Optimal Fitting Polynomial For Linear Time Bilateral Filter

Longquan Dai, Mengke Yuan, Xiaopeng Zhang

The bilateral filter (BF) has demonstrated great effectiveness for a variety of problems. However, its brute-force implementation is time-consuming. One way of accelerating BF is to approximate the nonlinear range kernel of BF by a set of linear time shiftable kernels. To reach this goal, we notice that only finite values of the kernel of BF are used to perform smoothing due to the quantization of digital images. Thus, it does not change the filtering results by substituting the range kernel with the function having the same values at finite discrete points. We also learn that the lagrange interpolation polynomial can exactly pass through predefined points and therefore employ it to replace the original kernels for accurate accelerating BF. In addition, for speeding up BF at the cost of small approximation error, two approximation methods are proposed to obtain the optimal fitting polynomial. The performance of our method is validated in extensive experiments.

Introduction: The bilateral filter (BF) [1] is a valuable tool for many computer vision and graphics applications and its output at the pixel location $\boldsymbol{x} = (x, y)$ is given by

$$\hat{I}_{\boldsymbol{x}} = \frac{\sum_{\boldsymbol{y} \in \mathcal{N}_{\boldsymbol{x}}^r} K_s(\|\boldsymbol{y} - \boldsymbol{x}\|) K_r(I_{\boldsymbol{y}} - I_{\boldsymbol{x}}) I_{\boldsymbol{y}}}{\sum_{\boldsymbol{y} \in \mathcal{N}_{\boldsymbol{x}}^r} K_s(\|\boldsymbol{y} - \boldsymbol{x}\|) K_r(I_{\boldsymbol{y}} - I_{\boldsymbol{x}})} \tag{1}$$

where $K_s(\cdot)$ is the spatial domain kernel, $K_r(\cdot)$ is the range kernel and $\mathcal{N}_{\boldsymbol{x}}^r$ is the square window centered at $\boldsymbol{x}$ with radius r. For BF, the kernels $K_s(\cdot)$ and $K_r(\cdot)$ determine how the spatial and intensity differences are treated. The most commonly used kernel in literature is the Gaussian kernel $G_\sigma(z) = \exp(-\frac{z^2}{2\sigma^2})$. Other unusual kernels are reported in [10]. Due to the signal-dependant range kernel $K_r(I_{\boldsymbol{y}} - I_{\boldsymbol{x}})$, BF becomes an edge-aware filter. On the flip side of the power, the complexity of the brute-force implementation is $O(|\mathcal{N}^r||I|)$ that is too intensive for time-critical applications, where $|I|$ is the number of pixels in the entire image and $|\mathcal{N}^r|$ is the number of pixels in the window $\mathcal{N}_{\boldsymbol{x}}^r$ centered at $\boldsymbol{x}$. The inefficiency stems from that the signal-dependant range kernel $K_r(I_{\boldsymbol{y}} - I_{\boldsymbol{x}})$ introduces nonlinearity to BF and therefore rules out lots of fast calculating techniques. One solution is to approximate the nonlinear range kernel by a set of linear time shiftable kernels [2], where the linear time means the computational complexity is $O(|I|)$ which is independent of the window radius r. In literature, the existing accelerating methods of BF roughly fail into following two categories:

- ***Series Expansion Based BF***: This kind of BFs employ the Taylor series or linear time shiftable kernels [2] to represent the Gaussian range kernel. Chaudhury *et al.* [4] reduce the Gaussian kernel to the raised cosine range kernel. Dai *et al.* [3] propose Hermite polynomials to decompose the Gaussian kernel. Our method also belongs to this kind of methods.
- ***Histogram Based BF*** Porikli [5] found that BF can be computed in linear time by the fast integral histogram [6] if $K_s(\|\boldsymbol{y} - \boldsymbol{x}\|)$ is the box spatial kernel $B_{\mathcal{N}_{\boldsymbol{x}}^r}(\|\boldsymbol{y} - \boldsymbol{x}\|)$. Zhang *et al.* [7] use the de Moivre-Laplace theorem to decompose the Gaussian kernel into M_s box functions. Gunturk [8] employs the least squares optimization to find the optimal coefficients β_i^s that minimize the approximation error $(K_s(\|\boldsymbol{y} - \boldsymbol{x}\|) - \sum_{i=1}^{M_s} \beta_i^s B_{\mathcal{N}_{\boldsymbol{x}}^{r_i}}(\|\boldsymbol{y} - \boldsymbol{x}\|))^2$. Pan *et al.* [9] formulate their objective function from the sparsity perspective to solve the optimal coefficients β_i^s.

