a given threshold δ, so that

$$\min\left(\frac{1}{N}\sum_p \left(w_{px}^{t-1} - w_{px}^t\right)^2, \frac{1}{N}\sum_p \left(w_{py}^{t-1} - w_{py}^t\right)^2\right) < \delta_t \quad (13)$$

Meanwhile, following is the Algorithm,

Input: image I
Output: texture filtered image S
  while Eq. (14) is false **do**
    $(\nabla_x^{'} I) \leftarrow$ rescaled gradients along x-axis
    $(\nabla_y^{'} I) \leftarrow$ rescaled gradients along y-axis
    for $i := 0 \ldots (N_d - 1)$ **do**
      $\sigma_i \leftarrow \sigma \cdot \sqrt{3}\frac{2^{N_d - i}}{\sqrt{4^{N_d} - 1}}$
      $S_x \leftarrow$ 1D filtering result along x-axis

      $I \leftarrow S_x$
      $S_y \leftarrow$ 1D filtering result along y-axis
      $I \leftarrow S_y$
    end for
  end while
  $S \leftarrow I$

Figure 2: pseudo-code of the algorithm

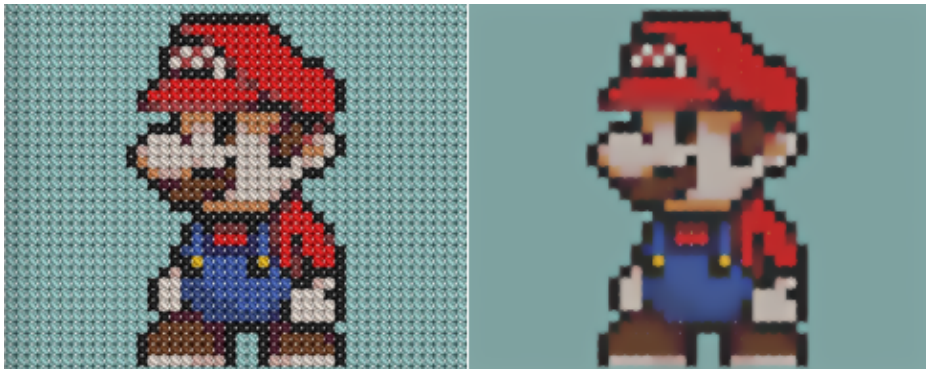
IV. Experimental Process and Results

At the beginning, I tried to use 32×32 chessboard(black-white picture) for easier understanding. However, due to I think the chessboard pictures does not have a lot of clear texture and I do not know what the correct output should be, so the result maybe keep the same as the input. Therefore, I did not keeping using the chessboard picture.

In the midterm presentation, my project can run, but it just gives me a totally wrong result. Finally, I know the mistake I did is the data type problem. After I debug and change the type to the "float", I got a better result. Which is the figure 3.

Figure 3: First time get a over smooth wrong result

Then, I figure out the bug and get good result. Following images are some results I got.



V. Conclusion

In the paper, we have talked about the novel filtering-based texture filter method with the new operator, named “Interval Gradient”, which is super useful for accurately suppressing the texture while keeping the structure of the images. On the other hand, the method filters the image by focusing on filtering image gradients instead of filtering the image color directly. As a result, the method can doing the structure and texture decomposition efficiently and accurately.

Reference

[1] L. Hyunjoon, J. Junho, K. Junho, L. Seungyong, “Structure-Texture Decomposition of Images with Interval Gradient”, Computer Graphics forum, V.00(2016), N.0 pp.1-13

[2] HE K., SUN J., TANG X.: “Guided image filtering”. pami 35, 6 (2010), 1397¨C1409.

[3] GASTAL E. S. L., OLIVEIRA M. M.: Domain transform for edge-aware image and video processing. ACM Transactions on Graphics 30 (2011), 69:1¡§C69:12.