IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 12 July 2024, accepted 12 August 2024, date of publication 15 August 2024, date of current version 28 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3443999

RESEARCH ARTICLE

An FPGA-Based Hardware Architecture of Gaussian-Adaptive Bilateral Filter for Real-Time Image Denoising

AILIN XIE, AO ZHANG, AND GUOHUI MEI
College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
Corresponding author: Guohui Mei (meiguohui@ise.neu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61873051.

ABSTRACT The bilateral filter exhibits remarkable efficacy in noise suppression and edge preservation. This article proposes a hardware architecture based on field-programmable gate array (FPGA) for a modified bilateral filter. To enhance the efficacy of the bilateral filter, the Gaussian-adaptive Bilateral Filter (GABF) is employed as a modified filtering method. Approximating the filter weights using look-up tables (LUTs) results in reduced storage requirements and eliminates the need for complex exponential weight calculations. Moreover, the GABF is markedly accelerated by the highly parallel functional modules and LUTs. In addition to the aforementioned features, the GABF is implemented as a parallel architecture, which results in a reduction in hardware resource utilization compared to previous works. The results of the image quality analysis demonstrate that this article can achieve superior image quality compared with state-of-the-art works. The implementation results indicate that the proposed architecture is capable of performing real-time denoising at a frame rate of 95.65 fps for a 640 × 480 video with a power dissipation of 93.22 mW.

INDEX TERMS Bilateral filter, edge-preserving, field-programmable gate array (FPGA), image denoising.

I. INTRODUCTION

Digital images can be corrupted during the process of acquisition, transmission and reception by various noises [1], [2]. Irrespective of the complexity of the realistic noise, the noise corruption process can be defined as the sum of a noise-free image I and the additive white Gaussian noise (AWGN) N: $I_{noise} = I + N,\ N \in \Omega\ (0, \sigma^2)$. Consequently, image denoising represents an essential constituent of the image processing system, given that image noise has the potential to impair the efficacy of other algorithms employed within the system. Among the previous references, it was demonstrated that the main challenges for image denoising were edge preservation, image restoration, computaional intensity and the blurring effect problem [3]. Compared with other denoising algorithms, the bilateral filter has a great advantage in image denoising owing to its ability to remove noise without causing structural information loss [4]. The bilateral filter comprises two parts. The nonlinear part, also known as the photometric filter or the range filter, mainly contributes to the edge-preserving property. The linear part, also called the spatial filter or the geometric filter, aims at averaging the pixels which are selected by the range filter. By virtue of the aforementioned properties, the bilateral filter has been applied in various areas, such as medical imagery [5], detail enhancement [6], depth reconstruction [7], tone mapping [8], image fusion [9], etc. Nonetheless, due to the nonlinear nature of the bilateral filter, the filter weights cannot be readily pre-calculated or stored in look-up tables (LUTs) in the manner of linear filters. This also leads to an increase in computational complexity and latency.

In order to enhance the performance and efficacy of the bilateral filter, a number of researchers have proposed modifications to the existing bilateral filtering algorithms. In [10], Gavaskar and Chaudhury proposed a fast adaptive bilateral filter where the size of the range kernel varied at different pixels, while the algorithm complexity didn't scale with the kernel size. To enhance the performance of color

The associate editor coordinating the review of this manuscript and approving it for publication was Ilaria De Munari.

images, Mozerov et al. put forth a rapid bilateral filtering algorithm founded upon color sparseness and local statistics. This approach offers a swift and precise bilateral filter approximation with promising applications in real-time video denoising [11]. In [12], Caraffa et al. developed a guided bilateral filter based on graduated nonconvexity, which can reduce the non-Gaussian noise in the image.

Recently, in order to enhance the filtering performance, Chen et al. proposed a Gaussian-adaptive bilateral filter (GABF) where a low-pass guidance was added as the input of the range kernel [13]. Building upon the original bilateral filter, Nogami et al. proposed the decomposed multilateral filtering (DMF) algorithm, which decomposes the filter into a set of constant-time filters. This approach markedly reduces computational time and enhances performance efficiency [14]. In [15], Yang et al. put forth a bilateral-regularized optimization model that exhibits an enhanced capacity for edge preservation. The experimental results demonstrate the versatility of the proposed filter in a range of image processing applications. In light of the algorithmic complexity and filtering performance, the GABF proposed in [13] is selected for implementation on the FPGA platform with the objective of enhancing the performance of noise suppression and edge-preserving.

Conventionally, bilateral filters were implemented on central processing units (CPUs) or graphics processing units (GPUs). However, their low throughput or energy efficiency has resulted in a bottleneck for their applications [16]. Recently, bilateral filters have been implemented on hardware platforms such as ASIC and FPGA to improve speed and efficiency. In comparison to alternative platforms, FPGA can provide higher energy efficiency, higher performance and better reconfigurability, which makes it more suitable for the implementation of bilateral filters [17].

Prior to the implementation of an image denoising algorithm on a hardware platform, modifications must be made to the existing software-based algorithm. Similarly, the computational complexity of these algorithms is considerable, resulting in high-level power dissipation and substantial hardware resource consumption. As the software-based algorithms presented in [10], [11], [12], [13], [14], and [15] are not suitable for real-time applications on hardware systems such as FPGAs, the necessity arises for the implementation of dedicated hardware algorithms, as exemplified by [18] and [19].

