Design and Implementation of Adaptive Gaussian Denoising Algorithm Based on FPGA

Xinghao Chen
School of Electronic Information Engineering
Beihang University
Beijing, China
xhchen@buaa.edu.cn

2024 IEEE 7th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE) | 979-8-3503-7703-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/AUTEEE62881.2024.10869804

***Abstract*—With the continuous progress of image processing and machine vision technology, the demand for efficient and real-time processing is becoming more and more prominent, especially in the field of high-noise image processing. In this study, an adaptive Gaussian filtering algorithm is proposed, which is implemented based on FPGA and aims to improve the computational efficiency and real-time performance of the image processing system. Compared with the traditional fixed-weight filter, this algorithm is able to dynamically adjust the filtering parameters according to different noise environments, effectively balancing noise suppression and image detail retention. We coded the algorithm using Verilog hardware description language and verified it on PYNQ-Z2 FPGA platform. The experimental results show that the adaptive algorithm outperforms the fixed-weight filtering method in terms of performance, especially in terms of noise suppression and detail preservation. Meanwhile, the FPGA hardware implements the reduction of filtering delay and optimization of resource consumption, making it well suited for real-time applications. This study demonstrates the promise of FPGA adaptive filtering for applications in medical imaging, remote sensing, and intelligent surveillance, which have stringent requirements for high-performance and high-efficiency processing. This research provides new hardware solutions for real-time, high-quality image processing in constrained environments.**

Keywords—FPGA, image denoising, gaussian filtering, adaptive algorithm

I. INTRODUCTION

With the rapid development of machine vision and image processing technology, real-time processing and high-performance computing have become the key requirements in the field of image applications. Traditional processor architecture often struggles to meet real-time requirements with large-scale image data. FPGA (Field Programmable Gate Array) is becoming an ideal choice for image processing due to its low power consumption, high parallel computing capability, and low latency. It provides an efficient hardware acceleration platform and significantly improves the real-time performance of image denoising and enhancement tasks. In medical imaging, remote sensing monitoring and intelligent security, and other scenarios that require high image quality, the effect of image denoising directly affects the quality of subsequent analysis and processing.

The core of image denoising lies in suppressing noise while preserving image details. Traditional denoising algorithms, such as Gaussian filtering, bilateral filtering, etc., although to a certain extent suppressed the noise, are often difficult to consider the complete retention of image details in complex conditions. For example, Gonzalez and Woods' study highlights this trade between noise suppression and detail preservation [1], while Tomasi and Manduchi's bilateral filtering method achieves better edge preservation by combining pixel geometric distance and colour similarity [2]. However, the implementation of complex algorithms on FPGAs often faces the difficulties of high computation and resource consumption, which limits their real-time performance in practical applications.

In recent years, with the development of convolutional neural networks (CNNs), denoising algorithms based on deep learning have demonstrated better performance than traditional methods, especially in complex image conditions [3]. For example, the FPGA-based multi-scale image enhancement method proposed by Liu et al, combined with improved bootstrap filtering, effectively improves the contrast and details of infrared images in harsh conditions [4]. The method utilizes the parallel computing capability of FPGA to achieve high-quality denoising effect and real-time processing capability. In addition, in FPGA hardware design, rational allocation of resources and optimization of the computational process are crucial. An et al. provide a flexible and highly resource-utilized solution for implementing complex image processing algorithms through OpenCL accelerator design [5]. Zhang et al.'s study, shows that the resource consumption and power consumption of FPGAs can be significantly reduced by quantizing the neural network model, making them promising for low-power applications [6].

In this paper, the research uses FPGA as a hardware platform to investigate how to efficiently implement the adaptive Gaussian denoising algorithm to cope with the noise processing requirements in different image conditions. This paper implements module design on the PYNQ-Z2 platform using Verilog language. A dynamic weighting mechanism is employed in the filter to adaptively adjust based on the noise level, aiming to achieve effective denoising under various noise conditions. The FPGA enables efficient parallel processing to improve image quality and optimize resource utilization.

