VLSI Implementation of an Adaptive Edge-Enhanced Color Interpolation Processor for Real-Time Video Applications

Shih-Lun Chen, *Member*, *IEEE*, and En-Di Ma

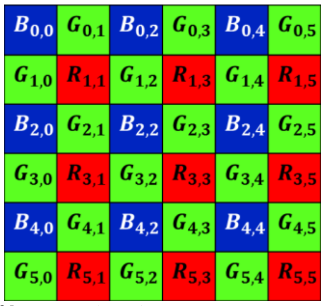
***Abstract*—In this paper, a low-complexity color interpolation algorithm is proposed for the VLSI implementation in real-time applications. The proposed novel algorithm consists of an edge detector, an anisotropic weighting model and a filter-based compensator. The anisotropic weighting model is designed to catch more information in horizontal than vertical directions. The filter-based compensation methodology includes a Laplacian and spatial sharpening filters which are developed to improve the edge information and reduce the blurring effect. In addition, the hardware cost was successfully reduced by hardware sharing and reconfigurable design techniques. The VLSI architecture of the proposed design achieves 200 MHz with 5.2 K gate counts, and its core area is 64,236 um^2 synthesized by a 0.18 um CMOS process. Compared with the previous low-complexity techniques, this work not only reduces gate counts or power consumption by more than 8 % or 91.7 %, respectively, but also improves the average CPSNR quality by more than 1.6 dB.**

***Index Terms*—Camera, charge-coupled device (CCD), Color filter array (CFA), color interpolation, demosacking, edge detector, Laplacian sharpening filter, and VLSI**

I. Introduction

RECENTLY, digital cameras are integrated into many consumer electronic products, such as digital camera, digital video, smart TV, smart phone, tablet PC, etc. The digital cameras are developed by a CCD or a CMOS image sensor that can capture images by color filter array (CFA) technique. The red (R), green (G), and blue (B) colors are sampled as one color in each pixel [1]. Fig. 1 shows a color filter arrays called Bayer CFA, in which two colors have disappeared in each pixel. Thus, it is important to reconstruct the images from CFA to full RGB formats.

Many efficient high-quality algorithms [2]-[18] have been proposed for reconstructing the full RGB color from CFA images. An adaptive color interpolation technique that used a 2-D locally stationary Gaussian process and an edge indicator was proposed by Chang *et al.* [2]. A spectral model for preserving spectral characteristics of refined CFA images was introduced by Lukac *et al.* [3]. A low-complexity interpolation method that used a simple image model was proposed by Pei *et al.* [4]. Moreover, Gunturk [5] and Li [6] presented high performance algorithms to refine the missing colors by using the correlation between the frequencies of the three colors components.


$B_{0,0}$ $G_{0,1}$ $B_{0,2}$ $G_{0,3}$ $B_{0,4}$ $G_{0,5}$
$G_{1,0}$ $R_{1,1}$ $G_{1,2}$ $R_{1,3}$ $G_{1,4}$ $R_{1,5}$
$B_{2,0}$ $G_{2,1}$ $B_{2,2}$ $G_{2,3}$ $B_{2,4}$ $G_{2,5}$
$G_{3,0}$ $R_{3,1}$ $G_{3,2}$ $R_{3,3}$ $G_{3,4}$ $R_{3,5}$
$B_{4,0}$ $G_{4,1}$ $B_{4,2}$ $G_{4,3}$ $B_{4,4}$ $G_{4,5}$
$G_{5,0}$ $R_{5,1}$ $G_{5,2}$ $R_{5,3}$ $G_{5,4}$ $R_{5,5}$
Fig. 1 Bayer color filter array (CFA).