Given the prevalence of the bilateral filter, a great deal of effort has been made into the hardware acceleration for practical use. Generally speaking, the hardware acceleration methods of the bilateral filter can be summarized into two categories. One category is aimed at simplifying the calculating process, such as modifying the weights calculation or the parallelization of the algorithm. The other one focuses on the pre-calculation of the weights and replacing the traditional computing process with the utilization of look-up tables (LUTs).

In terms of the first method, the approximation and parallelization of the filter kernels can significantly reduce the calculation complexity. Chaudhury and Dabhade [18] proposed a fast constant-time bilateral filter algorithm using the Gaussian-polynomial to approximate the range kernel and implemented the hardware design of this algorithm on FPGA [20], the experimental results showed that larger kernels can be realized without increasing resource utilization. Gabiger-Rose et al. [19] implemented separate pipelines on FPGA by sorting the pixels into equal groups and increasing the internal clock frequency according to the pixel groups. The hardware architecture demonstrated a constant delay regardless of the filter kernel size, eliminating the necessity for an additional data buffer. In [21], Wen et al. proposed a high-throughput hardware architecture for the bilateral filter, which reduces computational complexity by approximation calculations and enhances throughput for high-resolution image processing through a data prefetch strategy. In [22], Fanny Spagnolo et al. put forth a novel approximation strategy with the objective of enhancing energy efficiency while preserving real-time filtering performance and elaboration accuracy. The experimental results demonstrate that the 5×5 filter based on this method consumes 0.92 nJ per pixel, representing a 2.8-fold improvement in energy efficiency compared to other similar works.

As for the second method, storing the filter weights in LUTs can realize the pre-calculation process on hardware and greatly reduce calculation complexity. Nevertheless, storage and accuracy can not be optimized at the same time because less storage using means low accuracy. Spagnolo et al. [23] used piecewise functions to approximate the coefficients of both kernels to 7-bit unsigned integers and stored them into size-reduced LUTs, which achieved 926.8 fps for the 5×5 kernel size with the maximum frequency of 244 MHz. In [24], Yao et al. designed a low-cost hardware architecture of the bilateral filter with the LUT-based divider and parallelized architecture, which achieved 30fps processing speed for eight-million-pixel video.

The GABF algorithm, as presented in reference [13], was originally developed on the MATLAB platform. However, due to the high computational complexity of this algorithm, it is not suitable for real-time image denoising. This article presents the implementation of the GABF algorithm on the FPGA platform, with the objective of accelerating the filtering process. The main contribution of this article is listed as follows.

(1) A novel kernel approximation method is proposed, which can significantly reduce the number of the range filter weights and the spatial filter weights of the bilateral filter by the piecewise approximation method.

(2) In order to improve the real-time processing speed and reduce the hardware resource consumption, a novel FPGA design architecture is proposed. In this architecture, separate pipelines are designed by sorting the pixels into equal groups. Besides, the internal clock frequency is quadrupled according

to the pixel groups. Furthermore, the architecture doesn't need any external data buffer.

(3) The pixels and weights sum calculation process is greatly accelerated by replacing the traditional calculation process with the LUTs which store the pre-calculated weights.

Due to the aforementioned contribution, the computational complexity and the hardware resources consumption of the entire bilateral filter are significantly reduced by the LUT-based parallelized weight calculation and separated pipeline architecture. The hardware implementation result shows that the proposed architecture can reach real-time 95.65 fps processing speed for 640 × 480 video on a low-cost FPGA. Furthermore, the results of the performance analysis demonstrate that the FPGA-based GABF exhibits a near-identical level of image quality in comparison to the MATLAB-based GABF. Additionally, it has been observed that the FPGA-based GABF even outperforms several state-of-the-art methods described in [19], [23], and [25].

II. GABF WITH PIECEWISE APPROXIMATION METHOD FOR FILTER WEIGHTS

A. BILATERAL FILTER

Bilateral filter is a highly efficient method for denoising digital images while preserving the edges and detail information based on geometric distances and intensity differences between adjacent pixels. Bilateral filter consists of two components: the geometric kernel and the photometric kernel. The geometric kernel, which denoises images by averaging the adjacent pixels in the spatial domain, is on the basis of the traditional Gaussian function:

$$G_s(m,n,i,j) = \exp(-\frac{SD(m,n,i,j)^2}{2\sigma_s^2}) \tag{1}$$

where (m, n) is the coordinate of the original pixel within the image to be processed, (i, j) is the coordinate of the pixels adjacent to the original pixel, $SD(m,n,i,j) = \sqrt{(m-i)^2+(n-j)^2}$ represents the spatial distance between the pixel $(i.j)$ and the pixel (m, n), σ_s stands for the standard deviation of the Gaussian function.

The photometric kernel represents the nonlinear component of the bilateral filter and plays an essential part in preserving the edges in the image. The photometric weights are calculated on the basis of the intensity difference between the original pixel and its adjacent pixel in the filter window, which indicates the extent of similarity between the two pixels:

$$G_r(m,n,i,j) = \exp(-\frac{\Delta I(m,n,i,j)^2}{2\sigma_r^2}) \tag{2}$$

where $I(m,n,i,j) = |f(m,n) - f(i,j)|$ represents the absolute intensity difference between the original pixel (m, n) and its adjacent pixel (i, j), σ_r stands for the standard deviation of the Gaussian function.

