

An Analysis of Guided Image Filtering

Dhyanil Mehta

M.Tech(ICT) ML

DAIICT

Gandhinagar, Gujarat, India

202011032@daiict.ac.in

Kishan Vaishnani

M.Tech(ICT) ML

DAIICT

Gandhinagar, Gujarat, India

202011004@daiict.ac.in

Abstract—Guided Image Filtering [1] uses a guided filter which can be used as an edge-preserving smoothing operator similar to the popular *bilateral filter* [2], but with better behaviour near edges. It can be also be used to transfer the structures of the guidance image to the filtering output for applications like dehazing. This paper analyzes the performance of guided image filtering on various applications like Noise Removal, Image Enhancement, Structure Transfer, and Flash/No-Flash Denoising. But it can also be used in one more application which is to improve the segmentation output generated by a segmentation network. The paper discusses about it in the analysis and discussion section.

Index Terms—Guided filter, bilateral filter, linear time filtering

I. INTRODUCTION

In fields like computer vision and computer graphics, various applications involve some kind of image filtering to remove/suppress or extract contents in an image. In the Advance Image processing course we already learn basic filters like Gaussian, Laplacian, Sobel filters. After applying this filter are not able to preserve the edge- edge also smoothen by this filters. To preserve edge in the image and also for calculating gradient of the Guided Image Filtering uses an explicit filter called *guided filter*. The filter performs a linear transformation on the guidance image at a local level to generate the filtering output. Guided filter is basically used for three types of filtering:-

A. Edge-Preserving Filtering

When the guidance image is the same as the filtering input, the guided filter will filter out the noise in the input image while preserving clear edges. Specifically, one can define what is a “*flat patch*” or a “*high variance patch*” by the parameter ϵ of the guided filter. Those patches with variance much lower than the parameter ϵ will be smoothed, and those with variances much higher than ϵ will be preserved. The role of the range variance σ_r^2 in the bilateral filter is similar to ϵ in the guided filter. Both of them define “where are the edge(s) with high variance patches that ought to be kept and where are the noise/flat patches that ought to be smoothed.”

B. Gradient-Preserving Filtering

When using the bilateral filter [2] to filter an image, some gradient reversal artifacts may appear on the edges [3]. The guided filter performs better in avoiding gradient reversal.

Moreover, in some cases, it can be ensured that gradient reversal does not occur.

C. Structure-Transferring Filtering

Due to the local linear model of guided filter, it is possible to transfer the structure from the guidance image to the filtering output. This property enables some special filtering-based applications, such as feathering, matting and dehazing. Despite of this much work, guided filter takes on $O(N)$ time complexity in all such applications. Guidance image helps in calculating the gradient of the image.

In this report we follow the base paper Guided Image Filtering [1] and derive the guided filter with mathematical equations and reproduce the same results as the base paper got by filtering through guided filter. We reproduced the code for RGB and grayscale images in python language. We analyze the results based on structure transfer filtering of guided filter and general guided filter on RGB and grayscale images. This report is our basic understanding of guided filter.

II. RELATED WORK

The bilateral filter [2] is perhaps the most popular among explicit edge-preserving filters. It computes the filtering output at each pixel as the average of neighboring pixels, weighted by the Gaussian of both spatial and intensity distance. The bilateral filter smooths the image while preserving edges. It has been widely used in various applications. It is generalized to the *joint bilateral filter* in [4], where the weights are computed from another guidance image rather than the filtering input. The joint bilateral filter is particularly favored when the image to be filtered is not reliable for edge information, i.e., very noisy image or an intermediate result, such as in flash/no-flash denoising [4], etc.

The bilateral filter despite its popularity has some limitations. It has been noticed that the bilateral filter may suffer from “*gradient reversal*” artifacts [3]. This is because the abrupt change of pixel values on the edge. These artifacts are inherent and hard to avoid, because edges usually appear in all kinds of pictures.

Edge-preserving filtering can also be achieved by nonaverage filters. The median filter [5] is a well-known edge-aware operator. But the nonaverage filters can often be computationally expensive.

After all that, Gastal and Oliveira [6] proposed another $O(N)$ time filter known as the *Domain Transform* filter. The basic idea is to iteratively and separably apply 1D edge-aware filters. The $O(N)$ time complexity is achieved by integral images or recursive filtering.

III. THE APPROACH

One key assumption of the guided filter is that the relation between guidance Image I and output image q is linear.

q is the linear transformed of the I in a window w_k centered pixel k . Following equation represents the linearity in between the guidance image I and output image q .

$$q_i = a_k I_i + b_k, \forall_i \in w_k \quad (1)$$

If we take first derivative of the equation (1) then that will give us the $\Delta q = a \Delta I$. This shows that if the guidance image have edge then only output image have an edge otherwise not.

Model the output q as the input p and subtract unwanted noise and texture from it.

$$q_i = p_i - n_i \quad (2)$$

In the above equation (1) and (2)

q_i is the i^{th} output pixel

p_i is the i^{th} input pixel

n_i is the i^{th} pixel of noise components

I_i is the i^{th} guidance image pixel

(a_k, b_k) are some linear coefficients assumed to be constant in w_k .