II. RESEARCH METHODS

A. Basic Principles

1) Characteristics of Gaussian filtering: As a classical linear smoothing filtering method, Gaussian filtering denoises and smooths the detail of the image based on the convolution kernel with normal distribution. The kernel function of the filter is determined by the distance from the pixel to the central point, and the further the distance, the

smaller the weight. This design can effectively suppress high-frequency noise, so that the image can obtain a better smoothing effect in a noisy condition. However, due to the uniformity of the weight distribution, the details of the image will also be inevitably affected by a certain degree of blurring, especially when dealing with edges and complex textures.

Another distinctive feature of Gaussian filtering is its isotropy, the weights of the filter are symmetrically distributed in the horizontal and vertical directions. This means that the filtering effect is independent of the direction, and the filter can perform smoothing operations uniformly over pixels in different directions. Although the isotropic property is advantageous in denoising tasks, it may lead to loss of information in images with significant directional structure, such as edge detection and target recognition.

In Gaussian filtering process, there is also the problem of edge loss. Due to the nature of the convolution operation, the edges and other detailed parts of the image tend to be blurred while the noise is filtered out. This problem is particularly evident in tasks such as edge detection that require highly accurate information about the structure of the image. Therefore, the parameters of the Gaussian filter, such as the standard deviation σ and the convolution kernel size, need to be carefully tuned according to the specific needs of the application scenario to strike a good balance between noise suppression and detail preservation.

2) Implementation of adaptive Gaussian filter: In order to overcome the limitations of the performance of standard Gaussian filtering under different noise conditions, adaptive Gaussian filtering achieves flexible processing for different noise levels by dynamically adjusting the size and standard deviation of the convolution kernel [7]. In areas of higher noise, the adaptive filter increases the size and standard deviation of the convolution kernel to enhance the suppression of noise. Although a larger convolution kernel increases the amount of computation in the filtering process, this strategy can effectively reduce the impact of noise on image quality.

In less noisy areas or areas that contain a lot of edge information, the adaptive filter then uses a smaller convolution kernel and standard deviation to reduce the smoothing of image details. With this adjustment, the filter can maximize the retention of key structural information in the image while denoising. In addition, for the processing of edge areas, the filter uses a more conservative strategy to avoid blurring of important edge details.

The core of adaptive Gaussian filtering lies in its dynamic weighting mechanism. The mechanism is able to analyze the local features of the image in real time and select the appropriate parameter configuration according to the noise level and pixel structure information. Compared with the traditional fixed-weight filter, the adaptive filter shows stronger robustness in complex conditions and can better cope with different types of noise and diverse image contents. By implementing it on a hardware platform, the adaptive Gaussian filter can further take advantage of its efficient parallel computation, providing reliable support for real-time image processing.

3) Hardware implementation of Gaussian filtering: When implementing Gaussian filtering on the FPGA platform, this study employs a dynamic weighting mechanism to adjust the parameters of the Gaussian kernel according to the local characteristics of the input image. This dynamic adaptive processing strategy ensures that the filter achieves the best results in different noise conditions.

B. Assumption Setting

This study builds on several important assumptions to ensure that the process of designing and implementing the adaptive Gaussian denoising module is scientific and tractable. Firstly, this study assumes that the image dataset used contains samples with multiple noise levels, including common pretzel noise and noiseless images, and is used to test the filter's performance under different noise conditions. Secondly, all input image data are greyscale images, and their dimensions are uniformly 256 × 256 pixels to reduce the complexity due to differences in image sizes, thus making the experimental results more consistent and comparable. Finally, this paper assumes that the FPGA platform can give full play to its parallel computing capability to meet the demand for real-time processing, ensuring that the filtering process can be completed in a reasonable time to support efficient processing in practical applications.

C. Module Design and Step-by-step Implementation

To ensure that the design of the adaptive Gaussian denoising module meets the expected performance, this paper adopts a step-by-step optimizations and module comparison approach to implement the key components in phases and tests and validate the algorithm at each stage to assess its performance under different noise levels.

In the initial stage, this paper designs and implements a one-dimensional Gaussian filter along the horizontal and vertical directions, respectively. The main objective of this phase is to verify the effectiveness of the filter in removing simple noise and to test its efficiency and computation time in dealing with high frequency noise. Although the one-dimensional filter is effective in smoothing some of the noise in the image, its performance is more limited in complex noise conditions.