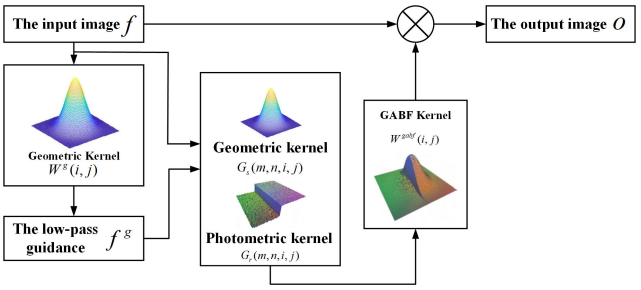


FIGURE 1. The entire architecture of the gaussian-adaptive bilateral filter.

The weights of the entire bilateral filter are given by the multiplication of both kernels:

$$W_{bf}(i,j) = G_s(m,n,i,j) \times G_r(m,n,i,j) \tag{3}$$

Finally, the output of the bilateral filter needs to be normalized otherwise the range of the output image would exceed the specified limits. The normalized output of the entire filter is given by (4):

$$O(m,n) = \frac{\sum_{(i,j)\in\Omega} f(i,j)W_{bf}(i,j)}{\sum_{(i,j)\in\Omega} W_{bf}(i,j)} \tag{4}$$

where $O(m, n)$ represents the output pixel of the bilateral filter; Ω is the filter window centred at the original pixel (m, n).

B. GAUSSIAN-ADAPTIVE BILATERAL FILTER

To improve the filtering performance, the GABF in [13] is adopted in this article. The main principle of the GABF is briefed as follows. Inside the bilateral filter, the photometric filter is designed to preserve the edges and the contours in the images. However, the photometric filter is considerably vulnerable to additive noise because the edge-preserving effect significantly degrades when the noise level of the image is relatively high. Therefore, a low-pass filter needs to be deployed in order to provide a guidance for the photometric filter, as illustrated in Fig. 1.

For a given image $\boldsymbol{f}$, in order to achieve the adaptive bilateral filter, the traditional Gaussian filter is applied to generate a low-pass guidance for the photometric filter. The Gaussian filtering process is achieved by averaging the adjacent pixels with weights based on the distances from the original pixel. The entire process can be described as:

$$f^g(m,n) = \frac{\sum_{(i,j)\in\Omega} W^g(i,j)f(i,j)}{\sum_{i,j\in\Omega} W^g(i,j)} \tag{5}$$

where the filter kernel is defined as:

$$W^g(i,j) = \exp(-\frac{SD(m,n,i,j)^2}{2\sigma_s^2}) \tag{6}$$

After the low-pass filtering, the output of the Gaussian filter can be used as the input of the aforementioned bilateral filter. Thus, the GABF kernel can be

1	5	8	5	1
5	23	38	23	5
8	38	63	38	8
5	23	38	23	5
1	5	8	5	1

(a)

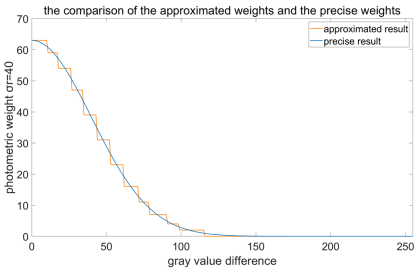
PSNR:28.0632
SSIM:0.7830

(b)

PSNR:28.1372
SSIM:0.7828

(c)

FIGURE 2. (a) The approximated 6-bit geometric kernel; (b) Pirate image filtered by the 6-bit kernel; (c) Pirate image filtered by the standard 8-bit kernel.



(a)

$$61 \le G_r(i,j) \le 63 \Rightarrow AG_r(i,j) = 63$$
$$57 \le G_r(i,j) < 61 \Rightarrow AG_r(i,j) = 59$$
$$51 \le G_r(i,j) < 57 \Rightarrow AG_r(i,j) = 54$$
$$43 \le G_r(i,j) < 51 \Rightarrow AG_r(i,j) = 47$$
$$35 \le G_r(i,j) < 43 \Rightarrow AG_r(i,j) = 39$$
$$27 \le G_r(i,j) < 35 \Rightarrow AG_r(i,j) = 31$$
$$19 \le G_r(i,j) < 27 \Rightarrow AG_r(i,j) = 23$$
$$13 \le G_r(i,j) < 19 \Rightarrow AG_r(i,j) = 16$$
$$9 \le G_r(i,j) < 13 \Rightarrow AG_r(i,j) = 11$$
$$5 \le G_r(i,j) < 9 \Rightarrow AG_r(i,j) = 7$$
$$3 \le G_r(i,j) < 5 \Rightarrow AG_r(i,j) = 4$$
$$1 \le G_r(i,j) < 3 \Rightarrow AG_r(i,j) = 2$$
$$G_r(i,j) < 1 \Rightarrow AG_r(i,j) = 0$$

(b)

FIGURE 3. (a) The comparison of the approximated weights and the precise weights, $\sigma_r = 40$; (b) The piecewise functions of the photometric weights approximation method.

described as:

$$W^{gabf}(i,j) = \exp\left(-\frac{SD(m,n,i,j)^2}{2\sigma_s^2}\right)\exp\left(-\frac{|f(m,n) - f^g(i,j)|^2}{2\sigma_r^2}\right) \quad (7)$$

where the low-pass guidance is obtained by (5). The combination of the geometric filter on $\boldsymbol{f}$ and the photometric filter on $\boldsymbol{f^g}$ can perform strong preservation of the edges and contours in the images, which significantly reduce the impact of various noise on the photometric filter of the bilateral filter. Therefore, the filtering output $O(m,n)$ of the GABF can be described as:

$$O(m,n) = \frac{\sum_{(i,j)\in\Omega} f(i,j) W^{gabf}(i,j)}{\sum_{(i,j)\in\Omega} W^{gabf}(i,j)} \quad (8)$$

The complete process of the GABF is illustrated in Fig. 1.