The Optimal Fitting Polynomial: Since the intensity values of digital images are quantized to N_r discrete values $r_j, 1 \le j \le N_r$, only finite discrete points $K_r(r_j)$ of BF's range kernel are used to compute the filtering results. We employ the least squares fitting algorithm to find the optimal fitting polynomial that approximates to the range kernel $K_r(\cdot)$ with the smallest approximation error at the discrete points $(r_j, K_r(r_j))$ and replace the range kernel $K_r(\cdot)$ in Eq. (1) by the optimal polynomial. In this way, the nonlinear convolution of $K_r(\cdot)$ can be decomposed into a set of box filters. More importantly, we can compute the box filter in linear time by the fast histogram technique [6].

In order to obtain a versatile fitting polynomial $P_r(\cdot)$, it is reasonable to expect that the optimal $P_r(z) = \sum_{i=0}^{M_r} \beta_i^r z^i$ can exactly pass through some predefined points in the set $\{(r_j, K_r(r_j))\}$ while minimizing the approximation error for the remaining points. Hence, we divide $\{(r_j, K_r(r_j))\}$ into two subsets, where the first L points denote the points being used to compute the approximation error and the last $N_r - L$ points represent the predefined constrain points. Considering the two subsets, we employ the constrained quadratic optimization (2) to compute the coefficients β_i^r of the optimal fitting polynomial $P_r(\cdot)$.

$$\begin{aligned} &\min_{\beta_i^r} \sum_{j=1}^{L} (K_r(r_j) - \sum_{i=0}^{M_r} \beta_i^r (r_j)^i)^2 \\ &\qquad \sum_{i=0}^{M_r} \beta_i^r (r_j)^i \ge 0, \quad 1 \le j \le L \\ &s.t. \quad \sum_{i=0}^{M_r} \beta_i^r (r_j)^i = K_r(r_j), \quad L+1 \le j \le N_r \end{aligned} \tag{2}$$

The constrains $\sum_{i=0}^{M_r} \beta_i^r (r_j)^i \ge 0$ are added to prevent negative fitting curve complained by Chaudhury [4]. Moreover, the optimization can be efficiently solved by the matlab function 'lsqlin'.

Case 1: $L = 0$. The optimal polynomial $P_r(\cdot)$ passes through all N_r points in the set $\{(r_j, K_r(r_j))\}$ if the degree M_r of the polynomial $P_r(\cdot)$ is no less than $N_r - 1$. Actually, employing the lagrange interpolation polynomial which is capable of exactly passing through all N_r points in the set $\{(r_j, K_r(r_j))\}$, we can directly write down the closed form solution of the optimization (2) as

$$P_r(z) = \sum_{i=1}^{N_r} K_r(r_i) \prod_{j=1, j \ne i}^{N_r} \frac{z - r_j}{r_i - r_j} \tag{3}$$

Although substituting the lagrange interpolation polynomial $P_r(\cdot)$ with $K_r(\cdot)$ does not change the filtering result, adopting it to speed up BF is not very efficient because the high degree polynomial increases the computation complexity. For instance, $r_j \in \{-255, \ldots, 255\}$ for 8-bits image. In this situation, the degree of $P_r(\cdot)$ is up to 510 which implies 510+ times linear filtering for accelerating BF. One possible solution is to relax the exact equality constrains.

Case 2: $L = N_r$**.** The optimal polynomial does not prefer any special points, and the constrained optimization (2) is free to decide the optimal polynomial $P_r(\cdot)$ with arbitrary degree. Although the produced $P_r(r_j)$ is no longer equal to $K_r(r_j)$, the approximation error is usually very small. We can safely replace $K_r(r_j)$ by $P_r(r_j)$ without introducing significant numeric error. More importantly, the degree of $P_r(\cdot)$ will be very low compared with the lagrange polynomial. However, when $L = N_r$, Eq. (2) is not appropriate to compute the optimal fitting curve for nonsmoothing range kernel $K_r(\cdot)$.

