Proceedings Paper:
Yang, P, Wang, Q, Guo, T et al. (2 more authors) (2018) Adaptive Optimization Design of Vector Error Diffusion Algorithm and IP Core for FPGA. In: Proceedings - 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation. 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation, 08-12 Oct 2018, Guangzhou, China. IEEE , pp. 1193-1198. ISBN 978-1-5386-9380-3

https://doi.org/10.1109/SmartWorld.2018.00208

Adaptive Optimization Design of Vector Error Diffusion Algorithm and IP Core for FPGA

Pengfei Yang, Quan Wang, Tao Guo, Wei Li
School of Computer Science and Technology
Xidian University
Xi'an, China
qwang@xidian.edu.cn

ZhiQiang Zhang
School of Electronic and Electrical Engineering
University of Leeds
Leeds, UK
Z.Zhang3@leeds.ac.uk

***Abstract*—A vector error diffusion algorithm can obtain better halftone results than a scalar error diffusion algorithm in digital printing, thus, extensive research work about the vector error diffusion has been done to decrease the time that users wait for printing. In this paper, an improved vector error diffusion IP core is proposed. The IP is implemented in FPGA and can meet the requirement of real-time printing by the improvements as follow: three R G B planes are computed in parallel, a matrix-valued error filter is designed to diffuse error among the three planes, matrix-valued pre-stored memory is created to speed multiplications and five stage pipelines are adopted to replace traditional sequential processes. Based on the improvements, we build a practical hardware test system on the SoCKit platform. The test results show that optimal algorithm only needs one clock circle to get the halftone result of a pixel on average and can meet the requirements of practical printing.**

***Index Terms*—Vector error diffusion, FPGA**

I. Introduction

For a typical process of printing, the Raster Image Processor(RIP) module, which converts the content to be printed into a bitmap is processed first, then, by translating, caching and decompressing, the printed output obtained. The RIP module is the kernel technology, it usually includes data resolution, half-tone processing, data recombining and compressing. As the most important process of printing, the halftone values determine the performance and quality of printing. Every pixel of a source image is synthesized by red, green and blue, with different grey values arranging from 0 to 255, thus, there are so many possible colours for any pixel. However, there are only two choices for the printer: print the dot or not. The halftone algorithm builds the connection between them.

There are many methods to get halftone images, including lookup-based schemes such as Look Up Table (LUT) halftoning, error diffuse-based schemes such as Error Diffusion(ED), Dot Diffusion iteration-based schemes such as Direct Binary Search(DBS), and threshold-based schemes such as Ordered Dithering(OD). Of those mentioned above, the error diffusion algorithm is most advantageous because it can produce a relatively satisfactory image, and it has become the mainstream solution for the printer industry. Many works have been carried out to get a better error diffusion results.

The research is supported by National Natural Science Foundation of China, under Grant Nos. 61572385, 61702395, 61711530248.

Nowadays the print density becomes higher with the development of printing technology, and more ink dots can be printed in a unit area, the amount of calculation of halftoning is becoming larger and larger, However, as a typical serial algorithm, traditional error diffusion algorithm can process only one pixel at a time even on a PC platform. A larger storage is also required to caching the error value of the previous element before calculating the value of the next pixel. One more thing, off-line printing which without a computer is becoming more and more popular.This requires an adaptive design of the error diffusion algorithm to avoid significantly increasing the print waiting time and detracting from the user experience.

II. Related works

People always hope that the high-quality images or videos can be obtained in a quick and convenient way, so the image and video processing methods and platforms are required to optimized continuously[1-4]. For the error diffusion of colour imagines, the error of one plane can diffuse not only in its plane but also among other planes to reduce the loss of colour information. The traditional colour error diffusion algorithms handle each colour separately, by considering the R, G, and B planes as three unattached images with continuous grey error, the propagation and quantization of every plane are carried out independently, and the halftone results are always unsatisfactory. Vector colour error diffusion algorithm has been proposed, during which the correlation among different colour planes is considered, and error diffusion is extended to colour images by using error filters with matrix-valued coefficients.