Several efficient techniques such as selecting [7], [8] or fusing [9] the information of the vertical and horizontal directions has also been presented. A gradient-base scheme with a Gaussian low-pass filter to enhance the performance of the color interpolation was proposed by Yun *et al.* [10]. An effective scheme to enhance state-of-the-art demosaicking methods using image spatial and spectral correlation was presented by Chang *et al.* [11]. Su *et al.* [12] presented a wavelet-based classifier. This method produced a high-quality color interpolation. A high-quality full RGB color images from the mosaic sensor was adopted by Mmuresan *et al.* [13]. Alleysson [14] and Lian [15] *et al* reconstructed the full color images by filtering the luminance components.

An edge estimation method using variance of the different colors was invented by Chung *et al.* [16]. In addition, an iterative K-SVD based algorithm [17] successfully improved the quality of interpolated images through iteration technique. Prior knowledge was used to improve the performance of demosaick images by Menon *et al.* [18]. The high-quality color interpolation algorithms, [2]-[18], mentioned above made great contributions in CFA images correction. However, these high-quality color interpolation algorithms have the characteristics of high complexity and high memory requirement. Furthermore, these algorithms are not easy to be realized using VLSI technique.

For this reason, previous studies, [19]-[21], concerning the VLSI architecture of low-complexity and low-memory-requirement color interpolation algorithms were conducted. Doswald *et al.* [19] implemented a high speed image processor by VLSI technique. The throughput achieves a real-time process of 30 frames per second. Although this study proposed a high-quality and high-throughput color interpolation processor, the chip area and power consumption

Manuscript received July 5, 2013; revised December 26, 2013; accepted March 22, 2014. This work was supported by the National Science Council, Taiwan, under Grant numbers of NSC-101-2221-E-033-066, NSC-102-2221-E-033-063 and NSC-102-2622-E-033-009-CC2, and the National Chip Implementation Center, Taiwan. This paper was recommended by Associate Editor Toshiaki Fujii.

The authors are with the Department of Electronic Engineering, Chung Yuan Christian University, Chung Li City 320, Taiwan (ROC) (e-mail: chrischen@cycu.edu.tw).

of this design are quite few. Moreover, it demands a frame memory to buffer the input CFA image. An efficient color interpolation processor based on edge-direction weighting and local gain approach techniques was proposed by Hsia *et al.* [20]. The performance of this design was improved by a pipeline schedule and time-sharing techniques. Although the local gain that is obtained by the edge-direction weighting information, it can efficiently improve the quality of the interpolated images. It is necessary to use two division and three multiplication operations to obtain the edge-direction weighting and local gain information. For VLSI implementation, the chip area was greatly increased by realizing these dividers and multipliers due to the high complexity and hardware cost.

Hsia *et al.* [21] also presented a high-performance and low-complexity algorithms to develop a camera DSP system. Although this study achieved a better quality by using a white balance, a color space transformation, an auto gain control, an edge enhancement, and a color enhancement techniques, it is necessary to use the division and multiplication operations to find the gain and edge information. The hardware cost and chip area were efficiently increased due to the implementation of these dividers and multipliers. Recently, a low-cost and high-performance color demosaicking VLSI design was proposed by Hhiau *et al*. [22]. This design improved the quality of the interpolated images by using edge-information and inter-channel correlations. By using a pipeline architecture, the performance was greatly improved. Since the parameters of the edge-information and inter-channel correlations were divided by 2, 4, or 8, it is unnecessary to use any divider to realize this design. With this implementation technique, the chip area significantly reduced.