To further improve the denoising effect, this study extends the one-dimensional Gaussian filter into a two-dimensional filter so that it can perform smoothing operations in both horizontal and vertical directions. At this stage, this study focuses on comparing the effects of the one-dimensional and two-dimensional filters, especially the rejection ability and the degree of preservation of image details under complex noise conditions. The experimental results show that the two-dimensional Gaussian filter has a significant advantage over the one-dimensional filter in dealing with complex noise.

On the basis of the two-dimensional filter, this study introduces a dynamic weighting mechanism to realize an adaptive Gaussian filter. The filter can adaptively adjust the weights according to the noise levels in different areas of the image, thus achieving a better balance between denoising and detail preservation. In this paper, the advantages of the dynamic weighting strategy in complex images, better preservation of image details while suppressing noise, are verified by comparing with the fixed weight filter.

Finally, in order to further optimize the performance of the filter, this paper designs an adaptive parametric filter and introduces two parameters, Gradient Sensitivity (GS) and

Texture Sensitivity (TS). By adjusting these parameters, the filter is able to achieve personalized denoising effects in different image areas. This study compares the outputs of different parameter combinations and finds the optimal parameter configurations to ensure that the filter achieves an ideal balance between denoising and image detail retention.

This step-by-step implementation and comparative validation approach systematically analyses the improvement points and performance enhancements at each stage of the design to ensure that the final adaptive Gaussian filter achieves good denoising results under different noise levels and image features, providing a reliable solution for image processing in complex application scenarios.

D. Validation and Assessment Methods

To comprehensively assess the performance of the designed adaptive Gaussian filter, this study employs a variety of verification and evaluation methods. Specifically, this paper combines objective metrics evaluation, subjective visual evaluation, platform testing and hardware verification to ensure the effectiveness of the filter in different application environments.

Firstly, in the objective metrics assessment, the Structural Similarity Index (SSIM) is used as the main metric in this paper to quantify the similarity between two images in terms of brightness, contrast and structure. The value of SSIM ranges from 0 to 1, where 1 means that the two images are exactly the same. Compared with the traditional mean square error (MSE) and peak signal-to-noise ratio (PSNR), SSIM is more in line with the perceptual properties of the human visual system with respect to image structure and contrast, and therefore has higher confidence in image quality assessment [8].

Secondly, a subjective visual assessment was carried out in this paper. By re-decoding the output of the filter as an image and comparing it to the filtered image, the filter was observed to determine how well it balanced noise suppression with detail retention. Particular attention was paid to the effect of edge areas and areas of complex detail to ensure that the adaptive filter was visually achieving the desired effect. This evaluation method can visually reflect the viewing quality of the image and provides a reference for the practical application of the algorithm.

Finally, this study tests and validates the filter on a hardware platform by deploying the filter to the PYNQ-Z2 platform and testing its performance in a real hardware environment. During the testing process, this paper focuses on the resource consumption of the filter on the FPGA platform and performs timing performance analysis. Through platform testing, this study verifies whether the filter can achieve the expected performance in a resource-constrained environment and provides data support for further optimization.

Combining the above verification and evaluation methods, this study comprehensively analyses the performance of the designed filter. These evaluation results provide a basis for the improvement of the filter and lay the foundation for future applications in more complex scenarios.

III. RESEARCH PROCESS

A. Design Architecture

The system adopts a single-module design, and the core module is the adaptive Gaussian filter module. This module is responsible for processing the input noisy image and adjusting the filter parameters to accommodate different noise levels and image area characteristics.

B. Module Design and Implementation

In order to design an adaptive Gaussian filter module, the filter kernel size is adjusted according to the noise level to ensure maximum image detail retention while eliminating noise. In this paper, a two-dimensional dynamically weighted filter is used to adapt to the noise characteristics of different areas to ensure the best balance between noise removal and detail retention.

1) One-dimensional Gaussian filter: The one-dimensional Gaussian filter uses fixed weights and performs a linear convolution operation to smooth image details and reduce high frequency noise. Its kernel weights are initialized as:

$$\text{Κερνελ}_{1\text{D}} = [1,2,1,2,4,2,1,2,1] \tag{1}$$

$$\text{Kernel}_{1D} = [1,2,1,2,4,2,1,2,1] \tag{2}$$

The kernel is based on a discrete approximation of the Gaussian distribution with larger central weights to enhance the local response. Due to the low computational overhead of the one-dimensional operation, it is well suited for FPGA implementations with limited computational resources.