Vector colour error diffusion was first described by Haneishi al.[5], and many improvements were proposed later. Zhigang Fan[6] presented a simple but effective method for reducing the boundary artefacts. Shaked Doron al.[7][8] designed a vector colour error diffusion algorithm based on the Minimal Brightness Variation Criterion(MBVC). They proposed a minimum brightness distribution standard MBVC according to the principle of colour and the characteristics of human vision and applied it to the quantization of a vector error diffusion algorithm. Compared with the algorithm proposed earlier, the colour noise of the halftones was greatly improved. Dantera al.[9][10] development an optimal noise distortion error dif-

fusion coefficient matrix using half_toning, that converted the brightness error into a colour component that was insensitive to human vision. Similar to a MBVC-based vector error diffusion algorithm, it still used the average quantization method, the halftone result was eight primary colors, and the problem of brightness difference still existed. Meng al.[11] fine-tuned the Floyd-Steinberg matrix of classical error diffusion coefficient, and added a source image input to realize edge enhancement, and thereby greatly improve the problem of a less hierarchical level of vector error diffusion based on MBVC. Jing Zhang combined the advantages of the above and proposed a quantizer with a minimum brightness distribution standard; she also replaced the Floyd-Steinberg error kernel with the vector of the optimal error diffusion filter error diffusion algorithm to get better colour halftone images.

For the application of a vector error diffusion algorithm, Ryosuke al.[12] proposed a new colour quantization method to generate dichroic images based on vector error diffusion and particle swarm optimization. The Floyd-Steinberg error diffusion coefficient matrix was used to implement the vector error diffusion algorithm, and the particle swarm optimization palette based on human vision was used to improve the vector error diffusion algorithm.

At the same time, The field programmable gate array(FPGA) has undergone rapid development. The larger programmable area and more flexible internal connection style provided greater convenience for a programmer to perform complex functions in a simple way[13], one more thing, the easy IP encapsulation facilitates the reuse of the function modules[14]. Before the work in this article, we designed an error diffusion IP core on FPGA[15], but the results of halftones were not satisfactory for high-quality print because it was implemented with the scalar halftone algorithm; and the colour information among different planes was ignored. This paper presents an improved vector error diffusion IP core that can be implemented in FPGA in real- time. The main contributions of this paper are as follows:

(1) The difficulties and challenges in implementing the vector halftone algorithm on FPGA are analyzed and detailed design and implementation are carried out.

(2) The algorithm is encapsulated into a reusable IP core and tested with the actual hardware system to prove that the algorithm can meet the actual needs of the offline printer printing.

The rest of this paper is organized as follows. In section III, the halftone results of the vector error diffusion algorithm and the scalar error diffusion algorithm are compared to prove the superiority of former; then, the details of the improvements are presented. Section IV describes the encapsulation of the algorithm into an IP core and its test in a practical system. Section V summarizes our work.

III. Vector Error Diffusion Optimization

At first, we compare the halftone colour images obtained by the vector error diffusion algorithm and the traditional error diffusion algorithm separately. These two algorithms were implemented with Matlab, the vector error diffusion algorithm adopts the optimal error coefficient matrix Linearized CIELab[9], the source image is a 256 * 256 Lena image. We calculated the peak signal to noise ratio(PSNR), weighted signal to noise ratio(WSNR), linear distortion measure(LDM) and Universal Quality Index(UQI)[16][17][18] of the scalar error diffusion algorithm and vector error diffusion algorithm using Linearized CIELab, Opponent Colour Space[19][20], YUV and YIQ[21]. The result is shown in table I. It can be seen that, the PSNR, WSNR and UQI of the vector error diffusion algorithm are higher than those of the scalar error diffusion algorithm, and that the LDM is lower than that of the scalar error diffusion algorithm, the vector error algorithm obtains better halftone results for the colour images than the scalar algorithm does. The vector error diffusion algorithm is the inevitable choice to get better print quality for a printing system. Following is a detailed discussion of the specific optimization and implementation of the algorithm.