Regardless of the apparent advantages, such as high-quality, cost-effective, and low-memory-requirement, of the previous designs in literature [19]-[22], it is necessary to develop a higher performance and lower complexity color interpolation algorithms for VLSI implementation. Hence, a novel low-cost, high-quality, and low-memory-requirement adaptive edge-enhanced color interpolation processor is proposed in this paper. First, a register bank was added in the proposed interpolator to provide 15 CFA pixels in real-time for one green (G) and three red/blue (RB) interpolators processing in each cycle. This implementation requires a two line-buffer memory which is much less than a frame memory in [19]. Second, a low-complexity edge detector was created to enhance the edge information. It used only addition, subtraction, and absolute operations to obtain the edge information. The hardware cost of the edge detector is much less than the previous designs such as [20] and [21], which utilize dividers and multipliers to obtain the edge and gain information. Third, a novel anisotropic weighting model was designed for the proposed color interpolator. It can improve the quality of the interpolated image by acquiring more information from the horizontal direction than the vertical without adding line-buffer memory. Fourth, the proposed filter design can achieve a good quality of the interpolated images because it uses various colors of the original CFA pixels and double interpolated green pixels as elements of the filters rather than single color of original CFA pixels and single color pixel to compensate the interpolated pixel as presented in [20]-[22]

It consists of an edge detector to enhance the edge information in the images, an anisotropic weighting model to reduce the memory requirement, a filter-based RB compensator to improve the quality, and a register bank to process streaming data directly by using only a two-lime-buffer memory.

This paper is organized as follows: In Section II, the proposed novel color interpolation is presented. Section III describes the VLSI architecture of this work. Section IV shows the simulation results and chip implementation. Finally, in Section V, the conclusions are presented.

II. The Proposed Algorithm

The proposed novel color interpolation algorithm is composed of a low-complexity edge detection, a green color interpolation, and a red-blue color interpolation techniques. Each color is interpolated by different methodologies according to the relative locations and reference neighboring samples as shown in Fig. 2, in which the $BRg_{(i, j)}$ and $RBg_{(i, j)}$ represent that the green color pixel $g_{(i, j)}$ was interpolated and prepared when it interpolates $R_{(i, j)}$ and $B_{(i, j)}$. The details of each technique will be described in the following subsections.

A. Low-Complexity Edge Detection

In order to discover the edge information by low-complexity methodology, the difference in the vertical (DV) and horizontal (DH) directions were used. The DV neighboring around the pixel $G_{(i, j)}$ or $RB_{(i, j)}$ can be evaluated by

$$DV_{i,j} = \left|RB_{i-1,j-1} - RB_{i+1,j-1}\right| + \left|G_{i-1,j} - G_{i+1,j}\right| + \left|RB_{i-1,j+1} - RB_{i+1,j+1}\right| \quad (1)$$

where *G* is the pixel in green color and *RB* is the pixel in red or blue color in the CFA images. The relative locations and reference neighboring RGB pixels are shown in Fig. 2 (c). On the other hand, the difference in the horizontal direction (DH) neighboring around the pixel $G_{(i, j)}$ or $RB_{(i, j)}$ can be evaluated by

$$DH_{i,j} = \left|RB_{i+1,j+1} - RB_{i+1,j-1}\right| + \left|G_{i,j+1} - G_{i,j-1}\right| + \left|RB_{i-1,j+1} - RB_{i-1,j-1}\right| \quad (2)$$

After obtaining the difference in the horizontal (DH) and vertical (DV) directions, the total difference (TD) neighboring around the pixel $G_{(i, j)}$ or $RB_{(i, j)}$ can be calculated by

$$TD_{i,j} = DH_{i,j} + DV_{i,j} \quad (3)$$

where *DH* and *DV* are all positive value. The value of total difference (TD) provides a quantification to judge if the edge