Case 3: $0 < L < N_r$**.** The optimal polynomial $P_r(\cdot)$ exactly passes through the last $N_r - L$ predefined points in the set $\{(r_j, K_r(r_j))\}$ with the smallest approximation error for the first L points in the set $\{(r_j, K_r(r_j))\}$, if the degree $M_r \ge N_r - L - 1$. Note that the optimal polynomial $P_r(\cdot)$ is a compromise for the first two cases. In the first case, $P_r(\cdot)$ produces accurate values at the cost of high degree (*i.e.* high computational burden). Although the optimal $P_r(\cdot)$ in the second case achieves the lowest degree, its value is not accurate. The third case makes a compromise between the degree (efficiency) and the number of the accurate values (accuracy).

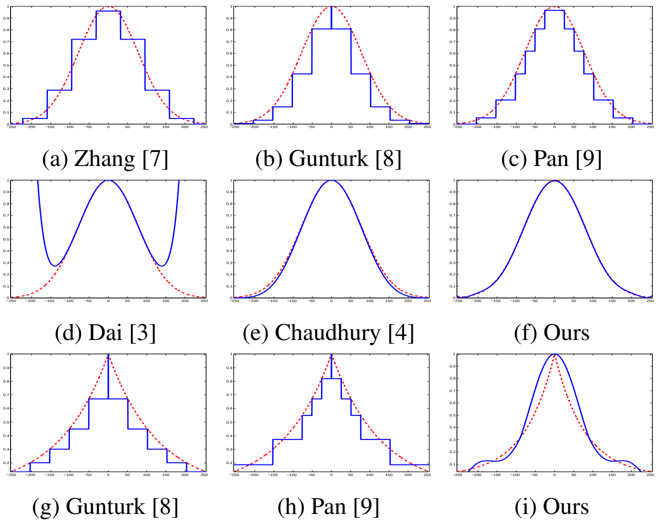
Frankly, all three cases have pros and cons. Users can choose different solutions according to different requirements. Here, we provide an unified framework that is able to trade off the efficiency and accuracy for different situations.

Our Accelerating method: We employ the optimal polynomial $P_r(z) = \sum_{i=0}^{M_r} \beta_i^r z^i$ with degree M_r to approximate arbitrary range kernel $K_r(\cdot)$ as $P_r(z)$ is able to linearize the nonlinear convolution of $K_r(\cdot)$. Then, substituting the nonlinear convolution of $K_r(\cdot)$ with $P_r(z)$, we have

$$\hat{I}_{\boldsymbol{x}} = \frac{\sum_{i=0}^{M_r} \sum_{j=0}^{i} \varphi_{ij}(I_{\boldsymbol{x}}) \sum_{\boldsymbol{y} \in \mathcal{N}_{\boldsymbol{x}}^r} K_s(\|\boldsymbol{y} - \boldsymbol{x}\|) F_{\boldsymbol{y}}^{ij}}{\sum_{i=0}^{M_r} \sum_{j=0}^{i} \varphi_{ij}(I_{\boldsymbol{x}}) \sum_{\boldsymbol{y} \in \mathcal{N}_{\boldsymbol{x}}^r} K_s(\|\boldsymbol{y} - \boldsymbol{x}\|)} \tag{4}$$

where $\varphi_{ij}(I_{\boldsymbol{x}}) = \beta_i^r \binom{i}{j} (-I_{\boldsymbol{x}})^{i-j}$, $\psi_{ij}(I_{\boldsymbol{y}}) = I_{\boldsymbol{y}}^j$ and $F_{\boldsymbol{y}}^{ij} = \psi_{ij}(I_{\boldsymbol{y}}) I_{\boldsymbol{y}}$. Once again, the most time-consuming operation in Eq. (4) is the linear convolution of $K_s(\cdot)$ because $\sum_{i=0}^{M_r} \sum_{j=0}^{i}$ is pointwise sum operation.

In literature, the Fast Fourier transform is exploited to fast compute the linear convolution. We can also employ Pan's multiple boxes approximation strategy [9] to compute the filtering results. According to Pan's method, we decompose the spatial kernel $K_s(\|\boldsymbol{y} - \boldsymbol{x}\|)$ into



(a) Zhang [7] (b) Gunturk [8] (c) Pan [9]

(d) Dai [3] (e) Chaudhury [4] (f) Ours

(g) Gunturk [8] (h) Pan [9] (i) Ours

Fig. 1: The Gaussian kernel approximations (in the first and second rows) and the exponential decay kernel $\exp(-\frac{|x|}{\sigma})$ approximations (in the third row), where the variance $\sigma = 80$.