TABLE I
Comparison between tradition error diffusion and vector error diffusion

Method	scalar Algorithm	Linearized CIELab	Opponent Color Space	YUV	YIQ
PSNR(dB)	5.4975	5.5670	5.5329	5.4938	5.5233
WSNR(dB)	13.5442	14.0807	13.8455	13.6149	13.4952
LDM	0.1497	0.1390	0.1460	0.1490	0.1514
UQI	0.0741	0.1029	0.0673	0.0662	0.0716

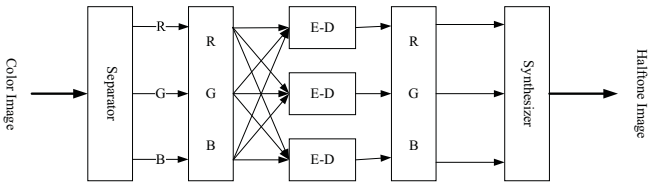

Fig. 1. Parallel processing of vector error diffusion

A. Parallel Processing

The traditional colour error diffusion algorithms neglect colour information among different planes. Error propagation and quantization of each channel are handled separately, it is easy to make a parallel design. The parallel design of the vector error diffusion algorithm is difficult, because the colour information among RGB planes is not independent, the error of one channel diffuses among other planes to reduce the loss of colour information. In this paper, we process the three colours in parallel fashion as shown in figure 1. Error diffusion between different planes is carried out in part called "E-D", which includes the operation of the vector colour error diffusion filter coefficient. This can greatly shorten the processing time compared with serial processing and can improve the processing efficiency.

For an RGB image, the error filter coefficient is a 3*3 matrix, which is adapted to reduce the mean squared error between the halftone and the original. Based on the human

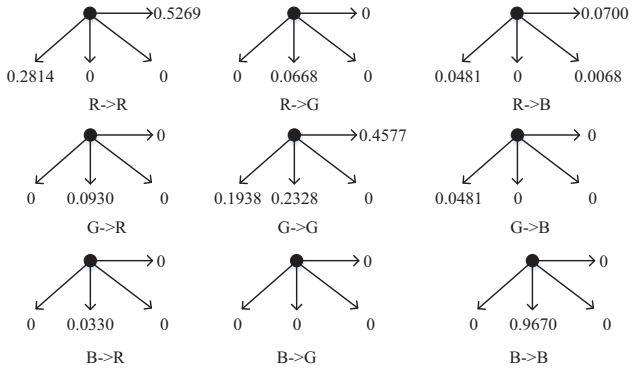

Fig. 2. Error diffusion coefficient based on Linearized CIELab

visual system, four optimized matrix-valued error filters were proposed: YIQ space, YUV space, opponent colour space, and Linearized CIELab[19][20][21] colour space. In this paper, we test our design using these four matrix-valued error filters. Figure 2 is a sample of an error diffusion coefficient based on Linearized CIELab, The sum of coefficients of any channel is 1, ensuring the error can be diffused to the local colour plane and other planes completely without information losing.

B. Process of Halftone and Error Value Generation

Many hurdles must be crossed when using FPGA to implement error diffusion calculations. First, a hardware description language such as Verilog HDL cannot address negative operations directly, but the calculation of halftone and error value involves negative numbers. Second, an established system always sets a fixed data width; for example, 8 bits is defined for a variable ranged from 0 to 256, which would cause data overflow during data processing.

