Proceedings of the 4th International Conference on Computing and Data Science (CONF-CDS 2022)
DOI: 10.54254/2755-2721/2/20220569

Hardware oriented bilateral filter algorithm and architecture survey

Yao Gong[1]

[1]University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China

mrgongyao@outlook.com

Abstract. Images and videos are often interfered with and affected by various noises. To filter out noise, researchers have proposed many filtering algorithms. bilateral filtering is a nonlinear filtering technique that can retain detailed information such as image edges well while denoising. However, it is a difficult task to implement fast and high-precision bilateral filtering algorithms in hardware platforms. In this paper, for the optimization of bilateral filtering algorithms, we analyze the gradient bilateral filtering and the piecewise approximate bilateral filtering algorithm that can be implemented in hardware, respectively. Then, we analyze and compare the implementation and optimization of bilateral filtering algorithms on FPGA and VLSI hardware architectures, and finally, make suggestions for choosing the appropriate bilateral filtering algorithms and hardware architectures for different purposes.

Keywords: Bilateral filter, Optimal algorithm, Hardware design, FPGA, VLSI.

1. Introduction

With the rapid development of communication technology, images and videos have become the most commonly used information carriers in human activities. However, in the process of image and video acquisition and transmission, it is often interfered with and affected by various noises. For example, during the digitization process of image acquisition, the image sensor itself and the external environment will generate noise. And there will also be noise in the transmission channel during the image transmission process. Therefore, to obtain high-quality digital images, we need to perform noise reduction processing while maintaining the integrity of the original information as much as possible.

Commonly used filtering algorithms include mean filtering, median filtering, and Gaussian filtering [1]. However, the general filtering algorithm often blurs the edge of the image while removing noise. Therefore, researchers proposed a bilateral filtering algorithm based on the principle of Gaussian filtering [2]. While smoothing the noise, it solves the problem of blurred image edges caused by Gaussian filtering, which helps to preserve the edge information of the image and make it become one of the most popular filtering methods.

Most images and videos require real-time processing, especially for different hardware devices. Hence, although the initial bilateral filter performance is acceptable for image noise processing, it consumes much time to be implemented on hardware systems, which challenges a lot for real-time application. Therefore, to accelerate the speed of bilateral filter methods for real-time image denoising, numerous hardware-friendly methods are proposed recently.

DOI: 10.54254/2755-2721/2/20220569

In [3], the authors proposed an FPGA hardware design for implementing the bilateral filtering algorithm. It was based on the fast-exponential operation of lookup tables and the division method of shift-subtraction to achieve the acceleration of the bilateral filter. The design improved the processing performance and achieved the real-time performance of the image processing system. In [4], referring to the improved scheme of the bilateral filtering algorithm in [5], the author optimized the speed of bilateral filtering by adding a filtering function based on pixel similarity based on FPGA implementation technology. In the experiments, the computational speed test works well, and the denoising visual effect is also optimized. In [6], the authors developed a compact architecture of bilateral filters for piecewise approximate computation of distance weights on an FPGA platform by combining approximate computation and lookup tables. The work achieved comparable filtering effects and denoising performance in terms of PSNR and SSIM. In [7], the authors proposed a low-cost VLSI bilateral filtering hardware architecture. The required multipliers and storage space are reduced by applying distance-oriented grouping and resource sharing. Experimental results show that this architecture is more cost-effective while maintaining acceptable denoising performance, which is significant for industrial mass production and large-scale use.

It can be seen that these hardware-friendly bilateral filter real-time solutions can be categorized as improved algorithms and highly efficient hardware architecture designs. This paper will comprehensively analyze and summarize the typical bilateral filtering hardware technologies described above, to provide a reference guide to choosing the appropriate bilateral filtering algorithms and hardware structure designs for different applications.

The rest of the paper is organized as follows: Section II introduces the principle of bilateral filtering algorithm, and analyzes two typical bilateral filtering algorithms based on hardware design. Section III describes the hardware architecture of two efficient bilateral filters, and analyzes the implementation of the bilateral filter algorithm on its architecture. Section IV concludes the paper and gives an outlook on the hardware implementation of bilateral filtering.