2) Two-dimensional Gaussian filter: The two-dimensional Gaussian filter is suitable for dealing with complex noise by smoothing in both horizontal and vertical directions. The convolution kernel designed in this paper has a size of 3×3 and its specific weight matrix is as follows:

$$\text{Kernel}_{2D} = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{3}$$

The filter is able to retain the overall structure of the image while suppressing noise, but its performance is limited in complex noise conditions. To further improve the filtering effect, the filter introduces a weight adjustment mechanism based on local variance.

To control the noise level, the filter first calculates the mean and variance of the local region. Based on the calculated local variance, the filter dynamically adjusts the weights of the central pixel and the surrounding pixels to adapt to different noise levels. The formula for calculating the adaptive weights is as follows:

$$W_{i,j} = \begin{cases} \omega_c \cdot \left(1 - \alpha \cdot \frac{\sigma^2}{\sigma^2_{\max}}\right), & (i,j) = (1,1) \\ \omega_s \cdot \left(1 + \beta \cdot \frac{\sigma^2}{\sigma^2_{\max}}\right), & \text{other pixels} \end{cases} \tag{4}$$

The definitions and values of the parameters in the above equations are shown in Table I.

TABLE I. DEFINITION AND VALUES OF FILTER PARAMETERS

Parameter	Value	Description
$W_{i,j}$	Dynamic	Adaptive weights
ω_c	32	Baseline weight of the central pixel
ω_s	8	Baseline weights for surrounding pixels
σ^2	Dynamic	Local variance within the current window
$\sigma^2_{\max}$	255	Maximum value of variance, used for normalization
α	0.125	Coefficient to adjust the weight of the central pixel
β	0.0625	Coefficient to adjust the weights of surrounding pixels

When the variance of the local area is small, the weight of the central pixel is close to ω_c, which effectively preserves the image details; as the variance increases, the weight of the centre pixel gradually decreases to enhance the smoothing effect. In the high noise region, the weights of the surrounding pixels increase, which enables the filter to better suppress the noise; while in the region with small variance, the weights of the surrounding pixels are close to ω_s, which reduces the blurring effect caused by the smoothing operation.

This adaptive two-dimensional Gaussian filter realizes adaptive processing for different noise conditions by dynamically adjusting the weights. In the high noise region, the filter can effectively suppress the noise; in the detail-rich region, the filter can retain more image information. The implementation of this design on FPGA platform further improves the parallel computation efficiency of the filter, which is suitable for image processing tasks that require real-time processing capability.

Adaptive parameterized Gaussian filter: The adaptive parametric Gaussian filter, based on the two-dimensional dynamic weight filter, realizes the dynamic adjustment of the filter weights by combining the gradient and texture information of the image to make it more adaptive to the local features of the image. In the design of this filter, two key parameters, Gradient Sensitivity (GS) and Texture Sensitivity (TS), are introduced to adaptively adjust the weights of the filter according to the image features in different areas. This design enables the filter to effectively remove noise in complex noise conditions and to preserve the edges and details of the image as much as possible.

Gradient sensitivity is used to measure the edge strength of the region in which a pixel is located, and its calculation is based on the gray scale variation of the image [9]. In this study, the Sobel operator is used to compute the gradient in the horizontal and vertical directions of the image.

In the specific implementation, in order to improve the computational efficiency, this paper simplifies the Sobel operator, which is applicable to the gradient calculation of the window as follows [9]:

$$\text{Gradient}_x = \left(I_{0,2} + 2I_{1,2} + I_{2,2}\right) - \left(I_{0,0} + 2I_{1,0} + I_{2,0}\right) \quad (5)$$

$$\text{Gradient}_y = \left(I_{2,0} + 2I_{2,1} + I_{2,2}\right) - \left(I_{0,0} + 2I_{0,1} + I_{0,2}\right) \quad (6)$$

$$\text{GS}=|\text{Gradient}_x| + \left|\text{Gradient}_y\right| \quad (7)$$

Unlike the interpretation of (i, j) mentioned above, GS here represents a single value, numerically equal to the sum of the gradients in Eq. By setting a threshold for this summed value, it becomes easier to adjust the strength of the filter kernel, while also reducing the number of operations and enhancing hardware computational performance.

