CONSTANT TIME BILATERAL FILTERING FOR COLOR IMAGES

Wei-Chih Tu, Ying-An Lai, and Shao-Yi Chien*

Graduate Institute of Electronics Engineering, National Taiwan University
No. 1, Sec. 4, Roosevelt Rd., Taipei 10617, Taiwan

ABSTRACT

Bilateral filtering is a commonly used technique in image processing. However, being nonlinear, it is computationally expensive. The situation gets worse while the filter radius grows up. Several works have been proposed to accelerate the computation. Nevertheless, most techniques are tailored for gray-scale image bilateral filtering or confined to specific kernel functions. In this paper, we propose a constant time bilateral filter for color images. Specifically, we extend an existing constant time bilateral filtering technique, which has been demonstrated the state-of-the-art for gray-scale images. We generalize the original approach as a three-stage algorithm. Based on the generalization, we explain the drawbacks of its naïve extension for color images and propose a solution that adapts to the image content. Experimental results demonstrate the effectiveness of our solution.

Index Terms— Constant time filtering, bilateral filtering, adaptive sampling

1. INTRODUCTION

Bilateral filtering [1] has long been an important technique in image processing. It smooths images while carefully preserves edges. The general form can be expressed as:

$$I^F(\mathbf{p}) = \frac{\sum_{\mathbf{q}\in\Omega(\mathbf{p})} g_s(\mathbf{p},\mathbf{q}) g_r(T(\mathbf{p}),T(\mathbf{q})) I(\mathbf{q})}{\sum_{\mathbf{q}\in\Omega(\mathbf{p})} g_s(\mathbf{p},\mathbf{q}) g_r(T(\mathbf{p}),T(\mathbf{q}))}, \quad (1)$$

where I and I^F denote the input and filtered image. The reference image T is used to determine the appearance affinity of pixels. T can be the same image as I or another image for joint bilateral filtering [2]. The filter response is a weighted sum of pixels in the neighborhood $\Omega(\mathbf{p})$ of center pixel $\mathbf{p}$. The filter weight contains spatial and range kernel functions g_s and g_r. Typically, both kernel functions are low-pass filters, such that spatial kernel assigns lower weights to farther pixels and range kernel assigns lower weights to pixels with larger appearance difference. The choice of range kernel function is essential to the edge-preserving effect of bilateral filter. One common choice for the range kernel function is the Gaussian kernel $g_r(x,y) = exp(-\frac{(x-y)^2}{2\sigma_r^2})$.

*Primary contact: wctu@media.ee.ntu.edu.tw

The bilateral filter for color images can also be expressed as Eq. (1), where $I(\mathbf{p})$ becomes a color vector and the range kernel takes the color difference of two pixels into account. For example, the Gaussian range kernel can be expressed as:

$$g_r(I(\mathbf{p}),I(\mathbf{q})) = exp(-\frac{\sum_{c\in r,g,b}(I_c(\mathbf{p})-I_c(\mathbf{q}))^2}{2\sigma_r^2}). \quad (2)$$

Owing to the wide application of bilateral filtering, the fast implementation of bilateral filtering has attracted much research interests in image processing. However, being nonlinear, common techniques like separable filtering are not directly applicable to the bilateral filtering. Let r be the filter radius, the complexity of direct computation is $O(r^2)$, which means the computational load gets higher when the filter radius grows up. The issue is critical since the image resolution is getting higher and higher today such that the corresponding filter radius is getting larger as well. Several works have been proposed to accelerate bilateral filtering either by numerical approximation [3, 4, 5, 6, 7] or by confining specific kernel functions that are cheaper to evaluate [8, 4]. The key idea behind is that we don't really need exact computation of the bilateral filter. Empirically [4], when the numerical accuracy is high enough (PSNR $>$ 35 dB), the approximated bilateral filter can still perform edge-preserving filtering.