2. Hardware friendly bilateral filter method

2.1. The principle of bilateral filtering

The bilateral filter is based on the principle of nonlinear filtering. On the one hand, it is similar to the traditional Gaussian filtering algorithm, using the idea of local weighted averaging. On the other hand, it considers both the spatial proximity of pixel points and the similarity of grayscale values between pixel points, which are called spatial proximity factor and grayscale similarity factor, respectively. By the nonlinear combination of the two kernels, the noise is effectively filtered and the image edges are well preserved. The gray value of each pixel after bilateral filtering is equal to the weighted average of its neighboring pixels, and the weighting factor of the neighboring pixels is equal to the product of the spatial proximity factor and the grayscale similarity factor. This ensures that only neighboring pixels with close spatial distance and little difference in gray value have a greater impact on the filtering result of the central pixel. Suppose I_p is the filtered image, the whole filtering process is described as,

$$I_p(x,y) = \frac{\sum_{(i,j)\in M_{x,y}} w_s(i,j) w_r(i,j) I(x,y)}{\sum_{(i,j)\in M_{x,y}} w_s(i,j) w_r(i,j)} \quad (1)$$

$$w_s(i,j) = e^{-\frac{|i-x|^2+|i-y|^2}{2\sigma_s^2}} \quad (2)$$

$$w_r(i,j) = e^{-\frac{|I(i,j)-I(x,y)|^2}{2\sigma_s^2}} \quad (3)$$

where $M_{x,y}$ denotes the set of $(2\mathrm{N}+1)\times(2\mathrm{N}+1)$ spatial neighboring pixels centered on (x,y). $I(x,y)$ denotes the centroid pixel value of $M_{x,y}$, $w_s(i,j)$ is the spatial proximity factor, $w_r(i,j)$ is the grayscale similarity factor, and σ_s and σ_r are the filtering parameters. Then the expression of bilateral filtering is as follows.

Assuming that w is the weight factor, then,

DOI: 10.54254/2755-2721/2/20220569

$$w(i,j) = \frac{w_s(i,j) \times w_r(i,j)}{c} \quad (4)$$

where c is a constant, equation (4) shows that the weight factor is controlled by both the spatial proximity factor and the grayscale similarity factor.

In a region of gentle image change, all pixels in a certain neighborhood have very similar gray values, and the bilateral filter can be regarded as a low-pass filter. The differences of those pixels that are less correlated are eliminated by averaging. While in a region of sharp image change, the gray value similarity factor tends to 1 for pixels on the same side of the edge and 0 for pixels on the opposite side of the edge. This property of the grayscale similarity factor enables the bilateral filters to protect the image edges from blurring while denoising.

2.2. The improved hardware-friendly gradient based bilateral filtering algorithm

Instead of the traditional distance-based spatial filters kernel, a novel gradient bilateral filtering is implemented based on bilateral filtering with the introduction of local patterns of images [5]. Gradient bilateral filtering uses the gradient distance of neighboring pixel luminance values to construct the gradient similarity kernel. The geometric proximity kernel function and the gradient similarity kernel function are used to weight the pixels in the image neighborhood to achieve filtering. Gradient Bilateral Filtering is defined by the following equation,

$$u_{GBF}(x) = \frac{1}{C_{d,g}} \sum_{y \in \Omega} w_d(x,y) w_g(x,y) u(y) \quad (5)$$

$$w_d(x,y) = e^{-\frac{|x-y|^2}{2\sigma_d^2}} \quad (6)$$

$$w_g(x,y) = e^{-\frac{(d_x^g)^2}{2\sigma_g^2}} \quad (7)$$

$$d_x^g = \sqrt{\left|\frac{\partial u(x)}{\partial x} - \frac{\partial u(y)}{\partial x}\right|^2 + \left|\frac{\partial u(x)}{\partial y} - \frac{\partial u(y)}{\partial y}\right|^2} \quad (8)$$