Texture sensitivity is used to detect the texture complexity within a local region of an image, and its calculation is based on the gray scale variance of the region [10]. In order to simplify the calculation, the following approximate formula is used in this paper:

$$\text{TS}=\frac{1}{9}\sum_{i=0}^{2}\sum_{j=0}^{2}\left|I_{i,j} - \frac{1}{9}\sum_{i=0}^{2}\sum_{j=0}^{2} I_{i,j}\right| \quad (8)$$

Based on the gradient sensitivity and texture sensitivity, the adaptive Gaussian filter dynamically adjusts the weights of the center pixel and the surrounding pixels. The specific weight adjustment strategy is as follows:

$$W_{\text{center}} = \begin{cases} W_{\max}, & \text{if } GS > GS_{\text{thr}} \text{ or } TS > TS_{\text{thr}} \\ W_{\max} - f(\sigma^2), & \text{otherwise} \end{cases} \quad (9)$$

In this case, $W_{\max}$ is the maximum weight of the center pixel, GS_{thr} and TS_{thr} is the threshold set by the user, and σ^2 represents the local variance of the region. This weighting strategy reduces the smoothing intensity in gradient and texture complex areas to protect edges and details; in flat areas, the weights are dynamically adjusted based on the local variance to achieve stronger denoising.

The weights of surrounding pixels are computed according to a multivariate function based on GS, TS and σ^2 to ensure that the weights of surrounding pixels match the noise level and detailed information in the region. In edges and highly textured areas, the weights of surrounding pixels are lower to minimize blurring effects, while in flat areas, the weights are higher to enhance noise suppression.

Finally, the output of the filter is calculated by weighted convolution of the above weights.

This adaptive filtering strategy can dynamically respond to the characteristics of different image areas in the hardware implementation, ensuring that the filter strikes a balance between denoising and detail preservation. With the parallel computing implementation on an FPGA platform, the filter is suitable for complex image processing tasks that require efficient real-time processing.

C. Simulation and Testing

This section shows the simulation results using Xilinx Vivado 2023.2 and verifies the effectiveness of the proposed algorithm by comparing the performance of different filter designs. During the testing process, this paper focuses on analyzing the performance of one-dimensional and two-dimensional filters, two-dimensional fixed and dynamic weight filters, and explores the effect of parameterized tunable adaptive filters with different parameter combinations.

This paper first compares one-dimensional and two-dimensional filters to analyze their differences in denoising performance and detail retention. Next, the performance of two-dimensional fixed-weight and dynamic-weight filters is evaluated to highlight the advantages of dynamic tuning in different image regions. Finally, the impact of adaptive

parameterized filters under various parameter combinations is tested to examine how gradient and texture sensitivity adjustments affect filtering performance.

1) One-dimensional filtering vs. two-dimensional filtering: As shown in Figure 1, the one-dimensional fixed-weight filter has a better denoising effect in a single direction, but its processing is limited to the horizontal or vertical direction, and its ability to suppress complex noise is limited. In contrast, the two-dimensional fixed-weight filter filters in both the horizontal and vertical directions, and thus provides a more comprehensive treatment of noise in the image and performs better in retaining details.

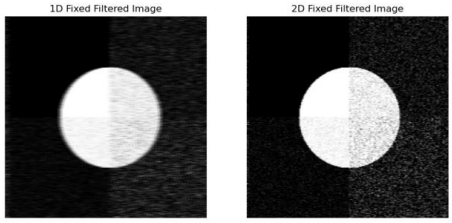

Figure 1. Comparison of 1D and 2D fixed weight filters

2) Comparison of two-dimensional fixed and dynamic filtering: The figure 2 illustrates the performance comparison between a two-dimensional fixed weight filter and a two-dimensional dynamic weight filter. The dynamic weighting filter is able to adaptively adjust the filter strength according to the local noise level and characteristics of the image. In high-noise areas, the filter enhances the denoising effect; in edge and detail-rich areas, it reduces the smoothing intensity, thus retaining more detail information. This adaptive tuning strategy improves the robustness of the filter in different conditions.