Recently, $O(1)$ bilateral filtering becomes more favorable for its processing time being constant regardless of the filter radius. Porikli et al. [4] confined the spatial kernel as the box filter such that the bilateral filter can be evaluated from local histograms. They explored the redundancy of local histograms and proposed a constant time solution using integral histogram. The algorithm can be further accelerated by quantizing the histogram such that the number of bins is reduced. However, as reported in later literature [5], it can not produce accurate results with quantized histograms when the range parameter σ_r is small. Yang et al. [5] proposed another $O(1)$ implementation by decomposing the problem into a set of principle bilateral filtered image components (PBFIC). The method reduces computation by only conducting a few linear filtering and interpolates the others. The PBFIC method can be applied to arbitrary spatial and range kernel functions. Moreover, it is highly accurate even with few components. As reported in [5], 4 to 8 components are sufficient for gray-scale images, while histogram-based method [4] typically needs

more than 64 bins to achieve pleasant results. To the best of our knowledge, the PBFIC method is the state-of-the-art in terms of accuracy, efficiency and kernel generality for bilateral filtering of gray-scale images.

The fast bilateral filtering for color images is still on demand since all above methods are tailored for gray-scale images. Adams et al. [9, 10] proposed fast algorithms for high-dimensional Gaussian filtering. It can be applied to bilateral filtering for color images. However, the processing time of [9, 10] is considerably long compared to other techniques as reported in [7]. The advantage of [9, 10] is better demonstrated in non-local means filtering [11] where the number of dimensions is equivalent to the patch size. Yang et al. [7] presented a simple extension of the PBFIC method for color images. They proposed to take N intensities for each color channel and thus totally N^3 components will be evaluated. Even the PBFIC method becomes computationally expensive as the number of components grows up. For example, take 4 to 8 components for each channel, then there will be 4^3 to 8^3 components to evaluate. In this paper, we propose an extension of the PBFIC method for color images. Specifically, we generalize the PBFIC method as a three-stage algorithm. The three stages are sampling, filtering and interpolation. We notice that numerical accuracy is highly related to the sampling and interpolation strategies. As a result, we can improve the quality with better sampling and interpolation methods. As each stage is constant time for each pixel, the overall algorithm is also constant time for each pixel.

2. REVIEW OF THE PBFIC METHOD

For better understanding, we briefly review the principle bilateral filtered image component (PBFIC) method [5] here. Eq. (1) can be regarded as per-pixel division of two filter responses, J and W.

$$J(\mathbf{p}) = \sum_{\mathbf{q}\in\Omega(\mathbf{p})} g_s(\mathbf{p},\mathbf{q}) g_r(T(\mathbf{p}),T(\mathbf{q})) I(\mathbf{q}) \tag{3}$$

$$W(\mathbf{p}) = \sum_{\mathbf{q}\in\Omega(\mathbf{p})} g_s(\mathbf{p},\mathbf{q}) g_r(T(\mathbf{p}),T(\mathbf{q}))) \tag{4}$$

$$I^F(\mathbf{p}) = \frac{J(\mathbf{p})}{W(\mathbf{p})}. \tag{5}$$

J is the unnormalized bilateral filtered image and W is the normalization term. One can regard W as another unnormalized bilateral filtered image of an all-one image using the same weights. The PBFIC is defined as the bilateral filter response computed with fixed center pixel value k. Then Eq. (3) becomes

$$J^k(\mathbf{p}) = \sum_{\mathbf{q}\in\Omega(\mathbf{p})} g_s(\mathbf{p},\mathbf{q}) g_r(k,T(\mathbf{q})) I(\mathbf{q}). \tag{6}$$

The advantage of this design is that computing J^k is equivalent to spatial filtering of image $g_r(k,T)I$. Note that $g_r(k,T)I$ can be computed independently for each pixel. If the spatial filtering is $O(1)$, then computing J^k is also $O(1)$. W^k can be computed in the same way, resulting the overall complexity $O(1)$. If we compute J^k for every possible $k \in [0,255]$, then the exact bilateral filter response becomes

$$I^F(\mathbf{p}) = \frac{J^{T(\mathbf{p})}(\mathbf{p})}{W^{T(\mathbf{p})}(\mathbf{p})}. \tag{7}$$

As described in [5], we can approximate the bilateral filter by only computing a subset of possible intensities. For those pixels $T(\mathbf{p})$ exist in the subset, their filter responses are still exact. If $T(\mathbf{p})$ is skipped, let k_1 and k_2 be the two nearest intensities in the subset, assume $T(\mathbf{p}) = \alpha k_1 + (1-\alpha)k_2$, then the final result can be approximated by

$$I^F(\mathbf{p}) = \alpha \frac{J^{k_1}(\mathbf{p})}{W^{k_1}(\mathbf{p})} + (1-\alpha)\frac{J^{k_2}(\mathbf{p})}{W^{k_2}(\mathbf{p})}. \tag{8}$$

Empirically, 4 to 8 intensities are sufficient (PSNR $>$ 35 dB) for gray-scale images. As the computational load is the set of spatial filtering. Using fewer components turns in shorter processing time.