$$C_{d,g} = \sum_{y \in \Omega} w_d(x,y) w_g(x,y) \quad (9)$$

where the weight function $w_d(x,y)$ measures the geometric proximity of the neighborhood centroid x to its neighbor y. Its value is not related to the content of the image, but only to the geometric position. And the weight function $w_g(x,y)$ measures the gradient similarity between the neighborhood centroid x and the neighbor y, i.e., the similarity of the gradient values of the two pixel-points. d_x^g is the gradient distance between the neighborhood centroid x and the neighbor y. $C_{d,g}$ is the normalization factor. σ_d is the spatial distance weight coefficient parameter and σ_g is the image gradient weight coefficient parameter, respectively. The gradient bilateral filtering weight function is composed of the geometric proximity function $w_d(x,y)$ and the pixel gradient similarity function $w_g(x,y)$. The two are multiplied to obtain the gradient bilateral filtering weight kernel function.

To verify the denoising effect of gradient bilateral filtering, one typical grayscale image is used for the analysis and discussion of experimental results. The results show that the ability of the gradient bilateral filtering to suppress noise is better than that of the conventional bilateral filtering to suppress noise. Fig.1(c) and Fig.1(d) show the results of noise removal by conventional bilateral filtering and gradient bilateral filtering for images with Gaussian white noise with a noise variance of 50.

Also, this gradient bilateral filtering algorithm can be implemented on a hardware platform. For example, the authors used FPGA-based implementation techniques with parallelism and flow [4], which can significantly speed up the algorithm.

DOI: 10.54254/2755-2721/2/20220569

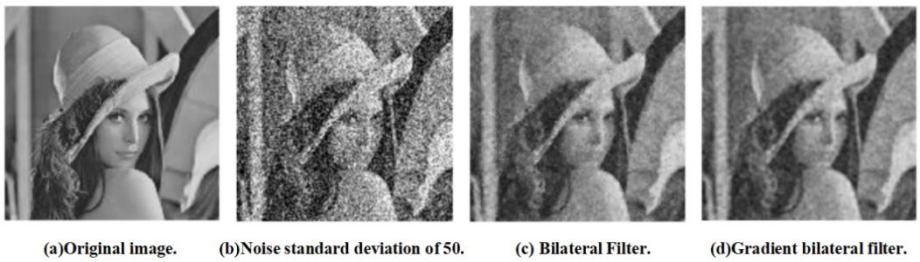

Figure 1. After adding Gaussian noise with standard deviation of 50 in (a), the filtering algorithm denoising result [5]

Although the method above can improve the filtering performance of traditional bilateral filtering, the processing is much more complex, which induces harder hardware real-time design with more hardware resources. Therefore, when studying the bilateral filtering algorithm implemented on the hardware platform, not only should the algorithm be optimized, but also its adaptability and implementation difficulty on the hardware platform should be considered.

2.3. The Hardware-friendly Bilateral Filter Based on Piecewise Approximate Computing

The bilateral filter consists of spatial kernel and range kernel. Image edges are protected by the range kernel of the bilateral filter. But this also results in a huge computational complexity, and the weights cannot be simply precomputed and stored. This limits the utilization of bilateral filtering, especially in application scenarios with harsher noise. So, [6] proposed a piecewise approximate bilateral filtering algorithm that can be implemented in hardware, using the least square root method to segment the computed weights of the minimal value range. The typical process of the algorithm is shown in Fig.2.

In Fig.2, taking a 5×5 filter mask as an example, the weights of the piecewise approximate bilateral filter are calculated. The specific process is roughly divided into five steps. First, the spatial kernel weights are calculated. Second, using the least square root method, the best fitting point for the range kernel weights is calculated. Third, the spatial kernels are respectively multiplied by the fitted range kernels. Next, the computed weights are normalized to the final filter weights. Finally, the matrix calculation is performed to calculate the product of pixels and weights to reduce noise.