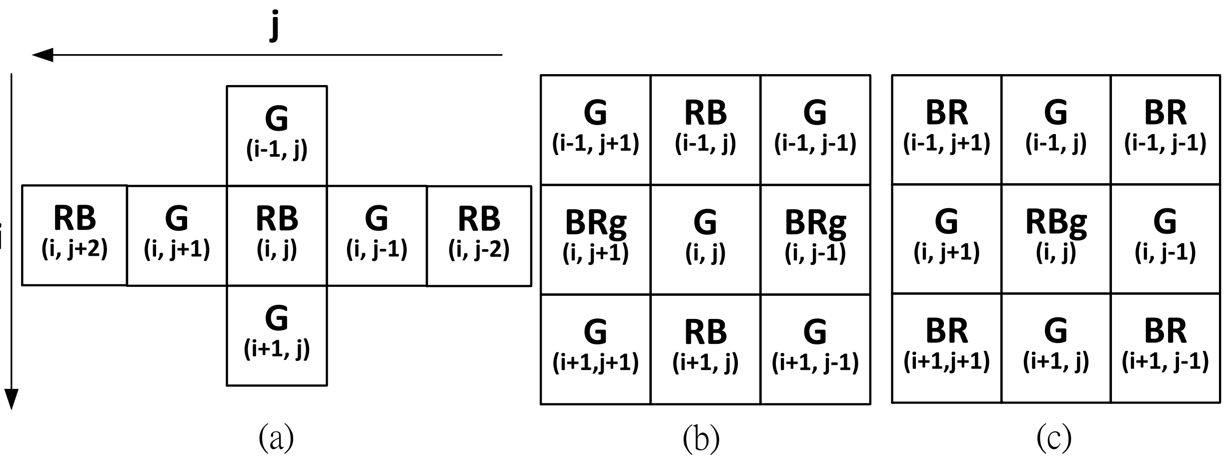


Fig. 2 (a) Relative locations and reference neighboring samples of $G_{(i,\ j)}$ in CFA format for green color interpolation. (b) Relative locations and reference neighboring samples of $G_{(i,\ j)}$ in CFA format for red or blue color interpolation. (c) Relative locations and reference neighboring samples of $R_{(i,\ j)}$ or $B_{(i,\ j)}$ in CFA format for green, blue or red color interpolation.

information is obvious or not obvious. It helps the G, R, and B colors interpolations to determine if it is necessary to enhance the edge features. Moreover, the values of DH and DV can be used to determine whether the edge information should be further enhanced in the horizontal or vertical directions.

B. Green Color Interpolation

In order to improve the quality of interpolated images, an anisotropic weighting model was created for this design. As shown in Fig. 2 (a), the reference neighboring pixels in horizontal direction are much more than those in vertical direction. The green color pixels can be interpolated by referring to 75 % weighting of pixels in the horizontal direction and 25 % weighting of pixels in the vertical direction. In order to reduce the computing resource of the proposed color interpolation algorithm, all division operations were replaced by shift operations.

In the same manner, to reconstruct the green color in CFA format image, there are two different causes for interpolating $G_{(i,\ j)}$. The location of $G_{(i,\ j)}$ could be the color pixel of $B_{(i,\ j)}$ or $R_{(i,\ j)}$ as shown in Fig. 2 (a). Hence, the interpolated pixel $G_{(i,\ j)}$ is described as $G^{(RB)}_{(i,\ j)}$ where the location of $P_{(i,\ j)}$ is $R_{(i,\ j)}$ or $B_{(i,\ j)}$ in the original CFA images, in which the $P_{(i,\ j)}$ represents the pixel at the location of i and j in the y and x coordinates, respectively. In addition, the values of TD, DH, and DV can be used to adaptively select one of the three green color interpolation models, without edge enhancement, edge enhancement in horizontal direction, and edge enhancement in vertical direction, as the interpolation model according to the edge information neighboring around the pixel $G_{(i,\ j)}$. If the value of the total difference (TD) is less than the threshold value, the value of $G^{(RB)}_{(i,\ j)}$ can be calculated by the without edge enhancement model as

$$G_{i,j}^{(RB)G} = \frac{3}{8}\left(G_{i,j-1} + G_{i,j+1}\right) + \frac{1}{8}\left(G_{i-1,j} + G_{i+1,j}\right) + RB_{i,j} - \frac{1}{2}\left(\frac{1}{2}\left(RB_{i,j} + RB_{i,j-2}\right) + \frac{1}{2}\left(RB_{i,j} + RB_{i,j+2}\right)\right) \quad (4)$$