C. THE PIECEWISE APPROXIMATION METHOD

In order to save hardware resources, the filter weights arep-recalculated and stored in the LUTs. However, storing all the weights in the LUTs also consumes lots of hardware resources like registers. Therefore, in order to reduce the number of the filter weights without weakening the denoising effect, a piecewise approximation method for filter kernels is proposed. In this article, the standard deviation σ_s of the geometric kernel is set to 1.0. Considering the trade-off between smoothing effect and noise suppression, a filter mask of 5 × 5 is selected.

In the approximation of geometric kernel, to limit the bit-width of weights, all the geometric weights are approximated to 6-bit unsigned integers. As shown in Fig. 2(a), the geometric weights are approximated to integer values ranging from 1 to 63. Additionally, to test the denoising effect of the approximated kernel, the Pirate image with $\sigma_n = 18$ Gaussian white noise is filtered in MATLAB by the proposed kernel and the standard 8-bit kernel separately. The results of both kernels are presented in Fig. 2(b) and Fig. 2(c), where very little difference between these two kernels can be found in terms of peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) [26].

For the photometric kernel, the weights depend on the gray value difference which ranges from 0 to 255 under the situation of 8-bit pixels. Likewise, the photometric weights are scaled to 6-bit unsigned integers, which limits the highest weight value to 63. Besides, the parameter σ_r is set to $3\sigma_n$ according to the 3σ principle.

In Fig. 3(a), the precise weights curve shows that the gradient of the curve varies at different gray value differences. Therefore, in order to limit the number of the photometric weights, a piecewise approximation method is proposed. The main idea of this approach is deleting the weights whose corresponding gradients are close to zero. As shown in Fig. 3(b), the proposed piecewise functions are adopted to approximate the photometric weights. Furthermore, in Fig. 3(a), the yellow curve represents the approximated values of the photometric weights when σ_r is set to 40, which shows little difference from the precise curve.

III. THE NOVEL HARDWARE ARCHITECTURE

The entire hardware architecture of the FPGA-based GABF is illustrated in Fig. 4. The design concept of the proposed architecture is subdividing the filter into three functional modules, which are a register matrix module with line buffers, a pixels and weights sum calculation module and a normalization module. The register matrix module transforms the input pixels into 5 × 5 filter mask and separate the pixels into 6 groups, and each of them is connected to the calculation module respectively. Eventually, the calculation result is normalized to ensure the output is within the specified limits.

The module-based design concept can significantly reduce computing complexity and simplify the validation process [27]. Besides, by virtue of the design concept, the filter can be implemented as a highly parallel structure which can greatly reduce the hardware resources utilization.

A. REGISTER MATRIX MODULE

Due to the nonlinearity of the photometric filter, the photometric weights need to be calculated respectively for every pixel in the filter mask. As mentioned in Section II-C, the size of the filter mask is set to 5 × 5. In the current situation, the filtering of one pixel demands the calculation of 24 photometric weights, which results in high-level

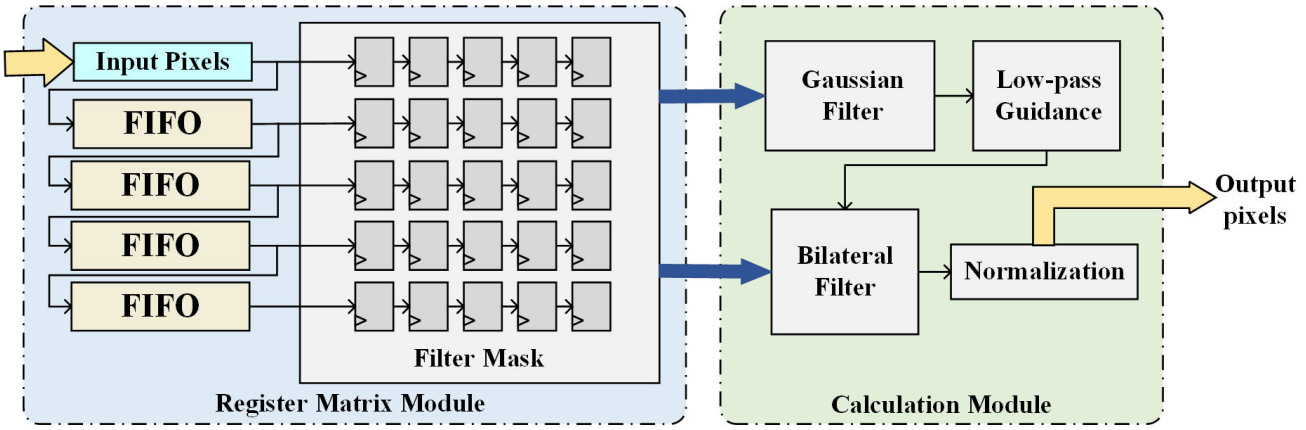


FIGURE 4. The entire hardware architecture of the proposed FPGA-based GABF.