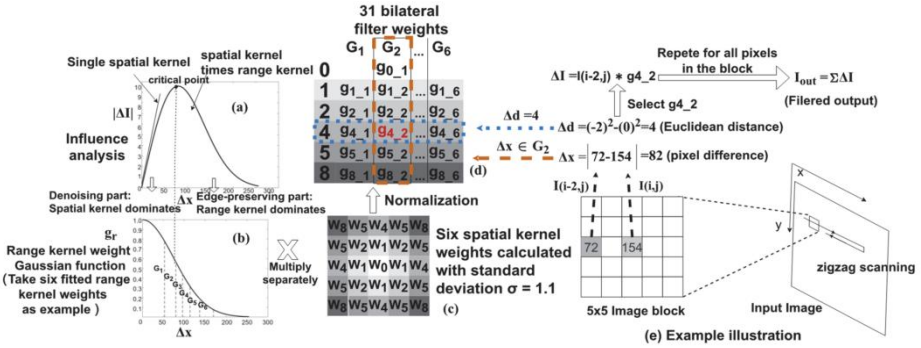

Figure 2. Flow framework and example description of piecewise approximate bilateral filtering algorithm [6].

The denoising performance and filtering effect of the piecewise approximate bilateral filtering algorithm are comparable to or even better than most bilateral filtering algorithms, as shown in Fig.3. What's more, the algorithm is very hardware friendly, which can save hardware cost and resources. As proved, this algorithm is implemented on FPGA with a detailed hardware architecture design and achieves high acceleration results as described later.

DOI: 10.54254/2755-2721/2/20220569

Figure 3. Denoising results of MATLAB and bilateral filter hardware implementation [6].

Although the denoising and edge-holding performance of this algorithm are acceptable, the fitting and normalization process cannot be global precisely optimal. What's more, it does not consider the adaptive filtering strength for different parameters, which means the filtering performance cannot be adaptive according to the image contents. Thus, it is suggested more parameters adaptive and precise bilateral filtering fast methods are deserved to study.

3. Hardware architecture design for bilateral filter

Bilateral filtering requires a large amount of computation, which greatly increases the computation time for processing high-resolution images. Usually, the bilateral filtering algorithm is implemented on serial-structure processors such as CPU, ARM, or DSP, which is hard for real-time computation. Therefore, hardware implementation design is inevitable for real-time applications. With the improvement of chip process and technology, FPGA and VLSI (Very Large-Scale Integration) have significant advantages in power consumption, performance and cost. Hence, in this section we will introduce an FPGA-based and a VLSI-based bilateral filtering hardware structure, respectively.

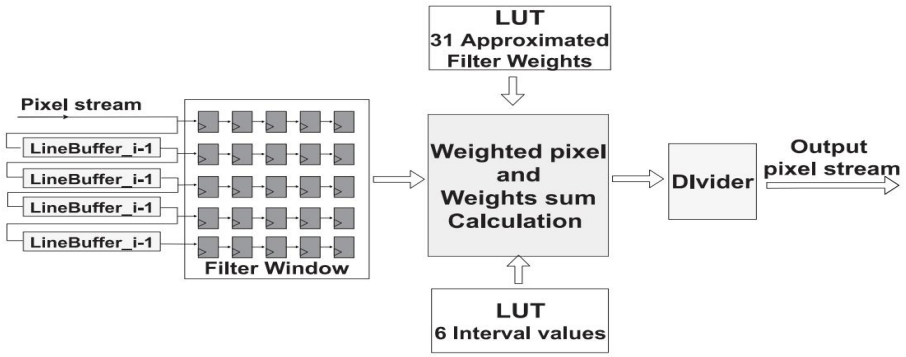

Figure 4. The hardware architecture for the piecewise approximate bilateral filtering [6].