In this paper, the mosaic operator of unsigned in Verilog HDL is used as a sign expansion such that the updated pixel values are applicable to signed operations. Every number is composed of an unsigned number and the sign bit. The unsigned number is the absolute value of the original data, and the sign bit, which is stored in an extra register, indicates the positive or negative of the data. The modified design converts the calculation of signed numbers into the operation of unsigned numbers, the problems involving negative numbers and data overflow are all resolved. Figure 3 illustrates the process of error diffusion and error value generation. ERR[7..0] feeds back the error diffusion values of neighbouring pixels in the last row. Accdata[9..0] updates the pixel value based on ERR[7..0] and data[7..0]; then, the module proceeds to get the err-data and the halftone values of the pixel by comparing accdata[9..0] with the threshold value. To facilitate looking up the error diffusion values in the next step, the sign bits(err_data_sig) of the error values(signed or unsigned) are extracted alone, the negative error values are output in the form of their absolute value, and the next level's operation

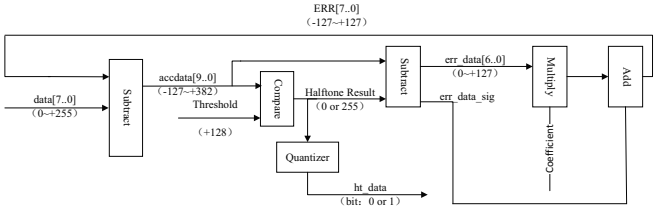

Fig. 3. Halftone and error value generation block structure

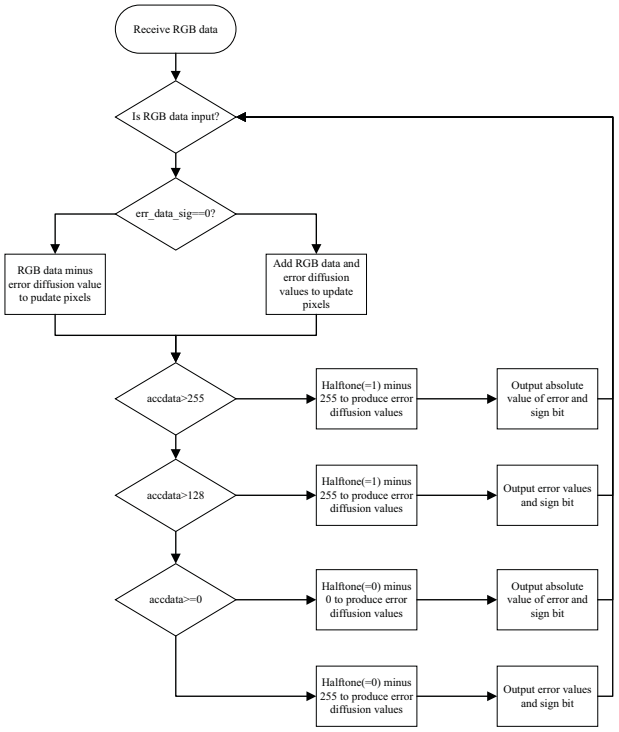

Fig. 4. Flowchart of Error values generation

is carried out by judging the sign bit of the error diffusion values.

Figure 4 is the flowchart of generating error values. When updating the values of a pixel, four different cases should be considered: greater than 255, between 255 (equal) and 128, between 128 (equal) and 0, and less than 0. The error values and halftone values will be determined and output corresponding to different cases. The pipelining of the entire halftone system is allowed because the halftone values of a pixel can be produced within one clock cycle under this design.

As is shown in figure 2, for a single colour, there are 128 error values ranging from 0 to 127 that will diffuse into three RGB planes in four directions, the process of error diffusion mainly involves subtraction and multiplication operations. Direct multiplication in FPGA will require extensive logic resources and increase the system resource occupation.