3. PROPOSED METHOD

3.1. The three-stage generalization

We reformulate the PBFIC method into three stages. They are *sampling, filtering and interpolation*. For the single channel version reviewed in previous section, the sampling step is to determine which intensities are used as the subset. The filtering step is the spatial filtering of image $g_r(k,T)I$ in Eq.(6). And the interpolation is bilinear interpolation with two nearest samples in the subset. As for the extension described in [7], the sampling step is to take N intensities for each channel and in total N^3 color combinations are used as the sample set. The filtering step is the spatial filtering of image $g_r(k,T)I$, where k denote a fixed color in the sample set. In fact, the color sample set forms the grid points of a grid structure in color space. The interpolation step is bilinear interpolation with the eight vertices of the nearest grid cell.

The three-stage generalization helps us understand ways to improve the PBFIC method. We notice that the image quality is highly related to the sampling and interpolation strategies and the computational load is the filtering stage. Obviously, more samples result in higher accuracy while it takes more filtering components. On the contrary, taking the same number of samples while with different sampling strategies also results in different quality. Take an extreme example, considering two cases where only two components are used for gray-scale image bilateral filtering. The first case is sampling at the possible dynamic range, so the sampled intensities are 0 and 255. The second strategy is that intensities are sampled as I_{min} and I_{max} according to the image histogram. In

some cases, $|I_{max} - I_{min}|$ can be far less than 255, so the filtered image quality of latter sampling outperforms the former.

Now consider the extension described in [7]. Though the resulting algorithm still has $O(1)$ complexity, it is a huge constant owing to the cubic factor N^3. To reduce computation, N should be smaller. However, smaller N means fewer representative colors are used and thus the approximation quality may be low due to larger interpolation errors. The authors of [7] suggested to use at least $4^3 = 64$ colors for accuracy. This sampling strategy is ineffective such that the large number of filtering components also results in longer processing time. We address this problem with better sampling strategy, which significantly reduces the number of samples needed and thus reduces the processing time.

3.2. Adaptive sampling in color space

The proposed adaptive sampling is based on the fact that color distribution of a natural image [12] is usually sparse in the color space. Therefore, we should take the image content into account to determine the principle color samples. Furthermore, the sampled colors should distributed evenly in color space to avoid large interpolation errors. We propose a strategy based on the Poisson disk sampling [13, 14]. The Poisson disk sampling generates samples that are stochastic, evenly spaced, but no closer to each other than a given disk radius r_s. Specifically, we take the colors of the reference image as the sample pool. Then we randomly pick a sample from the pool so the sampled color must be close enough to the distribution. We accept a new sample if the distance from it to any previously sampled color is larger than the disk diameter $2r_s$. When there is no space for a new sample, r_s is reduced. The procedure continues until we get enough number of samples. Note that the number of samples is no longer limited to the cubic number N^3. This advantage makes our algorithm more flexible. Fig. 1 shows an example. The scatter plot of all colors from a natural image is shown. The blue dots are the $3^3 = 27$ samples used in [7] and the red dots are generated from our adaptive strategy with the same number of samples.

3.3. $O(1)$ spatial filtering

The spatial filter used in this framework is required to be a $O(1)$ approach such that the final algorithm can have $O(1)$ complexity. Box filter can be implemented in $O(1)$ by integral image [15]. Gaussian filter can also be $O(1)$ using IIR implementation [16]. We adopt these two common spatial filters in the proposed framework.