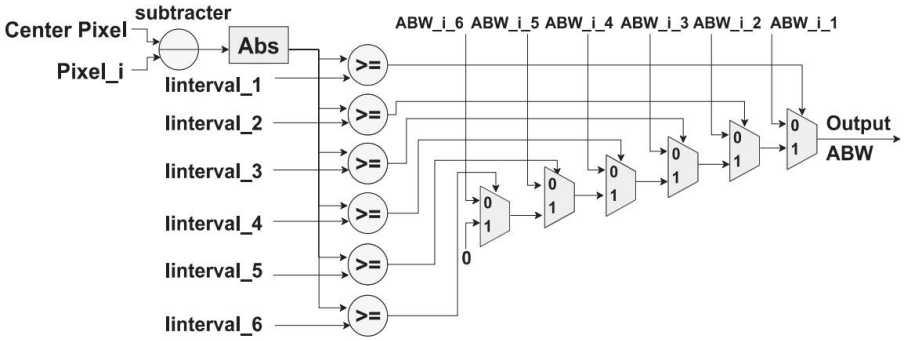

Figure 5. Pixel weight selection hardware structure [6].

DOI: 10.54254/2755-2721/2/20220569

3.1. A compact hardware architecture based on FPGA

As introduced in [6], the authors developed a bilateral filter compact architecture on the FPGA for the piecewise approximate bilateral filtering described above. Fig.4 shows the total hardware architecture, which includes three modules: filter mask window module, computation module, and normalization module. It can be seen that the input pixel stream is transferred through the line buffer to the mask module with the pixel-level pipeline.

The specific hardware architecture of the above modules is shown in Fig.5, Fig.6 and Fig.7. To accelerate the divider process, a parallel pixel-level pipeline architecture is employed in this architecture with the LUT-based design for bilateral filtering. Firstly, the pre-calculated weights of the bilateral filter are stored in LUTs, which will be employed in the weight and sum calculation module for higher calculation speed. Secondly, the critical path is optimized by converting division to multiplication. Bilateral filtering computations are greatly reduced through LUT-based dividers, pixel-level pipeline structures, and parallelizable weighted computations.

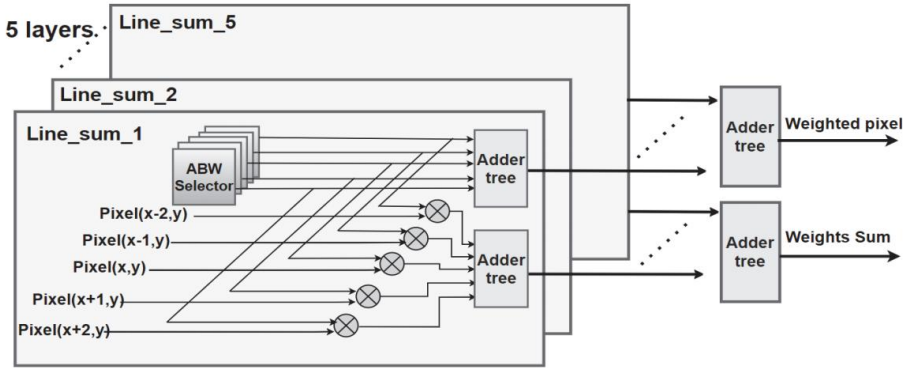

Figure 6. Pixel weighting and weight summation calculation module [6].

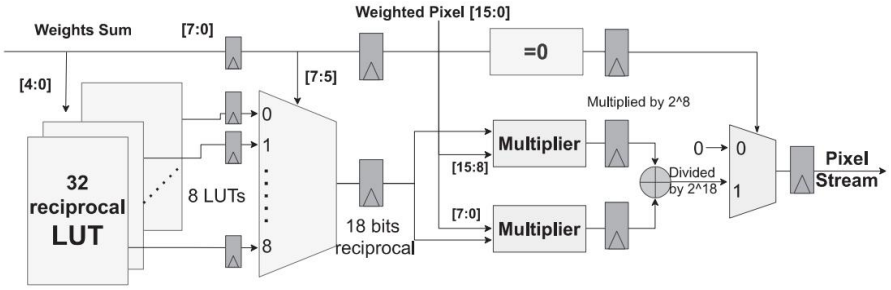

Figure 7. The divider architecture based on LUT [6].