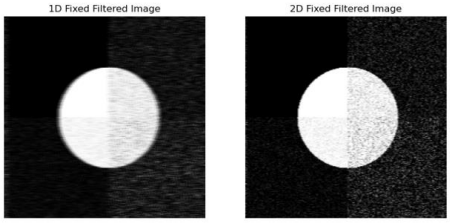

Figure 2. Comparison of 2D fixed weight filtering with 2D dynamic weight filtering

3) Comparison of SSIM: In order to evaluate the performance of various types of filters more comprehensively, firstly, in the objective metrics evaluation, the Structural Similarity Index (SSIM, Structural Similarity Index) is used as the main metric in this paper to quantify the similarity of the two images in terms of brightness, contrast and structure. The value of the SSIM is in the range of 0 to 1, where 1 indicates that the two images are identical. Compared with the traditional mean square error (MSE) and peak signal-to-noise ratio (PSNR), SSIM is more in line with the human visual system's perceptual characteristics of image structure and contrast, and therefore has higher confidence in image quality assessment [8]. SSIM can better reflect the quality of the filtered image by measuring the similarity of the images in terms of brightness, contrast and structure. Table II shows the results of SSIM and MSE of the output images of different filters with respect to the original image.

TABLE II. COMPARISON OF SSIM AND MSE FOR DIFFERENT FILTERS

Filter Type	MSE	SSIM
Noisy image	0.0166	0.2932
1D Fixed Weight Filter	0.0197	0.2989
2D Fixed Weight Filter	0.0605	0.3371
Adaptive 2D dynamic weight filter	0.1164	0.3804

The data in Table II show that the adaptive two-dimensional dynamic weighting filter achieves the highest value in the SSIM metric, indicating that it outperforms the other filters in preserving the structural information of the image. Although the higher MSE of this filter indicates that it is slightly deficient in terms of pixel-level error, its performance in terms of visual quality is more in line with the perceptual characteristics of the human visual system. In contrast, the fixed-weight filter has a poor balance between denoising and detail preservation, while the one-dimensional filter is limited by its directionality and is not as effective as the two-dimensional filter in dealing with complex noise.

4) Comparison of parameter combinations for parameterized tunable adaptive filters: Figure 3 illustrates the output of the filter for different combinations of gradient sensitivity (GS) and texture sensitivity (TS). Adjusting these two parameters can control the response of the filter in different image areas for personalized noise suppression and detail preservation.

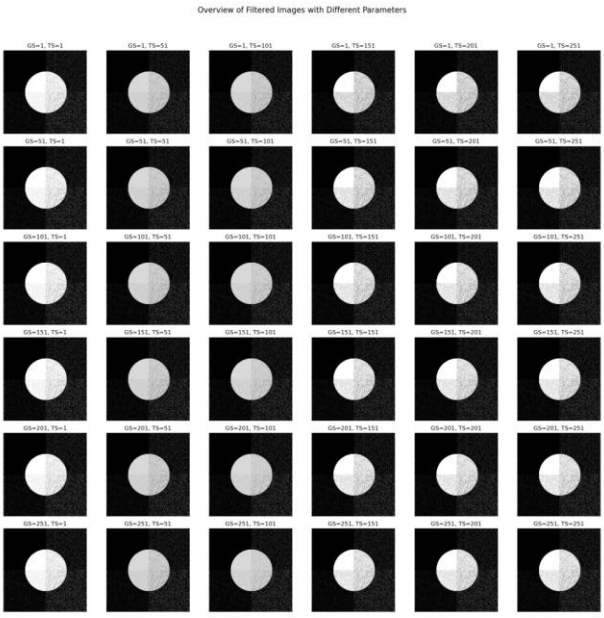

Figure 3. Output effect of parameterized tunable adaptive filter with different parameter combinations

The test results reveal that the filter provides a stronger smoothing of all areas when both GS and TS are higher. Although noise is effectively suppressed, edge and texture details are also significantly blurred. This combination is suitable for scenes where noise suppression is very demanding and details are not important.

When GS is high and TS is low, the filter reduces the smoothing effort when detecting edge areas, thus retaining more edge information. However, due to the lower TS, the filter still smoothes the texture areas strongly, resulting in the loss of some texture details. This configuration is suitable for applications where structural information (e.g., edges) needs to be preserved, such as edge detection.

On the contrary, at lower GS and higher TS, the filter

reduces the smoothing effort in texture-rich areas, preserving more texture information. However, due to the lower GS, the edge areas are also smoothed more strongly, which may result in a loss of edge detail. This combination is suitable for scenes that require high texture details, such as the processing of remote sensing images.