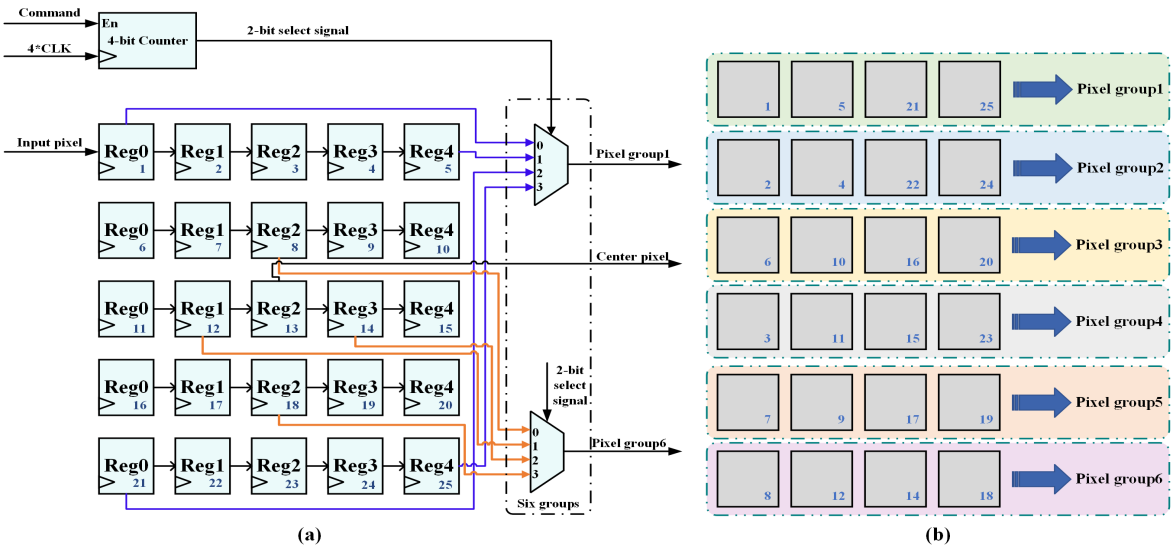


FIGURE 5. (a) Hardware architecture of the register matrix module. To improve readability, only the first and the last pixel group are presented. (b) The detailed pixels grouping method.

computing complexity. In addition, since the filter mask is shifted by one pixel at each clock along the pixel rows, extra storage for the adjacent pixels is required during the filtering process. In order to save hardware resources and reduce additional latency, four 8-bit FIFOs are implemented as data buffers so that five pixel rows can be stored.

The entire module is described in the left dashed box of Fig. 4. The bold black box represents the 5×5 filter mask. As the input pixel stream flows into the module by one pixel at a clock, the storage of block FIFOs and registers is shifted by one pixel, which generates a dynamic filter mask at each clock.

The parallel weights calculation and the normalization process at the final stage demand a great deal of hardware resources, which makes it difficult to be completed within just one pixel clock. In order to accelerate the filtering process, a design concept from [19] based on clock domain changing and pixels grouping is adopted and modified, which is illustrated in Fig. 5(a). To transfer the entire filter mask into next module simultaneously, all 25 pixels need to processed in one pixel clock in order to keep up with the input pixel stream. Furthermore, considering the symmetry of the geometric filter, all the pixels in the filter mask except the center pixel are sorted into six groups based on their Euclidean distances from the center pixel, while the center pixel doesn't belong to any group and is straightly transfered to the next module without any processing. The entire grouping method is depicted in Fig. 5(b). In view of the number of groups, the pixel clock is quadrupled so that all the pixels can be extracted from the module in just one pixel clock. As shown in Fig. 5, six 4-to-1 multiplexers are implemented to extract pixels from each group at every quadruple pixel clock. The counter in Fig. 5, which works under the quadruple pixel clock, produces a 2-bit signal to control the output of each pixel group. Besides,

the enable signal of the counter is triggered when the entire filter mask is full.

Since all the pixel groups are processed and pipelined to the next module simultaneously, the method of pixel grouping and clock domain change can significantly improve the synchronization and parallelism of the GABF.

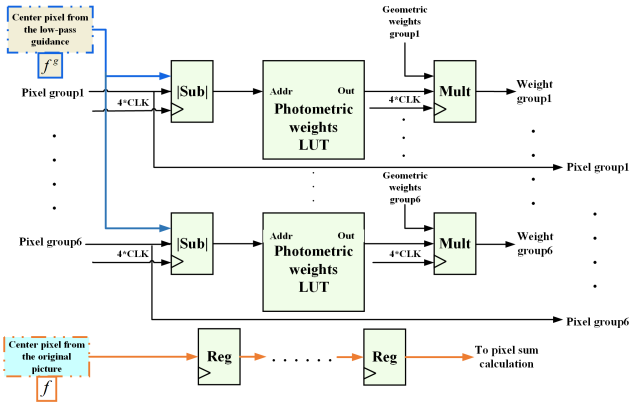


FIGURE 6. Hardware architecture of weights calculation.