In experiments, the present architecture achieved comparable filtering effects and denoising performance in terms of PSNR and SSIM. Compared with the work in [8] and [9], the compact architecture for approximate computational bilateral filtering not only greatly reduces the hardware resource consumption and increases the processing speed but also achieves comparable visual denoising results.

The main feature of the compact architecture is the reduced utilization of hardware resources. This hardware architecture only requires 2391 configurable LUTs, 1812 registers, 144Kbit SRAM, and only two dedicated computing blocks (DSP). The maximum frequency is 278MHz and the power dissipation is about 168mW. In addition, the parallel architecture speeds up the computation of processing weights and pixels.

This bilateral filtering hardware architecture greatly reduces the use of hardware resources based on algorithm optimization. Also, the parallel and flowing nature of the FPGA makes it very advantageous in the implementation of bilateral filtering algorithms. However, the FPGA-based

DOI: 10.54254/2755-2721/2/20220569

architecture and solution cannot be applied for VLSI due to different platform characteristics between FPGA and VLSI, while VLSI is widely used in more compact integrated circuit designs for the much smaller area. For comparison, we will analyze the VLSI platform below.

3.2. A VLSI architecture of the bilateral filter

It is necessary to realize the real-time application of bilateral filtering algorithms by hardware. The authors of [10] proposed a kernel-based design where they use position-oriented grouping. A total of 23 multipliers are needed to deal with a 5×5 filtering window in one pixel clock cycle. In contrast, [7] proposed a low-cost and real-time hardware architecture based on VLSI for bilateral filters. The architecture is optimized using methods such as size reduction of LUT, resource sharing and distance-oriented grouping. Fig.8 shows the block diagram description of the proposed VLSI architecture. More details can be found in [7].

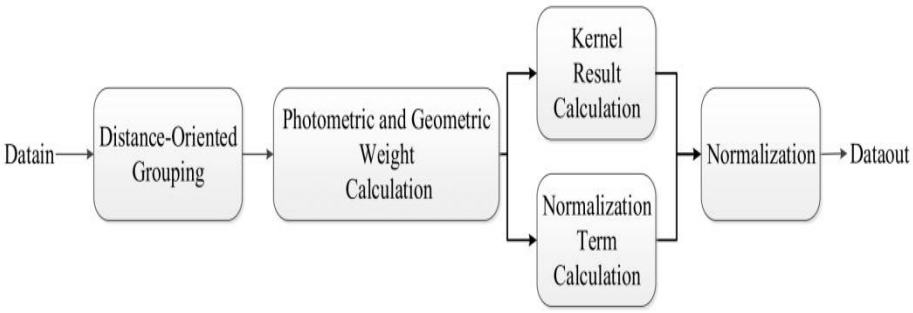

Figure 8. Block diagram of proposed VLSI architecture [7].

The authors of [7] grouped the input data in the filter window based on the Euclidean distance to the center pixel, assigning pixels that have the same Euclidean distance to the center pixel to a group for better hardware parallel computation. Also, the authors of [7] have simplified the processing steps by removing some redundant hardware resources. Since the specific coefficient weights for each pixel in the filter window of the algorithm have the same constituents, it is possible to reorganize the order in which the bilateral filtering is computed.

Therefore, to avoid the heavy computations required for exponential operations, we usually employ a LUT approach that records all precomputed weight values to obtain exponential results. The intensity difference can be regarded as the input value of the LUT, so that the corresponding luminosity component can be directly searched when used. Since the intensity difference can vary from 0 to 255, a 256-entry LUT is required to store all values as employed in [7].

Furtherly, observation of possible photometric component values shows that some are near zero. These values can slightly affect the results of the bilateral filter and therefore they can be deleted to reduce the resource occupied by the LUT. Based on a large number of experiments, the authors concluded that twelve groups of values are sufficient to obtain comparable bilateral filtering performance. Therefore, each LUT takes 12 entries in the actual design. Compared to [10] which needs 256 entries in each LUT, the design of [7] can extremely reduce the size of the LUT by 95%. This enables resource sharing and reduces the hardware cost. The number of adders and selectors required for the computation is also greatly reduced compared to [10].