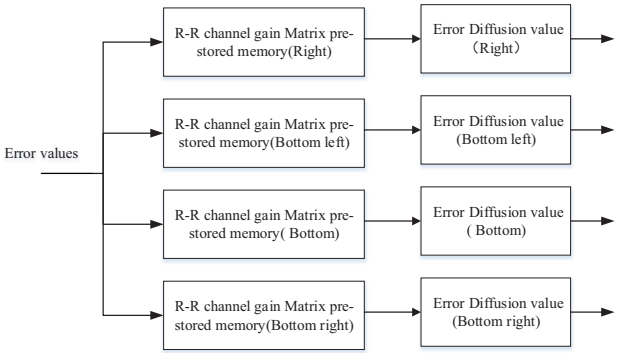

Fig. 5. Matrix-value pre-stored memory structure

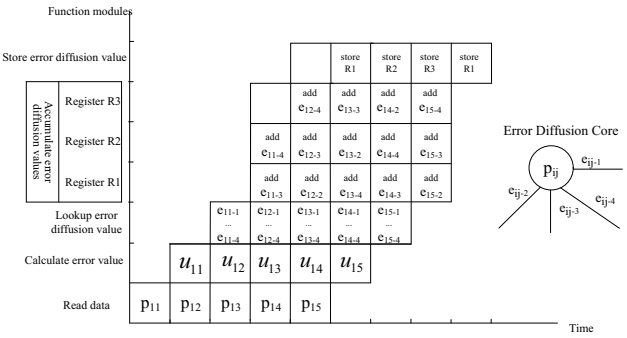

Fig. 6. Pipeline timing chart

This proposal can hardly meet the requirement of real-time processing, so optimizing the multiplication operations in the system is essential.

In our system, we calculate the error diffusion values and store them in a memory called matrix-value pre-stored memory. The error values will serve as indexes for looking up the error diffusion values in each direction in the corresponding table to minimize the computing time when error diffusion values are required. Complex multiplication operations will be eliminated from the system. Twelve matrix-value pre-stored memories are constructed to store error diffusion values in the four directions of three planes. The error diffusion values corresponding to each of these diffusion coefficients can be referenced from the error values. The figures for the memory design for the R channel diffused into it are shown in figure 5, this will save logic resources and shorten the computing time.

C. System Pipeline Design

Commonly, pixel data cannot be calculated until the error diffusion values of the previous pixel and the accumulation values of error diffusion of the last raw data from three planes are obtained. It is a typical serial execution and will seriously affect the real-time performance of the system. This paper introduced a parallel design so the algorithm can be processed in the pipeline.

Pipeline design is the most common speed optimization technique for this problem[22]. The pipeline timing diagram of the vector error diffusion algorithm is designed by indicating each pixel value and setting the optimal colour space filter, as shown in figure 6. The horizontal axis represents time, and the vertical axis represents the function modules. The error diffusion is divided into five parts, i.e., reading the original pixel data, calculating the halftone value, determining the error diffusion value corresponding to the error value, accumulating the error diffusion values and storing the error diffusion values.

To further improve the efficiency of computing, three registers are applied in the accumulation module to cache the multiple error diffusion values simultaneously, the final results will be written into the storage module later. High efficiency will be achieved through the pipeline process for the system if and only if these modules are in the required timing relations shown in the diagram.

IV. IP Core Design and Experiment Validation

FPGA provides an integrator for users to integrate their own design into an intellectual property for reusability and to accelerate the process of system design and development. We integrated our optimization into a vector error diffusion IP core, referring to the master-slave signal of Avalon[13]. Using the vector error diffusion IP, a verification system was constructed based on the SoCKit development board. Then, a series of tests and comparisons are as follows:

At first, we compare the halftone images generated through the verification system and Matlab using the same algorithm. The size of these diagrams are all 256 * 256, and the results are shown in figure 7, where (a), (b), and (c) show the source images, the results of Matlab and results of the verification system. The results obtained for the verification system were basically the same as those obtained in Matlab.

In addition, the PSNR, WSNR, LDM and UQI of these diagrams are calculated as shown in table II. The PSNR of halftone results generated by the two methods are almost the same, the WSNR is slightly lower than the Matlab result, the LDM is slightly higher than the Matlab result, and the UQI index is slightly lower than the Matlab result. These slightly worse results are foreseeable because the decimal of float numbers is ignored in the calculation process on the FPGA platform.