With the combination of both higher GS and TS, the filter provides the best protection in the edge and texture areas. The details and structural information of the image are fully preserved, but the suppression of noise is reduced due to weaker smoothing. This configuration is suitable for scenarios with less noise but higher requirements for image details and visual quality, such as medical image processing.

By adjusting the medium GS and TS values, the filter strikes a balance between noise removal and detail retention. This combination effectively removes noise while retaining edge and texture information to a certain extent and is suitable for applications that require a balance between noise suppression and detail quality, such as daily surveillance image processing.

These experimental results show that reasonable adjustment of GS and TS parameters can realize flexible filtering effects for different application scenarios. In medical imaging, high GS and TS values can be selected to maximize detail retention, while in surveillance scenarios, lower GS and TS values can be selected to enhance noise suppression.

Summary: By comparing the above three categories, this study verifies the performance of each type of filter in different applications. First, the two-dimensional filter performs better than the one-dimensional filter in dealing with complex noises, as it considers both horizontal and vertical pixel variations. Second, adaptive filters are able to adapt more flexibly to different noise conditions by dynamically adjusting the weights, which improves the filtering performance. Finally, the parametric tunable adaptive filter achieves a personalized image enhancement effect through the combination of gradient sensitivity and texture sensitivity.

The flexibility of the parametric filter makes it adaptable to a variety of application scenarios, such as medical image processing, remote sensing data analysis and intelligent surveillance. The results of this simulation demonstrate the parallel computing capability and flexibility of the FPGA platform, which provides data support for further optimization of the filter design. Future research can explore more complex parameter combination strategies and further improve the performance of the filter in terms of real-time and accuracy.

D. Hardware Implementation and Testing

After completing the simulation verification, this study deploys the design for testing on the PYNQ-Z2 development platform, an FPGA development board based on the Xilinx Zynq-7000 SoC with rich hardware resources and parallel computing capabilities. This platform provides a good foundation for the efficient implementation of filtering algorithms. The implementation of the adaptive Gaussian filtering algorithm on FPGA significantly improves the speed of image denoising and enhancement and ensures that the system has real-time processing capability. In practical tests, the system can process input data efficiently, laying the foundation for realizing millisecond response.

1) Implementation process and testing: In this study, Vivado was used to analyze the design for resource utilization and timing, and to ensure that the filter was capable of parallel computation on an FPGA platform. The filtering and image processing processes are fully parallelized to maximize the computational efficiency of the hardware. This implementation is intended to provide hardware support for complex real-time image processing tasks.

2) Analysis of resource utilization: Based on the resource utilization report generated by Vivado Implementation, this study counts the consumption of various key resources of the design on the PYNQ-Z2 platform. Table III shows the specific data of resource utilization.

TABLE III. PYNQ-Z2 FPGA RESOURCE UTILIZATION

Resource Type	Used	Available	Utilization
LUT	2808	53200	5.28%
FF	32	106400	0.03%
DSP	36	220	16.36%
IO	34	125	27.20%
LUT	2808	53200	5.28%
FF	32	106400	0.03%
DSP	36	220	16.36%

From the data in Table III, the design has a low occupancy of basic resources such as LUT and FF, which indicates that the implementation has a high resource utilization efficiency. The utilization rate of the DSP unit is 16.36%, which indicates that the system gives full play to the digital signal processing capability of the FPGA. In addition, the IO unit occupies 27.2%, which is consistent with the system's need for large data transfers. Overall, the design achieves a balanced use of resources and provides sufficient hardware support for the parallel execution of the filtering algorithm.

3) Timing and performance analysis: Based on the timing summary report generated by the Vivado tool, the timing performance of the system is analyzed in detail in this study. Table IV shows the main timing metrics.

TABLE IV. PYNQ-Z2 FPGA TIMING SUMMARY

Parameter	Value	Description
Worst Negative Slack (WNS)	inf	No negative timing margin, all paths satisfy timing
Total Negative Slack (TNS)	0.000 ns	Total Negative Slack 0.000 ns
Worst Hold Slack (WHS)	inf	Hold time satisfied, design has no hold time violations

From the data in Table IV, all the paths in this design satisfy the timing requirements without timing violations. This shows that the system can achieve the expected performance without violation despite the fact that no explicit timing constraints are set in the design. This optimization ensures that the filter can complete the image processing in milliseconds to meet the requirements of real-time applications.