where $G_{i,\ j}{}^{(RB)G}$ is the result of $G_{(i,\ j)}$ where the location of $P_{(i,\ j)}$ is $R_{(i,\ j)}$ or $B_{(i,\ j)}$. The $RB_{(i,\ j)}$ is the original sampled value of $P_{(i,\ j)}$ in the CFA image, which could be $R_{(i,\ j)}$ or $B_{(i,\ j)}$ depending on the location of $P_{(i,\ j)}$. Fig. 3 (a) shows the parameters of green color interpolation without edge enhancement model. Otherwise, if the value of the total difference (TD) is larger than the threshold value and DH is less than DV, the value of $G^{(RB)}_{(i,\ j)}$ can be obtained by the edge enhancement in horizontal direction model as

$$G_{i,j}^{(RB)G} = \frac{1}{2}\left(G_{i,j-1} + G_{i,j+1}\right) + RB_{i,j} - \frac{1}{2}\left(\frac{1}{2}\left(RB_{i,j} + RB_{i,j-2}\right) + \frac{1}{2}\left(RB_{i,j} + RB_{i,j+2}\right)\right) \quad (5)$$

Fig. 3 (b) shows the parameters of green color interpolation with edge enhancement in horizontal direction model. Also, if the value of total difference (TD) is larger than the threshold value and the value of DH is larger than DV, the value of $G^{(RB)}_{(i,\ j)}$ can be computed by the edge enhancement in vertical direction model as

$$G_{i,j}^{(RB)G} = \frac{1}{8}\left(G_{i,j-1} + G_{i,j+1}\right) + \frac{3}{8}\left(G_{i-1,j} + G_{i+1,j}\right) + \frac{1}{2}\left\{RB_{i,j} - \frac{1}{2}\left(\frac{1}{2}\left(RB_{i,j} + RB_{i,j-2}\right) + \frac{1}{2}\left(RB_{i,j} + RB_{i,j+2}\right)\right)\right\} \quad (6)$$

Fig. 3 (c) shows the parameters of green color interpolation with edge enhancement in vertical direction model. By analyzing the parameters of these three green color interpolation models, it is obviously that the characteristic of the compensation for green color is a spatial sharpening filter. This spatial sharpening filter can efficiently reduce the blurring effect. Moreover, the edge information can be enhanced efficiently by the proposed adaptive edge enhancement technique.

(a)

		1/8		
-1/4	3/8	1/2	3/8	-1/4
		1/8		

(b)

-1/4	1/2	1/2	1/2	-1/4

(c)

		3/8		
-1/8	1/8	1/4	1/8	-1/8
		3/8		

Fig. 3. The parameters of green color interpolation (a) without edge enhancement model (b) with edge enhancement in horizontal direction model (c) with edge enhancement in vertical direction model.

C. Red and Blue Colors Interpolation

To reconstruct the red and blue colors in CFA format image, it is important to use the information of the four neighboring green colors. Fig. 2 (a) shows the relative locations and reference neighboring samples of $G_{(i,\ j+1)}$, $G_{(i-1,\ j)}$, $G_{(i,\ j-1)}$, and $G_{(i+1,\ j)}$ where the pixel in CFA format is $R_{(i,\ j)}$ or $B_{(i,\ j)}$. To be able to enhance the edge information, the values of TD, DH, and DV can be used to adaptively select one of the three red and blue interpolation models, without edge enhancement, edge enhancement in horizontal direction, and edge enhancement in vertical direction, as the interpolation model according to the edge information neighboring around the pixel $R^{(B)}{}_{(i,\ j)}$ or $B^{(R)}{}_{(i,\ j)}$.

(a)

1/4	-1/4	1/4
-1/4	1	-1/4
1/4	-1/4	1/4

(b)

1/4	-3/8	1/4
-1/8	1	-1/8
1/4	-3/8	1/4

(c)

1/4	-1/8	1/4
-3/8	1	-3/8
1/4	-1/8	1/4

Fig. 4. The parameters of red and blue colors interpolation at $G_{(i,j)}$ (a) without edge enhancement model (b) with edge enhancement in horizontal direction model (c) with edge enhancement in vertical direction model.