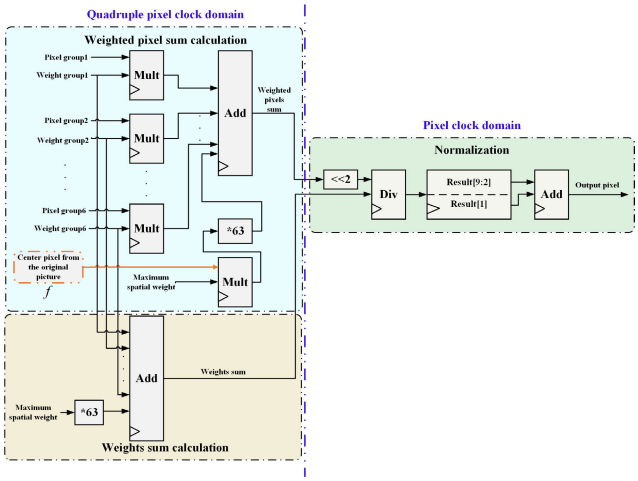


FIGURE 7. Hardware architecture of the output pixel calculation.

B. THE CALCULATION MODULE

The calculation module consists of two parts, which are the weights calculation part and the output pixel calculation part. The weights calculation part works under the quadruple pixel clock, and the corresponding detailed block diagram is illustrated in Fig. 6. After the pixels in the filter mask have been divided into six groups, each pixel group is connected to a calculation pipeline separately. First, the gray value of the pixel is subtracted from that of the center pixel from $\boldsymbol{f^g}$ in order to obtain the absolute value of the gray value difference between the original image and the low-pass guidance. To avoid the exponential calculation of the photometric weights, the approximated values are stored in the photometric weights LUT. The LUT utilizes the aforementioned absolute value as the address to select the corresponding weight for the pixel. The entire filter weight is then obtained by the product of the photometric weight and the corresponding geometric weight. Additionally, the original pixels and the filter weights are forwarded to the next stage for further calculation. To synchronize with the pixels and weights, the center pixel from the original image is also preserved and forwarded to the next stage through several registers for the weighted pixel sums calculation.

The output pixel calculation part contains three component, including the weighted pixels sum calculation, the weights sum calculation and the normalization, which are shown in Fig. 7. The six pixel groups are connected to six multipliers respectively in order to be multiplied by the corresponding weight group. As for the center pixel from $\boldsymbol{f}$, it is processed separately and multiplied by the highest values of photometric weights and geometric weights. The outputs of the seven multipliers are linked to an adder tree to calculate the weighted pixels sum. To calculate the weights sum, six weight groups together with the product of the maximums of both weights are connected to another adder tree. At the last stage, the output result needs to be normalized in case of exceeding the specified limits. Considering the fraction part, the weighted pixels sum is shifted 2 bits to the left before divided by the weights sum. Therefore, the first 8 bits of the result represent the integer part of the output gray value, while the ninth bit of the result stands for the fraction part. The sum of both parts is the normalized gray value of the output pixel. Additionally, the asynchronous clock domain synchronization is achieved through several registers, which are omitted in Fig. 7 for improving readability.

TABLE 1. The average PSNR and SSIM of the test images.

σ_n/σ_r	Noisy Image PSNR (dB) / SSIM	MATLAB Results PSNR (dB) / SSIM	ModelSim Results PSNR (dB) / SSIM
5/15	34.195/0.839	35.683/0.947	35.121/0.920
10/30	28.280/0.610	33.232/0.901	32.851/0.873
15/45	24.680/0.444	31.365/0.824	31.095/0.809
20/60	22.329/0.344	30.105/0.771	29.817/0.749
25/75	20.341/0.270	29.032/0.723	28.555/0.683
30/90	18.828/0.221	27.832/0.691	27.448/0.658

Test images: Cameraman, House, Pirate, Peppers, Boat, Lake[28][29].

TABLE 2. The average PSNR and SSIM comparison.

σ_n/σ_r	[19] PSNR / SSIM	[23] PSNR / SSIM	[25] PSNR / SSIM	Proposed PSNR / SSIM
10/30	31.00/0.85	32.74/0.86	31.786/0.831	32.851/0.873
20/60	27.00/0.71	N_A	28.295/0.724	29.817/0.749
30/90	24.70/0.59	26.76/0.65	26.497/0.643	27.448/0.658

N_A: Not available.

IV. EXPERIMENTAL RESULTS

A. PERFORMANCE ANALYSIS

In order to evaluate the efficacy of the proposed FPGA-based GABF in terms of noise reduction and edge preservation, the image results are assessed using the PSNR and SSIM metrics.

FIGURE 8. Comparison of the MATLAB results and the ModelSim results ($\sigma_n = 15$; $\sigma_r = 45$).

TABLE 3. Implementation results of the GABF.

	[19]	[23]	[25]	The Proposed Design
Filtering method	BF	BF	BF	GABF
Resolution	512×512	512×512	1024×1024	640×480
Platform	Xilinx Virtex-5	XC7Z020	Xilinx Virtex XC5VLX50-1	Altera Cyclone IV
Kernel Size	5×5	5×5	5×5	5×5
Registers	N_A	2635	N_A	2940
SRAM (kbit)	N_A	0	N_A	196
DSPs	29	32	36	8
Max. Freq. (MHz)	220	244	236.697	240
Max. Framerate (fps)	52.45	926.8	56.43	95.65

N_A: Not available.

The test images are processed using MATLAB-based GABF algorithm for comparison with the ModelSim simulation. The objective of the comparison is to analyze the impact of FPGA implementation on image quality. In order to simulate a range of noise levels, the grayscale test images are corrupted by additive white Gaussian noise with varying standard deviations.