4) Resource and performance optimization features: The design achieves high resource utilization and parallel computing efficiency on the FPGA platform. The low occupancy of LUT and FF resources indicates that the hardware design of the system has high resource

optimization capability. In the implementation of the filtering algorithm, the DSP unit is fully utilized to enhance the performance of the filtering process. The relatively high IO resource consumption of the system is in line with its characteristic of handling a large number of data transmission tasks. In addition, the timing analysis shows that the system has excellent real-time performance and can meet the demands of complex image processing tasks.

5) Summary: Through the implementation and testing on the PYNQ-Z2 platform, this paper verifies the high efficiency and feasibility of the adaptive Gaussian filtering algorithm on FPGAs. Thanks to the parallel computing capability and low-latency characteristics of FPGA, the system processes images faster and shows good real-time performance.

IV. Conclusion

In this study, an adaptive denoising algorithm based on FPGA is proposed, and the module design and simulation are completed by Verilog language. The algorithm utilizes an adaptive Gaussian filtering method to achieve an effective balance between denoising and detail preservation between different noise levels and image areas by dynamically adjusting the filter weights. The results show that the implementation of the algorithm on the PYNQ-Z2 development platform not only has good denoising effect, but also exhibits high resource utilization efficiency and excellent real-time performance.

Through experimental verification, the system is able to complete the filtering processing of images in a short time, ensuring its applicability in application scenarios with high real-time requirements. Compared with the traditional fixed-weight filter, the adaptive filter demonstrates greater flexibility and is able to dynamically adjust the filter strength according to the edge and texture features of the image. This feature provides a better balance between noise suppression and detail preservation. The parametric design supports individual requirements in different application scenarios, such as the need for detail preservation in medical images or noise suppression in remote sensing images.

In addition, the implementation of the system on the FPGA platform makes full use of hardware resources such as LUTs and DSPs to achieve low resource consumption and efficient parallel processing capability, and the timing analysis results of the Vivado tool show that the system is able to achieve stable performance without timing violations, which guarantees the stability and reliability of the filter in practical applications.

Despite the good results achieved in this study, there is still some room for optimization. First, future research can further optimize the algorithmic structure of the filter to reduce the consumption of FPGA resources and improve the computational efficiency of the system. Second, more complex parameter tuning strategies can be explored to enhance the performance of the filter in extreme noise conditions. In addition, with the continuous development of FPGA hardware technology, porting the algorithm to newer FPGA platforms can also help to further improve the system performance.

References

[1] R. C. Gonzalez and R. E. Woods, “Digital image processing. Upper Saddle River”, N.J. Prentice Hall, 2008.

[2] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in Sixth international conference on computer vision, IEEE, 1998, pp. 839–846.

[3] A. Ranjan and S. M. Azeemuddin, “Image denoising using convolutional neural network,” ICAC3N, IEEE, 2022, pp. 2315–2319.

[4] J. Liu et al., “Multi-scale FPGA-based infrared image enhancement by using RGF and CLAHE,” Sensors, vol. 21, no. 7, pp. 2446, 2021.

[5] J. An, D. Zhang, K. Xu, and D. Wang, “An OpenCL-based FPGA accelerator for faster r-CNN,” Entropy, vol. 24, no. 10, pp. 1346, 2022.

[6] X. Zhang, X. Wei, Q. Sang, H. Chen, and Y. Xie, “An efficient FPGA-based implementation for quantized remote sensing image scene classification network,” Electronics, vol. 9, no. 8, pp. 1344, 2020.

[7] G. Deng and L. W. Cahill, “An adaptive gaussian filter for noise reduction and edge detection,” in Proceedings of the IEEE international conference on pattern recognition, IEEE, 1994, pp. 1615–1619.

[8] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Transactions on Image Processing, vol 13, no 4, pp. 600–612, 2004

[9] I. Sobel and G. Feldman, A 3x3 Isotropic Gradient Operator for Image Processing. Stanford Artificial Intelligence Project, 1968.

[10] Haralick, R. M., Shanmugam, K., & Dinstein, I. "Textural Features for Image Classification." IEEE Transactions on Systems, Man, and Cybernetics, SMC-3(6), pp. 610–621, 1973.