If the value of total difference (TD) is less than a threshold value, the value of $R^{(B)}{}_{(i,\ j)}$ or $B^{(R)}{}_{(i,\ j)}$ can be calculated by the without edge enhancement model as

$$RB_{i,j}^{(BR)G} = \frac{1}{4}\left(RB_{i-1,j-1} + RB_{i-1,j+1} + RB_{i+1,j-1} + RB_{i+1,j+1}\right) + g_{i,j} - \frac{1}{4}\left(G_{i-1,j} + G_{i+1,j} + G_{i,j-1} + G_{i,j+1}\right) \quad (7)$$

where $g_{(i,\ j)}$ is produced by green color interpolation which is mentioned above. Fig. 4 (a) shows the parameters of red and blue color interpolation without edge enhancement model. Otherwise, if the value of TD is larger than the threshold value and DH is less than DV, then the value of $R^{(B)}{}_{(i,\ j)}$ or $B^{(R)}{}_{(i,\ j)}$ can be obtained by the edge enhancement in horizontal direction model as

$$RB_{i,j}^{(BR)G} = \frac{1}{4}\left(RB_{i-1,j-1} + RB_{i-1,j+1} + RB_{i+1,j-1} + RB_{i+1,j+1}\right) + g_{i,j} - \frac{1}{8}\left(3 * \left(G_{i-1,j} + G_{i+1,j}\right) + G_{i,j-1} + G_{i,j+1}\right) \quad (8)$$

where $g_{(i,\ j)}$ is produced by green color interpolation. Fig. 4 (b) shows the parameters of red and blue colors interpolation with edge enhancement in horizontal direction model. Also, if the value of TD is larger than the threshold value and DH is larger than DV, then the value of $R^{(B)}{}_{(i,\ j)}$ or $B^{(R)}{}_{(i,\ j)}$ can be computed by the edge enhancement in vertical direction model as

$$RB_{i,j}^{(BR)G} = \frac{1}{4}\left(RB_{i-1,j-1} + RB_{i-1,j+1} + RB_{i+1,j-1} + RB_{i+1,j+1}\right) + g_{i,j} - \frac{1}{8}\left(G_{i-1,j} + G_{i+1,j} + 3 * \left(G_{i,j-1} + G_{i,j+1}\right)\right) \quad (9)$$

Fig. 4 (c) shows the parameters of red and blue colors interpolation with edge enhancement in vertical direction model. By analyzing the parameters of these interpolation models having three R and B colors, it can be seen that the characteristics of the compensation for R and B colors are Laplacian filters [24].

(a)

-1/8	1/2	-1/8
0	1/2	0
-1/8	1/2	-1/8

(b)

1/2BR -1/2g	1	1/2BR -1/2g

Fig. 5. The parameters of the red and blue colors interpolation at $G_{(i,\ j)}$ (a) Laplacian sharpening filter (b) spatial sharpening filter

Fig. 2 (b) shows the relative locations and reference neighboring samples of $RBg_{(i,\ j+1)}$, $RB_{(i-1,\ j)}$, $BR_{(i,\ j-1)}$, and $RB_{(i+1,\ j)}$ where the pixel in the CFA format is $G_{(i,\ j)}$. If the upper and bottom pixels of $G_{(i,\ j)}$ in the CFA format are $R_{(i-1,\ j)}$ and $R_{(i+1,\ j)}$, the $R_{(i,\ j)}$ can be obtained by the Laplacian sharpening filter, as shown in Fig. 5 (a). Otherwise, the Laplacian sharpening filter is used to produce $B_{(i,\ j)}$ where the upper and bottom pixels of $G_{(i,\ j)}$ in the CFA format are $B_{(i-1,\ j)}$ and $B_{(i+1,\ j)}$. The red or blue color reconstructed pixel $RB_{(i,\ j)}$ where its upper and bottom pixels are $RB_{(i-1,\ j)}$ and $RB_{(i+1,\ j)}$ at $G_{(i,\ j)}$ can be calculated as