M_s box functions $K_s(\|\boldsymbol{y}-\boldsymbol{x}\|) \approx \sum_{i=1}^{M_s} \beta_i^s B_{\mathcal{N}_{\boldsymbol{x}}^{r_i}}(\|\boldsymbol{y}-\boldsymbol{x}\|)$, where $B_{\mathcal{N}_{\boldsymbol{x}}^{r}}(\|\boldsymbol{y}-\boldsymbol{x}\|) = 1$ when $\boldsymbol{y} \in \mathcal{N}_{\boldsymbol{x}}^{r}$, otherwise $B_{\mathcal{N}_{\boldsymbol{x}}^{r}}(\|\boldsymbol{y}-\boldsymbol{x}\|) = 0$. Then replacing the nonlinear convolution by $\sum_{i=1}^{M_s} \beta_i^s B_{\mathcal{N}_{\boldsymbol{x}}^{r_i}}(\|\boldsymbol{y}-\boldsymbol{x}\|)$, we can obtain the fast equivalent filtering form (5).

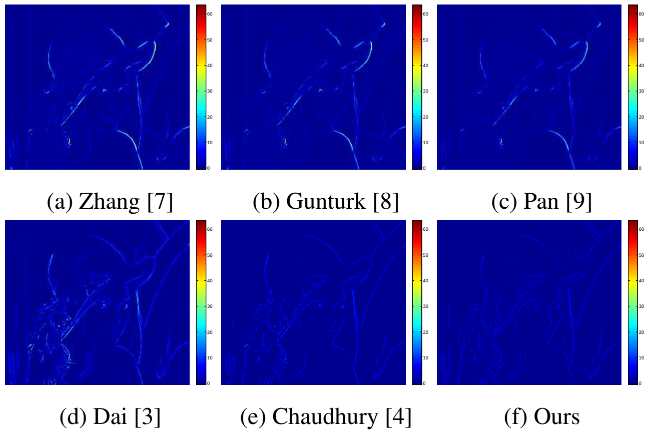
$$\hat{I}_{\boldsymbol{x}} = \frac{\sum_{i=0}^{M_r}\sum_{j=0}^{i}\varphi_{ij}(I_{\boldsymbol{x}})\sum_{k=1}^{M_s}\beta_k^s\mathcal{B}_{F^{ij}}^{\mathcal{N}_{\boldsymbol{x}}}}{\sum_{i=0}^{M_r}\sum_{j=0}^{i}\varphi_{ij}(I_{\boldsymbol{x}})\sum_{k=1}^{M_s}\beta_k^s\mathcal{B}_{E}^{\mathcal{N}_{\boldsymbol{x}}}} \qquad (5)$$

The integral histogram technique can compute the box filtering results $\mathcal{B}_{F^{ij}}^{\mathcal{N}_{\boldsymbol{x}}}$ and $\mathcal{B}_{E}^{\mathcal{N}_{\boldsymbol{x}}}$ of F^{ij} and the matrix E of ones. Moreover, the sum operations $\sum_{i=0}^{M_r}\sum_{j=0}^{i}, \sum_{k=1}^{M_s}$ are pointwise. Hence the overall computational complexity of Eq. (5) is $O(|I|)$.

Kernel Approximation: BF accelerating methods usually decompose BF into a set of linear time filters for fast computation. They do this by substituting the original kernel in BF with its approximation. Thereby the accuracy of the approximation kernel determines the accuracy of final results. In literature, Zhang [7], Gunturk [8] and Pan [9] all resort to a set of box functions to approximate the Gaussian kernel. The major difference is that the three methods take different techniques to determine the radiuses r_i of box functions and the coefficients β_i^s. However, due to the nonsmooth property, the box functions can only obtain rough approximations for the Gaussian kernel as illustrated in Fig 1. In contrast, Dai [3] and Chaudhury [4] take advantage of the Hermite polynomials and raised cosines, respectively, to approximate the Gaussian kernel. Nevertheless, the success of series expansion methods depends on the smoothness of the target kernel. For arbitrary nonsmooth kernels, there is no guarantee for the existence of series expansion. Even worse, the approximation for the Gaussian kernel with small variance is not satisfactory. The approximation errors of both two series expansion methods are significant using a low order expansion, when the variance of the Gaussian kernel is small. Moreover, the approximation using the box function is also not very well for the small variance. Since there are no special points on the curve of the smooth Gaussian kernel, we choose the second case in the The Optimal Fitting Polynomial section to compute the optimal polynomial of the Gaussian kernel and illustrate the optimal polynomial with degree 6 for the 511 discrete points $\{(r_i, G_\sigma(r_i))\}, r_i \in \{-255, \ldots, 255\}$ in Fig. 2.