TABLE II
COMPARISON BETWEEN MATLAB AND FPGA

Source images	PSNR(dB)		WSNR(dB)		LDM		UQI	
	Matlab	FPGA	Matlab	FPGA	Matlab	FPGA	Matlab	FPGA
Lena	5.5670	5.5538	14.0807	13.5479	0.1390	0.1505	0.1030	0.0743
Pepper	5.6957	5.6772	11.0739	10.9267	0.2470	0.2525	0.0238	0.0134
House	4.8483	4.8609	15.0766	14.7521	0.1212	0.1269	0.1707	0.1688
Girl	6.1206	6.1438	11.6486	11.5013	0.1951	0.2031	0.0658	0.0267
Trees	4.9902	5.0196	11.7978	11.5530	0.1908	0.1982	0.1603	0.1547
Sailboat	5.1973	5.1876	10.3822	10.1117	0.2246	0.2326	0.1376	0.1490

Third, we compared the results of 256*256 Lena colour image generated by the verification system and Matlab in three colour channels of R, G and B; a window was set to show the difference of these results. The pixel window size was 8 * 8, starting from the (70,70) pixels of each channel, and the

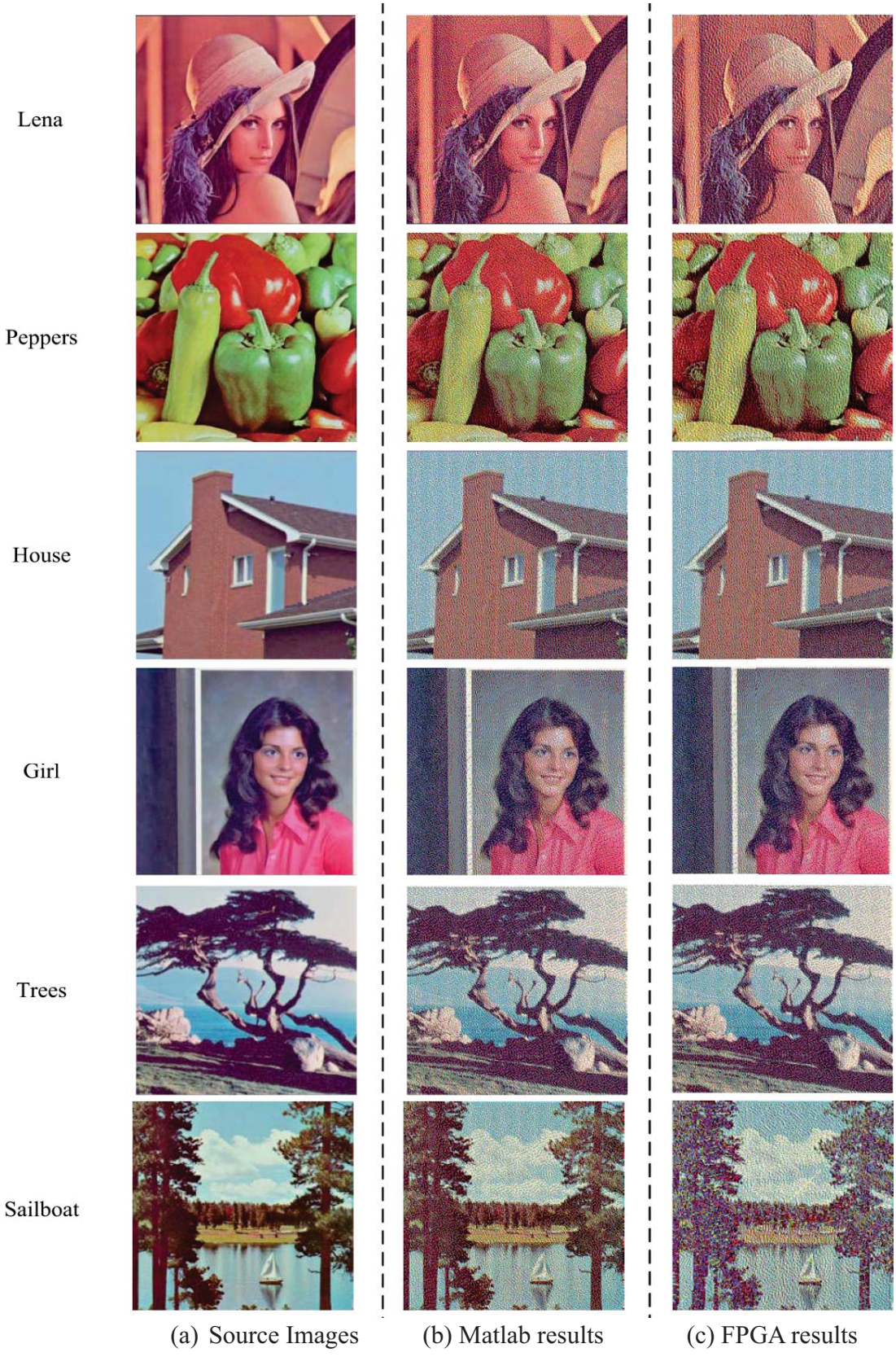

Fig. 7. Comparison results of color images

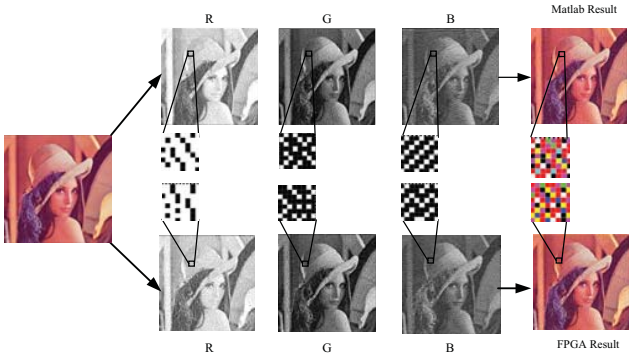

Fig. 8. Comparison of color image generated by Matlab and FPGA

TABLE III
DATA OF COMPARATIVE EXPERIMENT

Image	Size	Running time of Matlab(ms)	Running time of FPGA(ms)
Lena	256*256	9324.87	2.62
A4 72dpi	595*842	71854.07	20.04
A4 96dpi	794*1123	126358.58	35.67
A4 120dpi	1487*2105	442780.88	125.21
A4 150dpi	1240*1754	300109.38	87.00
A4 300dpi	2479*3508	1221622.54	347.85

result is shown in figure 8. The slight differences caused by the calculation process on the FPGA can be seen from the windows.