DOI: 10.54254/2755-2721/2/20220569

Table 1. Hardware results of different filtering window sizes [7].

Window size	Number of Adder	Number of Multipliers	Clock cycles/Output pixel	Max.clock frequency (MHz)	Pipeline stage	Throughput (pixel/s)
3×3	4	4	4	236.697	32	59171146.1
5×5	12	12	4	236.697	33	59171132.0
7×7	24	24	4	236.697	34	59171117.9
9×9	40	40	4	236.697	35	59171103.8

Table1 enumerates the hardware resource usage, speed, and throughput of bilateral filtering with different size filtering windows. It can be seen that the hardware resource usage increases with the window size, while the throughput and clock frequency remain almost constant, which demonstrates the robustness of the hardware design. Besides these, experiment results show that the bilateral filtering algorithm based on the VLSI design works well, which means it achieves a suitable compromise between hardware cost and filtering effect.

With analysis, we can find the VLSI-based hardware architecture proposed in [7] reduces the required multiplier and storage space. While maintaining the same image quality, frame rate, and operating clock frequency, the architecture proposed in [7] is more cost-efficient than [10], which is significant for industrial mass production, but there is still much room for filtering quality improvement in this architecture for high-precision image processing.

4. Conclusion

This paper summarizes the hardware implementation of typical bilateral filtering algorithms. On one hand, the original bilateral filtering algorithm, and two optimized bilateral filtering algorithms that can be implemented in hardware are analyzed. On the other hand, two typical bilateral filtering hardware designs for FPGA and VLSI are analyzed, respectively. With the detailed introduction and discussion, we can find different bilateral filter methods that could be employed and selected for different purposes with filtering performance, speed, precision, and hardware platform considerations. With the development of image processing technology, the need for the high performance and efficient hardware implementation of the bilateral filtering algorithms is still a hot topic and challenge, which deserves to pay more attention to studying.

References

[1] Babaud J, Witkin A P, Baudin M, et al. Uniqueness of the Gaussian kernel for scale-space filtering[J]. IEEE transactions on pattern analysis and machine intelligence, (1): 26-33 (1986).

[2] Tomasi C, Manduchi R. Bilateral filtering for gray and color images[C]//Sixth international conference on computer vision (IEEE Cat. No. 98CH36271). IEEE: 839-846(1998).

[3] Zhang Aijie, Liu Shijian, Zhang Rui, et al. Design of bilateral filter based on FPGA[J]. Infrared Technology, 41(1): 13-17(2019).

[4] Huang Jiye, Lu Yanyi. FPGA-based bilateral filtering algorithm [J]. laboratory research and Exploration, 38(8):48-51(2019).

[5] Jiang Hui, Wang Hui, Zhang Jiashu. Image denoising by gradient bilateral filtering[J]. Computer Engineering and Applications, 52(5): 231-235(2016).

[6] Yao R, Chen L, Dong P, et al. A Compact Hardware Architecture for Bilateral Filter with the Combination of Approximate Computing and Look-up Table[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, (2022).

[7] Dabhade S D, Rathna G N, Chaudhury K N. A reconfigurable and scalable FPGA architecture for bilateral filtering. IEEE Transactions on Industrial Electronics, 65(2): 1459-1469(2017).

DOI: 10.54254/2755-2721/2/20220569

[8] Gabiger-Rose A, Kube M, Weigel R, et al. An FPGA-based fully synchronized design of a bilateral filter for real-time image denoising. IEEE Transactions on Industrial Electronics, 61(8): 4093-4104(2013).
[9] Lien C Y, Tang C H, Chen P Y, et al. A Low-Cost VLSI Architecture of the Bilateral Filter for Real-Time Image Denoising. IEEE Access, 8: 64278-64283(2020).
[10] Charoensak C, Sattar F. FPGA design of a real-time implementation of dynamic range compression for improving television picture[C]//2007 6th international conference on information, communications & signal processing. IEEE: 1-5(2007).