$$RB_{i,j}^{(G)BR} = \frac{1}{2}\left(RB_{i-1,j} + RB_{i+1,j}\right) + \frac{1}{2}G_{i,j} - \frac{1}{8}\left(G_{i-1,j-1}\right. G_{i-1,j+1} + G_{i+1,j-1} + G_{i+1,j+1}\left.\right) \quad (10)$$

where $RB^{(G)B}{}_{i,j}$ is the result of $RB_{(i,\ j)}$ where the location of $P_{(i,\ j)}$ is $G_{(i,\ j)}$. Furthermore, if the right and left pixels of $G_{(i,\ j)}$ in the CFA format are $R_{(i,\ j-1)}$ and $R_{(i,\ j+1)}$, the $R_{(i,\ j)}$ can be obtained by the spatial sharpening filter, as shown in Fig. 5 (b). Otherwise, the spatial sharpening filter is used to produce $B_{(i,\ j)}$ where the right and left pixels of $G_{(i,\ j)}$ in the CFA format are $B_{(i,\ j-1)}$ and

$B_{(i,\,j+1)}$. The blue or red color reconstructed pixel $BR_{(i,\,j)}$ where its right and left pixels are $BR_{(i,\,j-1)}$ and $BR_{(i,\,j+1)}$ at $G_{(i,\,j)}$ can be computed as

$$BR_{i,j}^{(G)BR} = \frac{1}{2}\left(BR_{i,j-1} + BR_{i,j+1}\right) + G_{i,j} - \frac{1}{2}\left(g_{i,j-1} + g_{i,j+1}\right) \quad (11)$$

where $BR_{i,\,j}^{(G)BR}$ is the result of $BR_{(i,\,j)}$ where the location of $P_{(i,\,j)}$ is $G_{(i,\,j)}$.

The parameters of the proposed G, R, and B interpolation equations are designed as 1/2, 1/4, 1/8, and 3/8. It provides a cost-efficient base for VLSI implementation by replacing the multipliers and dividers with the shifters and adders.

III. VLSI Architecture

Fig. 6 shows the block diagram of the VLSI architecture for the proposed color interpolation processor. It consists of seven main blocks: a register bank, an edge detector, a green color interpolator (G interpolator), a red and blue colors interpolator model 1 (RB_M1 interpolator), red and blue colors interpolator model 2 (RB_M2 interpolator), a red and blue colors interpolator model 3 (RB_M3 interpolator), and a controller. The details of each part will be described in the following subsections.

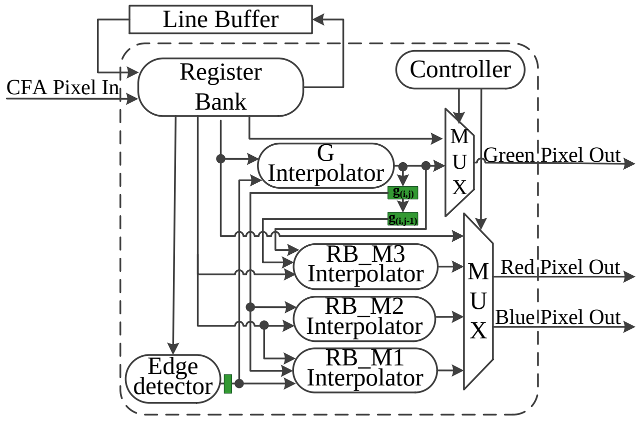


Fig. 6. Block diagram of the VLSI architecture for the proposed color interpolation processor.