By the use of equation (1) and (2), we are driving equation for the noise parameter

$$n_i = p_i - a_k I_i - b_k \quad (3)$$

Now, we need to derive cost function of the equation (3) and specifically we minimize the following cost function in the window w_k

$$E(a_k, b_k) = \sum_{i \in \omega_k} ((p_i - a_k I_i - b_k)^2 + \epsilon a_k^2) \quad (4)$$

In the above equation (4):

ϵ is a regularization parameter penalizing large a_k .

ω_k is a window centered at the pixel k .

Above equation (4) is the linear ridge regression model and it's solution [7] is given by

$$a_k = \frac{\frac{1}{|\omega|} \sum_{i \in \omega_k} I_i p_i - \mu_k \bar{p}_k}{\sigma_k^2 + \epsilon} \quad (5)$$

$$b_k = \bar{p}_k - a_k \mu_k \quad (6)$$

In the above equation (5) and (6):

μ_k and σ_k^2 are the mean and variance of I in ω_k .

$|\omega|$ is the the number of pixel in ω_k .

$\bar{p}_k = \frac{1}{|\omega|} \sum_{i \in \omega_k} p_i$ is the mean of p in ω_k .

Now, we consider equation (1) for single pixel, if we apply this window in all the pixel of the image then we need to take average of that q_i from all the different window. So equation will be

$$q_i = \frac{1}{|\omega|} \sum_{k|i \in \omega_k} (a_k I_i + b_k) \quad (7)$$

$$q_i = \bar{a}_i I_i + \bar{b}_i \quad (8)$$

In above equation (8)

\bar{a}_i and \bar{b}_i is the average coefficient of all windows overlapping i .

IV. IMPLEMENTATION AND RESULTS

A. Algorithm

The algorithm given in the original paper was used and implemented in python ¹.

1) *Grayscale Image Filtering*: For grayscale images, the algorithm is basic and goes as follows:

$$mean_I = f_{mean}(I)$$

$$mean_p = f_{mean}(p)$$

$$mean_{I^2} = f_{mean}(I * I)$$

$$mean_{Ip} = f_{mean}(I * p)$$

$$var_I = mean_{I^2} - mean_I^2$$

$$var_{Ip} = mean_{Ip} - mean_I * mean_p$$

$$a = mean_{Ip} / (var_I + \epsilon)$$

$$b = mean_p - a * mean_I$$

$$mean_a = f_{mean}(a)$$

$$mean_b = f_{mean}(b)$$

$$q = mean_a * I + mean_b$$

where f_{mean} is the box filter (average filter) or sometimes gaussian filter with the complexity $O(N)$.

2) *Color Image Filtering*: For an RGB image, the algorithm for grayscale filtering can be applied to each of the color channels by considering each channel as a grayscale image in itself and then concatenate the results.

B. Results

We analyzed the results of Guided Image Filtering for different applications and techniques. The images [1] used in the original paper were used in this paper also. *Table I* shows the results of applying guided filter with different guidance images and filtering inputs for different applications.

For the first application of noise removal in both the grayscale and color images, noisy images corresponding to the original images were generated by adding Gaussian noise with noise factor of 50 and 100 for gray-scale image and color image respectively.

For structure transfer filtering, ROI mask was used as an input and original image as guidance image so as to transfer the structure of the image onto the mask.

¹<https://www.python.org/>




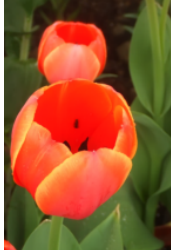

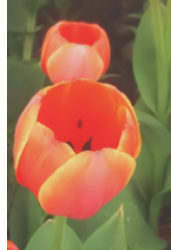
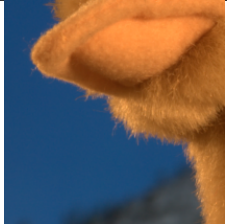

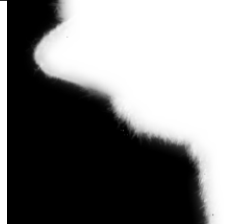

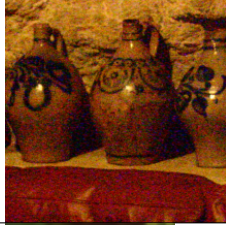
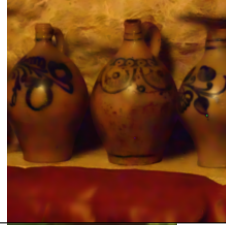
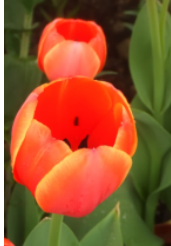
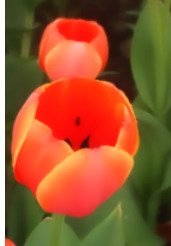
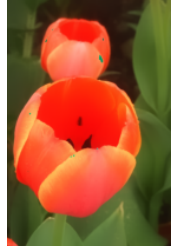
Application	Parameter	Guidance Image	Input Image	GF applied Image
Noise Removal in Grayscale Image	$r = 21$ $\epsilon = 0.001$			
Noise Removal in RGB Image	$r = 41$ $\epsilon = 0.03^2$			
Structure Transfer Filtering	$r = 41$ $\epsilon = 0.0045$			
Flash/No-flash Denoising	$r = 15$ $\epsilon = 0.003$			
Image Enhancement	$r = 61$ $\epsilon = 0.001$			

TABLE I: Applied Guided filter for different applications

For flash/no-flash denoising, image with flash was given as a guidance image and the no-flash image was given as a filtering input.