Finally, some images of A4 paper of 150 dpi (1240 * 1754) and 300 dpi (2479 * 3508) are used as the input to the verification system because the printer inputs are usually images with the A4 size. The halftone results of those images can be obtained correctly, which validates the correctness of the algorithm designed in this paper when dealing with large images.

For the efficiency of the IP, a clock frequency from 25 MHz to 100 MHz can be used as the input clock, the calculation of three colour planes are executed in parallel, and on average, it takes only one clock circle to obtain the halftone result of a pixel. Using 25 MHz as the input clock frequency can generate halftone results of 3 * 2.5 * 107 pixels per second. In this paper, the running time of the Lena image, A4 72 dpi image, A4 96 dpi image, A4 120 dpi image, A4 150 dpi image and A4 300 dpi image are calculated based on Matlab and the verification system separately, as shown in Table III. It can be seen from the table that the vector error diffusion algorithm based on the verification system has a shorter running time than does the algorithm based on Matlab.

V. CONCLUSION

We made a series of improvements on a vector colour error diffusion algorithm. The improved algorithm was implemented in FPGA, and the IP core of a vector colour error diffusion algorithm was designed. We implemented the IP core on the SOPC system and tested the function of the system with matrix-valued filters. The results of the experiment indicate that the IP core that we designed has a faster execution speed and higher scalability and could meet the requirements of real-time printing. In the next phase, we will integrate this IP core into our printer SoC controller to reduce the production costs and improve the user experience.