As introduced in Section II-C, the standard deviation of geometric kernel σ_s is set to 1.0, while the standard deviation of the photometric kernel is set to $\sigma_r = 3\sigma_n$. Table 1 presents a comparison of the average PSNR and SSIM values between the MATLAB and ModelSim results. As illustrated in Table 1, the proposed architecture exhibits minimal discrepancies when compared to the MATLAB-based GABF. The comparison results of PSNR and SSIM demonstrate that the denoising and edge-preserving performance is comparable to that of the MATLAB-based GABF.

Moreover, the PSNR and SSIM results are compared with several recent works that have also been implemented on FPGAs. For purposes of comparison, the ModelSim results are evaluated using the same image set and standard deviation. As evidenced in Table 2, the proposed FPGA-based GABF is capable of achieving superior image quality compared to previous works.

Additionally, Fig. 8 describes the image comparison between the MATLAB results and the ModelSim results.

The σ_n and σ_r of the test examples are set to 15 and 45. As shown in this figure, the quality of the ModelSim results is nearly the same as that of the MATLAB results. Furthermore, judging from the image examples in the enlarged detailed information part, it is difficult to discern any notable discrepancies between the simulation outcomes of Modelsim and MATLAB.

B. THE HARDWARE IMPLEMENTATION RESULTS

To analyze the hardware resources utilization, the GABF is implemented on the Altera Cyclone IV EP4CE10 FPGA with OV5640 CMOS Image Sensor. The detailed implementation results are shown in Table 3. The design in this article consumes 2940 registers, 196Kbit SRAM and 8 DSPs. The piecewise approximation method, which has significantly reduced the number of the filter weights, only costs 114 bits registers compared with [19] that stored all of the weights by occupying 2048 bits registers. The pixel clock of the image sensor is set to 60 MHz, and the image format is set to 8-bit gray scale image.

Due to the quadruplication of the pixel clock, the maximal frequency reaches 240 MHz. With the resolution of 640 × 480, the maximal framerate can reach 95.65 fps. By virtue of the parallelized architecture and the approximation method, the power dissipation of this design is limited to 93.22 mW, which is lower than the previous works.

V. CONCLUSION

In this article, we proposed a highly synchronized hardware architecture of the Gaussian-adaptive bilateral filter (GABF) with the piecewise filter weights approximation method for real-time image denoising. The approximated GABF was implemented on a low-cost Altera Cyclone IV FPGA platform.

The LUT-based approximation method has significantly reduced the computational complexity. Moreover, the hardware implementation of the GABF demonstrates minimal discrepancies when compared to the MATLAB-based GABF in terms of noise reduction and edge preservation. Furthermore, the highly parallel modules have greatly accelerated the filtering process and reduced hardware resource consumption. Eventually, the proposed design in this article has achieved better performance in terms of noise reduction and edge preservation compared with several recent works. In regard to practical applications, our proposed design is suitable for a range of scenarios, including medical imagery, depth reconstruction, data fusion, detail enhancement, and 3D fairing.

REFERENCES

[1] S. K. Panigrahi and S. Gupta, "Joint bilateral filter for signal recovery from phase preserved curvelet coefficients for image denoising," *Int. J. Image Graph.*, vol. 21, no. 4, Oct. 2021, Art. no. 2150049.

[2] S. K. Panigrahi, S. K. Tripathy, A. Bhowmick, S. K. Satapathy, P. Barsocchi, and A. K. Bhoi, "Multi-scale based approach for denoising real-world noisy image using curvelet thresholding: Scope and beyond," *IEEE Access*, vol. 12, pp. 25090–25105, 2024, doi: 10.1109/ACCESS.2024.3364397.

[3] T. K. Abuya, R. M. Rimiru, and G. O. Okeyo, "An image denoising technique using wavelet-anisotropic Gaussian filter-based denoising convolutional neural network for CT images," *Appl. Sci.*, vol. 13, no. 21, p. 12069, Nov. 2023, doi: 10.3390/app132112069.

[4] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, Jan. 1998, pp. 839–846.

[5] W. C. K. Wong, A. C. S. Chung, and S. C. H. Yu, "Trilateral filtering for biomedical images," in *Proc. 2nd IEEE Int. Symp. Biomed. Imag., Macro Nano*, vol. 2, Jul. 2004, pp. 820–823.

[6] R. Fattal, M. Agrawala, and S. Rusinkiewicz, "Multiscale shape and detail enhancement from multi-light image collections," *ACM Trans. Graph.*, vol. 26, no. 99, p. 51, Jul. 2007.

[7] Q. Yang, R. Yang, J. Davis, and D. Nister, "Spatial-depth super resolution for range images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.

[8] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 257–266, Jul. 2002.

[9] J. Li, D. Han, X. Wang, P. Yi, L. Yan, and X. Li, "Multi-sensor medical-image fusion technique based on embedding bilateral filter in least squares and salient detection," *Sensors*, vol. 23, no. 7, p. 3490, Mar. 2023.

[10] R. G. Gavaskar and K. N. Chaudhury, "Fast adaptive bilateral filtering," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 779–790, Feb. 2019, doi: 10.1109/TIP.2018.2871597.