For image enhancement filtering, original image was given as guidance image and its smoothed version was given as a filtering input. The result produces an enhanced version of the original image.

V. ANALYSIS AND DISCUSSION

From the above results given in Table I, we can observe that the filter with normal filter r and a small ϵ works as well as the bilateral filter but without the gradient reversal artifacts for noise removal by edge-preserving filtering for both grayscale and color images.

In Structure transfer filtering, although somewhat blurred the structure of the guidance image can be seen transferred as the edges in the binary mask are now representing the small details which were present in the original image.

In the last two applications, we can observe that in both the cases the edges are preserved while performing the task to be done viz flash/no-flash denoising and image enhancement. Even the edges around the darker side were preserved in the denoising output.

a) *Guided filtering in semantic segmentation:* From the results of structure transfer filtering, the edges in the binary mask became more clear. This result can also be used in improving results in tasks like semantic segmentation [8]. Semantic segmentation takes in a normal color image and

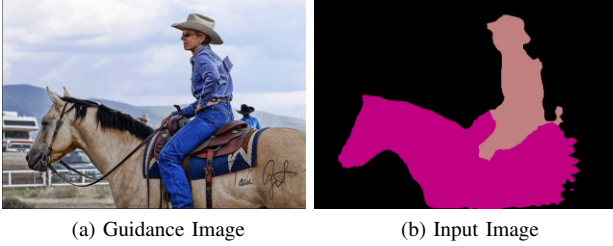


Fig. 1: Input Image to Guidance filter. Parameters were $r = 21$ and $\epsilon = 0.08$

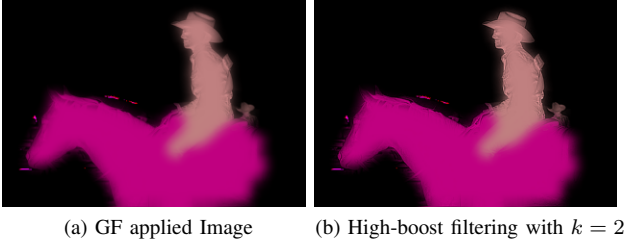


Fig. 2: Output Image

generates a masked output with different labels/objects represented by different pixel values. However, the generated result is not exact at the edges. This is due to the fact that the groundtruth label itself is not accurate around the edges due to human errors. To overcome this limitation we can use structure transfer filtering with the help of guided image filter on the output of the segmentation network with the input image as the guidance image. This will make the output preserve its accuracy around the edges. Guided filter can even be used as a post processing layer in the network.

To support our argument we took an image from the Pascal VOC 2012 [8] and its segmented output and performed structure transfer filtering with the help of guided filter and highboosted the result for more sharper edges. What we see in Figure 1 is that the segmented image is not tightly bound around the edges as it should be. By applying guided filter using original image as the guidance image and segmented image as the filtering input for structure transfer and then applying highboost filtering, we get a sharper and more detailed segmented output image. The unwanted artifacts in the output can be removed by tuning the parameters although it will affect the quality of the output.

VI. CONCLUSION

Guided Image Filtering has the same performance as the bilateral filter in edge-preserving filtering but without the gradient reversal artifacts. More than that, guided filter can also be used in a few other types of filtering and applications where it shows promising results. We also saw that by using structure transfer filtering on the output of semantic segmentation, we can get a more enhanced segmentation output which is strong around the edges.

REFERENCES

- [1] K. He, J. Sun and X. Tang, “Guided Image Filtering,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 6, pp. 1397-1409, June 2013, doi: 10.1109/TPAMI.2012.213.
- [2] C. Tomasi and R. Manduchi, “Bilateral Filtering for Gray and Color Images,” Proc. IEEE Int’l Computer Vision Conf., 1998.
- [3] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, “Edge-Preserving Decompositions for Multi-Scale Tone and Detail Manipulation,” Proc. ACM Siggraph, 2008.
- [4] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama, “Digital Photography with Flash and No-Flash Image Pairs,” Proc. ACM Siggraph, 2004.
- [5] R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, second ed. Prentice Hall, 2002.
- [6] E.S.L. Gastal and M.M. Oliveira, “Domain Transform for Edge-Aware Image and Video Processing,” ACM Trans. Graphics, vol. 30, no. 4, pp. 69:1-69:12, 2011.
- [7] N. Draper and H. Smith, *Applied Regression Analysis*, second ed. John Wiley, 1981.
- [8] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. *The Pascal visual object classes (VOC) challenge*. IJCV, 88(2):303–338, 2010.