3.4. Interpolation

The interpolation in single channel version is simply bilinear interpolation. The weights α and $(1 - \alpha)$ are used to balance

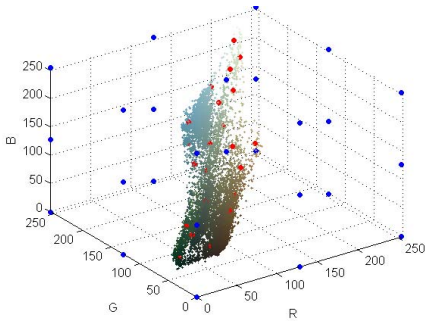


Fig. 1: Comparison of regular sampling [7] (blue dots) and our adaptive sampling strategy (red dots) using 27 samples.

two filtered components. Similarly, we propose to find the nearest colors in the sample set and use their filtered components to interpolate the final result. In three dimensional color space, at least four points are needed to form a volume to enclose a target point. As a result, we propose to find the four nearest sampled colors and combine their filtered components by assigning weights inverse proportional to their distance to the target color. Let the four nearest colors be $\mathbf{k}_1$, $\mathbf{k}_2$, $\mathbf{k}_3$ and $\mathbf{k}_4$. Then the final result can be obtained by

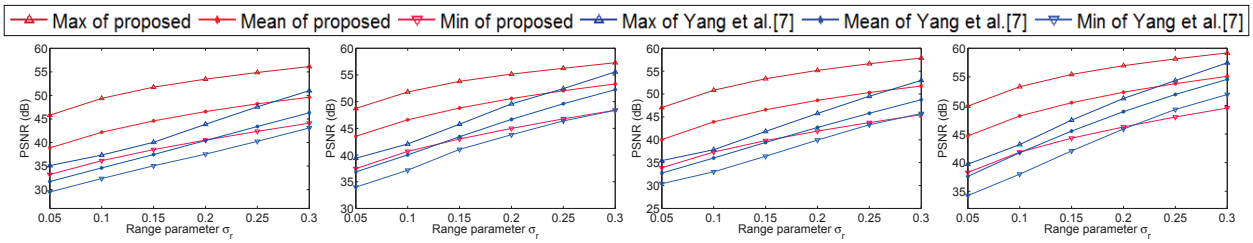
$$I^F(\mathbf{p}) = \frac{\sum_i \omega_i \frac{J^{\mathbf{k}_i}(\mathbf{p})}{W^{\mathbf{k}_i}(\mathbf{p})}}{\sum_i \omega_i}, \tag{9}$$

where ω_i is a function of distance d_i from a sampled color $\mathbf{k}_i$ to the target color. We set ω_i as the exponential weights $exp(-\frac{d_i}{2d_{min}})$, where d_{min} is the least distance in d_1, d_2, d_3 and d_4.

4. EXPERIMENTAL RESULTS

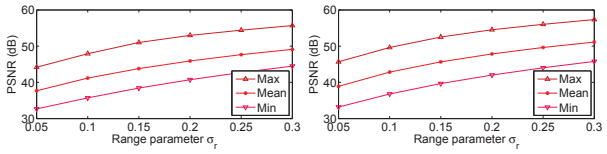
The bilateral filtering with box spatial and Gaussian range kernel (BsGr) and with Gaussian spatial Gaussian range kernel (GsGr) are tested. Test images are shown in Fig. 2. All these images are used in previous works to evaluate edge-preserving smoothing. Both numerical accuracy and processing time are evaluated.

Fig. 2: Standard images used to evaluate bilateral filtering.



(a) BsGr with 27 samples (b) BsGr with 64 samples (c) GsGr with 27 samples (d) GsGr with 64 samples

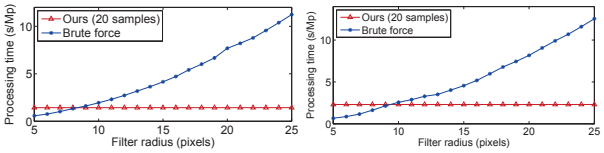
Fig. 3: Numerical accuracy evaluation with 3^3 and 4^3 sampled colors. The filter radius is fixed to 15 pixels and the range parameter σ_r is varying (shown in normalized intensity scale.)



(a) BsGr (b) GsGr

Fig. 4: Our method is accurate even using only 20 samples.

Fig. 5: GsGr bilateral filter, $\sigma_r = 0.15$ (a) Original image (b) Groundtruth (c) Proposed method with 20 samples



(a) (b)

Fig. 6: Processing time of (a) box spatial kernel and (b) Gaussian spatial kernel using 20 samples.

4.1. Numerical accuracy evaluation

As discussed in section 3, smaller number of samples is favored to reduce the overall computation. To compare with [7], the number of samples is set to a cubic number. We show the result of 3^3 and 4^3 color samples in Fig. 3. The brute force results are used as groundtruth to measure PSNR. For each parameter setting, we plot the maximum, mean and minimum PSNR among all test images.