The exponential decay kernel $\exp(-\frac{|x|}{\sigma})$ is not differentiable at zero. The methods of Zhang [7], Dai [3] and Chaudhury [4] can not deal with this situation. Although the box function can be used to approximate the kernel in the frameworks proposed by Gunturk [8] and Pan [9], the approximation curve is still very rough. In contrast, employing the third case in the The Optimal Fitting Polynomial section, our method is able to produce an approximation polynomial exactly passing through the nondifferentiable point as illustrated in Fig. 2.

Accuracy Evaluation: In this section, we directly compute the linear convolution of K_s and evaluate the PSNR index which reflects the accuracy of different nonlinear convolution decomposition methods to verify the approximation ability of our method. This is because BF accelerating methods adopt various techniques to decompose the nonlinear convolution of the range kernel into a set of the linear convolutions and employ linear time algorithms to fast compute the linear convolutions. If we take the brute-force method to compute linear convolution, the approximation error can only be caused by the nonlinear convolution decomposition method. In order to provide an intuitive illustration, we visualize the approximation error of Lena image in Fig 2. Both the spatial kernel and the range kernel of the prototype BF are chosen to be the Gaussian kernel $G_\sigma(\cdot)$. The results are obtained by the naive implementation of the accelerating methods Zhang [7], Gunturk [8], Pan [9], Dai [3], Chaudhury [4] and ours without using the brute-force linear convolution. Let I^{bf} denote the result of the naive implementation of BF and I^{ac} represent the result of accelerating methods, we compute the absolute error by $|I^{bf} - I^{ac}|$ and illustrate the filtering results and corresponding absolute error in Fig 2. It is easy to find that our method achieves the smallest absolute error among the six comparative accelerating methods.



(a) Zhang [7] (b) Gunturk [8] (c) Pan [9]

(d) Dai [3] (e) Chaudhury [4] (f) Ours

Fig. 2: Approximation error illustration. The spatial kernel and the range kernel of the prototype BF are chosen as the Gaussian kernel and we assign $\sigma_r = 80$ for the Gaussian range kernel.

Conclusion: The key of accelerating BF is to decompose the nonlinear convolution of BF into a set of linear convolutions. We do this by substituting the range kernel with the optimal fitting polynomial and then employing the shiftable property to decompose the nonlinear convolution into a set of linear convolutions. The major advantages of our method are that it is able to approach both differential and nondifferential range kernels and our accelerating algorithm only needs addition and multiplication operations to calculate filtering results.

Longquan Dai, Mengke Yuan and Xiaopeng Zhang (*NLPR, Institute of Automation Chinese Academy of Sciences, Beijing, China*)

E-mail: xiaopeng.zhang@ia.ac.cn

References

1 C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, 1998.

2 K.N. Chaudhury, Constant-Time Filtering Using Shiftable Kernels. *IEEE Signal Processing Letters*, 2011

3 Longquan Dai, Mengke Yuan and Xiaopeng Zhang, Accelerate bilateral filter using Hermite polynomials. *Electronics Letters*, 2014.

4 K.N. Chaudhury, D. Sage, and M. Unser. Fast o(1) bilateral filtering using trigonometric range kernels. *TIP*, 2011.

5 F. Porikli. Constant time o(1) bilateral filtering. In *CVPR*, pages 1–8, June 2008.

6 F. Porikli. Integral histogram: a fast way to extract histograms in cartesian spaces. In *CVPR*, volume 1, pages 829–836 vol. 1, June 2005.

7 K. Zhang, G. Lafruit, R. Lauwereins, and L. Van Gool. Constant time joint bilateral filtering using joint integral histograms. *TIP*, 21(9):4309–4314, Sept 2012.

8 B. Gunturk. Fast bilateral filter with arbitrary range and domain kernels. *TIP*, 20(9):2690–2696, Sept 2011.

9 X. A. Shengdong Pan and H. He. Optimal o(1) bilateral filter with arbitrary spatial and range kernels using sparse approximation. *MPE*, 2014.

10 F. Durand and J. Dorsey. Fast bilateral filtering for the display of high-dynamic-range images. *TOG*, 21(3):257–266, July 2002.