[11] M. G. Mozerov and J. van de Weijer, "Global color sparseness and a local statistics prior for fast bilateral filtering," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5842–5853, Dec. 2015, doi: 10.1109/TIP.2015.2492822.

[12] L. Caraffa, J.-P. Tarel, and P. Charbonnier, "The guided bilateral filter: When the joint/cross bilateral filter becomes robust," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1199–1208, Apr. 2015, doi: 10.1109/TIP.2015.2389617.

[13] B.-H. Chen, Y.-S. Tseng, and J.-L. Yin, "Gaussian-adaptive bilateral filter," *IEEE Signal Process. Lett.*, vol. 27, pp. 1670–1674, 2020.

[14] H. Nogami, Y. Kanetaka, Y. Naganawa, Y. Maeda, and N. Fukushima, "Decomposed multilateral filtering for accelerating filtering with multiple guidance images," *Sensors*, vol. 24, no. 2, p. 633, Jan. 2024, doi: 10.3390/s24020633.

[15] Y. Yang, Y. Sun, W. Gao, X. Wang, and L. Zeng, "Bilateral regularized optimization model for edge-preserving image smoothing," *Image Vis. Comput.*, vol. 146, Jun. 2024, Art. no. 105031, doi: 10.1016/j.imavis.2024.105031.

[16] S. Mittal, "A survey of FPGA-based accelerators for convolutional neural networks," *Neural Comput. Appl.*, vol. 32, no. 4, pp. 1109–1139, Feb. 2020, doi: 10.1007/s00521-018-3761-1.

[17] S. Mittal and J. S. Vetter, "A survey of methods for analyzing and improving GPU energy efficiency," *ACM Comput. Surv.*, vol. 47, no. 2, pp. 1–23, Jan. 2015, doi: 10.1145/2636342.

[18] K. N. Chaudhury and S. D. Dabhade, "Fast and provably accurate bilateral filtering," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2519–2528, Jun. 2016.

[19] A. Gabiger-Rose, M. Kube, R. Weigel, and R. Rose, "An FPGA-based fully synchronized design of a bilateral filter for real-time image denoising," *IEEE Trans. Ind. Electron.*, vol. 61, no. 8, pp. 4093–4104, Aug. 2014.

[20] S. D. Dabhade, G. N. Rathna, and K. N. Chaudhury, "A reconfigurable and scalable FPGA architecture for bilateral filtering," *IEEE Trans. Ind. Electron.*, vol. 65, no. 2, pp. 1459–1469, Feb. 2018.

[21] J. Wen, Y. Feng, and Z. Li, "A high-throughput hardware architecture for bilateral filter with configurable convolution and cost-effective MAC unit," *IEICE Electron. Exp.*, vol. 21, no. 13, 2024, Art. no. 20240276, doi: 10.1587/elex.21.20240276.

[22] F. Spagnolo, P. Corsonello, F. Frustaci, and S. Perri, "Approximate bilateral filters for real-time and low-energy imaging applications on FPGAs," *J. Supercomput.*, vol. 80, no. 11, pp. 15894–15916, Apr. 2024, doi: 10.1007/s11227-024-06084-y.

[23] F. Spagnolo, P. Corsonello, F. Frustaci, and S. Perri, "Design of approximate bilateral filters for image denoising on FPGAs," *IEEE Access*, vol. 11, pp. 1990–2000, 2023.

[24] R. Yao, L. Chen, P. Dong, Z. Chen, and F. An, "A compact hardware architecture for bilateral filter with the combination of approximate computing and look-up table," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 7, pp. 3324–3328, Jul. 2022.

[25] C.-Y. Lien, C.-H. Tang, P.-Y. Chen, Y.-T. Kuo, and Y.-L. Deng, "A low-cost VLSI architecture of the bilateral filter for real-time image denoising," *IEEE Access*, vol. 8, pp. 64278–64283, 2020.
[26] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[27] R.-X. Chen, L.-G. Chen, and L. Chen, "System design consideration for digital wheelchair controller," *IEEE Trans. Ind. Electron.*, vol. 47, no. 4, pp. 898–907, Aug. 2000.
[28] *USC-SIPI Image Daset*. Accessed: Jan. 3, 2023. [Online]. Available: https://sipi.usc.edu/database/database.php?volume=misc
[29] *Standard Test Images*. Accessed: Jun. 17, 2023. [Online]. Available: https://www.imageprocessingplace.com/downloads_V3/root_downloads/image_databases/standard_test_images.zip

AILIN XIE received the B.S. degree in automation from Dalian University of Technology, Dalian, China, in 2022. He is currently pursuing the M.S. degree in control science and engineering with Northeastern University, Shenyang, China.

His current research interests include infrared radiation temperature measuring theory and field-programmable-gate-array design.

AO ZHANG received the B.S. degree in measurement and control technology and instrumentation from Northeastern University, Shenyang, China, in 2021, where he is currently pursuing the Ph.D. degree in control science and engineering.

His current research interest includes infrared radiation temperature measuring theory.

GUOHUI MEI received the B.S. degree in mechanical manufacturing from Shenyang University of Technology, Shenyang, China, in 1995, the M.S. degree in mechanical design and theory from Dalian Polytechnic University, Dalian, China, in 1998, and the Ph.D. degree in mechanical design and theory from Northeastern University, Shenyang, in 2001.

He is currently a Professor with Northeastern University. His current research interests include infrared radiation temperature measuring theory and technology.