As one can see, our method is able to produce superior results that guarantee PSNR over 35 dB, while [7] fails to handle lower sample counts. Note that [7] takes eight samples per pixel for interpolation while our method takes only four samples. Another advantage of our interpolation method is that the number of samples is not limited to the cubic number any more. We can take fewer samples to lower overall processing time. In general, 20 principle colors are sufficient to produce decent quality as shown in Fig. 4 and Fig. 5. Visual comparison using GsGr kernel is demonstrated in Fig. 5.

4.2. Runtime evaluation

We evaluate the processing time on a 2.1 GHz Intel Core i7-4600U CPU. The box filter and the Gaussian filter takes 7.8 ms and 15.7 ms to process 1M pixels. Let the number of samples be N, then there will be $4N$ set of filtering, N for W^k and $3N$ for J^k (3 channels). The overall processing time can be understood by $T_{sampling}+4N\times T_{filtering}+T_{interpolation}$.

$T_{sampling}$ and $T_{interpolation}$ both increase with N. Using 20 samples to process 1M pixels, it takes 4 ms and 112 ms, respectively. Fig. 6 compares the processing time with brute force computation with varying filter radius. It takes 1.3 sec and 2.3 sec in total for box spatial and Gaussian spatial kernels.

5. CONCLUSION

In this paper, we generalize the PBFIC method as a three-stage algorithm and propose an extension for color images. Based on the observation that image colors are sparsely distributed in color space. We propose an adaptive sampling strategy which is able to achieve high accuracy while lowers the overall computation due to fewer samples needed compared to [7], yielding a fast bilateral filtering technique for color images.

Acknowledgements. This work is partially supported by Himax Technologies, Inc. and Ministry of Science and Technology of Taiwan, under Grant MOST 104-3115-E-002-002.

6. REFERENCES

[1] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *IEEE International Conference on Computer Vision*, Jan 1998, pp. 839–846.

[2] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama, "Digital photography with flash and no-flash image pairs," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 664–672, Aug. 2004.

[3] T. Q. Pham and L. J. van Vliet, "Separable bilateral filtering for fast video preprocessing," in *IEEE International Conference on Multimedia and Expo*, July 2005, pp. 4 pp.–.

[4] F. Porikli, "Constant time o(1) bilateral filtering," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2008, pp. 1–8.

[5] Q. Yang, K.-H Tan, and N. Ahuja, "Real-time o(1) bilateral filtering," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 557–564.

[6] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," *International Journal of Computer Vision*, vol. 81, no. 1, pp. 24–52, Jan. 2009.

[7] Q. Yang, N. Ahuja, and K.-H Tan, "Constant time median and bilateral filtering," *International Journal of Computer Vision*, vol. 112, no. 3, pp. 307–318, 2014.

[8] B. Weiss, "Fast median and bilateral filtering," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 519–526, July 2006.

[9] A. Adams, N. Gelfand, J. Dolson, and M. Levoy, "Gaussian kd-trees for fast high-dimensional filtering," *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3, pp. 21:1–21:12, July 2009.

[10] A. Adams, J. Baek, and M. A. Davis, "Fast high-dimensional filtering using the permutohedral lattice," *Computer Graphics Forum*, vol. 29, no. 2, pp. 753–762, 2010.

[11] A. Buades, B. Coll, and J. M. Morel, "A non-local algorithm for image denoising," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2005, vol. 2, pp. 60–65 vol. 2.

[12] I. Omer and M. Werman, "Color lines: image specific color representation," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2004, vol. 2, pp. 946–953.

[13] M. McCool and E. Fiume, "Hierarchical poisson disk sampling distributions," in *Proceedings of the Conference on Graphics Interface*, San Francisco, CA, USA, 1992, pp. 94–105, Morgan Kaufmann Publishers Inc.

[14] R. Bridson, "Fast poisson disk sampling in arbitrary dimensions," in *ACM SIGGRAPH 2007 Sketches*, New York, NY, USA.

[15] P. Viola and M. J. Jones, "Robust real-time face detection," *IEEE International Conference on Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[16] L. J. van Vliet, I. T. Young, and P. W. Verbeek, "Recursive gaussian derivative filters," in *IEEE International Conference on Pattern Recognition*, Aug 1998, vol. 1, pp. 509–514.