REFERENCES

[1] Shi Y, Wang K, Chen C, et al. Structure-Preserving Image Super-resolution via Contextualized Multi-task Learning[J]. IEEE Transactions on Multimedia, 2017.
[2] Lu Z, Ramakrishnan S, Zhu X. Exploiting Video Quality Information with Lightweight Network Coordination for HTTP-based Adaptive Video Streaming[J]. IEEE Transactions on Multimedia, 2017.
[3] Song J, Yang Y, Huang Z, et al. Effective multiple feature hashing for large-scale near-duplicate video retrieval[J]. IEEE Transactions on Multimedia, 2013, 15(8): 1997-2008.
[4] Ding K, Fan B, Huo C, et al. Cross-Modal Hashing via Rank-Order Preserving[J]. IEEE Transactions on Multimedia, 2017, 19(3): 571-585.
[5] Haneishi H, Suzuki T, Shimoyama N, et al. Color digital halftoning taking colorimetric color reproduction into account[J]. Journal of Electronic Imaging, 1996, 5(1): 97-107.
[6] Fan Z. Boundary artifacts reduction in vector error diffusion[C]//Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts IV. International Society for Optics and Photonics, 1998, 3648: 480-485.
[7] Shaked D, Arad N, Fitzhugh A, et al. Ink relocation for color halftones[C]//PICS. 1998: 340-343.
[8] Shaked D, Arad N, Fitzhugh A E, et al. Color diffusion: error diffusion for color halftones[C]//Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts IV. International Society for Optics and Photonics, 1998, 3648: 459-466.
[9] Dantera-Venkata N, Evans B L, Monga V. Color error-diffusion halftoning[J]. IEEE Signal Processing Magazine, 2003, 20(4): 51-58.
[10] Damera-Venkata N, Evans B L. Design and analysis of vector color error diffusion halftoning systems[J]. IEEE transactions on image processing, 2001, 10(10): 1552-1565.
[11] MENG Xiaojie, ZENG Ping. A Vector Error Diffusion Half-toning Algorithm for Color Image. The research of computer application, 2005, 22(10):171-172.
[12] Kubota R, Tamukoh H, Kawano H, et al. A Color Quantization Based on Vector Error Diffusion and Particle Swarm Optimization Considering Human Visibility[C]//Pacific-Rim Symposium on Image and Video Technology. Springer, Cham, 2015: 332-343.
[13] Cyclone V Avalon-ST Interface for PCIe Solutions User Guide.(2017, May 24). Retrieved June 12, 2017, from Altera Corporation.
[14] Altera SoC Golden System Reference Design User Guide.(2017, May 24). Retrieved June 12, 2017, from Altera Corporation.
[15] Yang P, Wang Q, Zhang J. Parallel design and implementation of Error Diffusion Algorithm and IP core for FPGA[J]. Multimedia Tools and Applications, 2016, 75(8): 4723-4733.
[16] Mitsa T, Varkur K L. Evaluation of contrast sensitivity functions for the formulation of quality measures incorporated in halftoning algorithms[J]. 1993, 5(1-2):53-81.
[17] Mannos J, Sakrison D. The effects of a visual fidelity criterion of the encoding of images[J]. IEEE Transactions on Information Theory, 1974, 20(4):525-536.
[18] Wang Z, Bovik A C. Universal Image Quality Index[J]. IEEE Signal Processing Letters, 2002, 9(3):81-84.
[19] Poirson A B, Wandell B A. Appearance of colored patterns: pattern-color separability[J]. JOSA A, 1993, 10(12): 2458-2470.
[20] Zhang X, Wandell B A. A spatial extension of CIELAB for digital color-image reproduction[J]. Journal of the Society for Information Display, 1997, 5(1): 61-63.
[21] Monga V, Geisler W S, Evans B L. Linear color-separable human visual system models for vector error diffusion halftoning[J]. IEEE Signal processing letters, 2003, 10(4): 93-97. Poirson A B, Wandell B A. Appearance of colored patterns: pattern-color separability[J]. JOSA A, 1993, 10(12): 2458-2470.
[22] Venugopal, R. K., Heath, J. R., Lau, D. L. (2011, June). FPGA based parallel architecture implementation of Stacked Error Diffusion algorithm. In Application Specific Processors (SASP), 2011 IEEE 9th Symposium on. IEEE. Pages:66-69.