A. Register Bank

The register bank was designed to real-time provide fifteen pixels in CFA format for processing the G interpolator and three RB interpolators during each cycle. Fig. 7 shows the architecture of the register bank. It is designed with a two-line-buffer memory and constructed with fifteen shift registers. The proposed register bank is designed such that

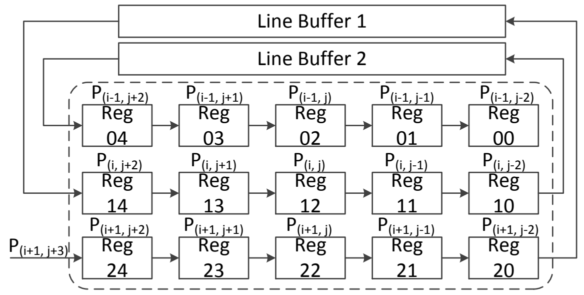


Fig. 7. Architecture of the register bank.

only one value of pixel from memory is received in each cycle time, and then provides fifteen values of CFA pixels as inputs for color interpolation. By adding this register bank, the proposed color interpolation processor achieves the memory access through pixel in and pixel out.

B. Edge Detector

Fig. 8 shows the architecture of the proposed edge detector. It consists of six absolute subtractors (|Sub|) and five adders (Add).The eight input signals receive their inputs from the register bank. After being processed by the edge detector, the values of TD, DH, and DV at the position of $P_{(i,\,j)}$ are produced for the green color interpolator and the first model of red and blue color interpolator (RB_M1). In addition, the output signals of the edge detector are also sent to the controller. The output signal of TD provides information of the edge intensity. The other two output signals, DH and DV, give the direction information of the edges.

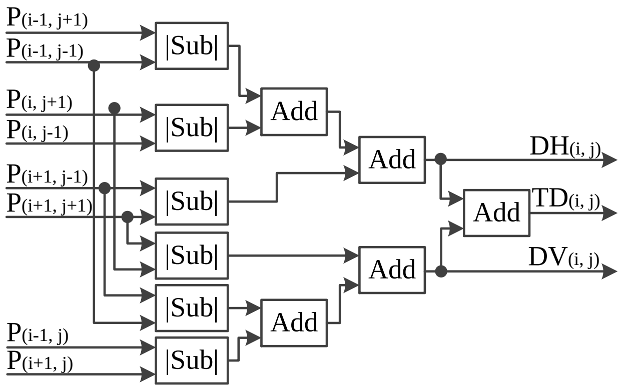


Fig. 8. Architecture of the edge detector.

C. Green Color Interpolator (G Interpolator)

To analyze equations (4), (5) and (6), it is obvious that all the input signals are the same except the values of the parameters. In this paper, a reconfigurable technique was used to design the hardware architecture of the green color interpolator. Fig. 9 shows the architecture of the reconfigurable green color interpolator design. It consists of eight adders, one subtractor, four multiplexers, and five shifters. This design can be reconfigured as the functions of Eq. (4), (5) or (6) by multiplexer selection during each processing cycle. The control signals, which are sent to multiplexers, are produced by the controller according to the values of TD, DH, and DV. By using reconfigurable technique, the proposed green color interpolator has the characteristics of low cost, high flexibility and high performance.

In order to shorten the critical path and improve the design performance, three registers were added in this architecture. The first register is used to store the result of the G interpolator $g_{(i,\,j)}$. It provides input signal $g_{(i,\,j)}$ for the RB_M1 and RB_M2 modules. The second register is used to store the value of $g_{(i,\,j)}$ that provides the input signal $g_{(i,\,j-1)}$ for the RB_M3 module. Lastly, the third register is used to store the edge information of the TD, DH, and DV. By adding these three registers, this implementation is a pipeline architecture design. It achieves shortening the critical path and providing the interpolated green pixels of $g_{(i,\,j)}$ and $g_{(i,\,j-1)}$ for R and B interpolators to improve the quality of the interpolated $R_{(i,\,j)}$ and $B_{(i,\,j)}$ pixels.

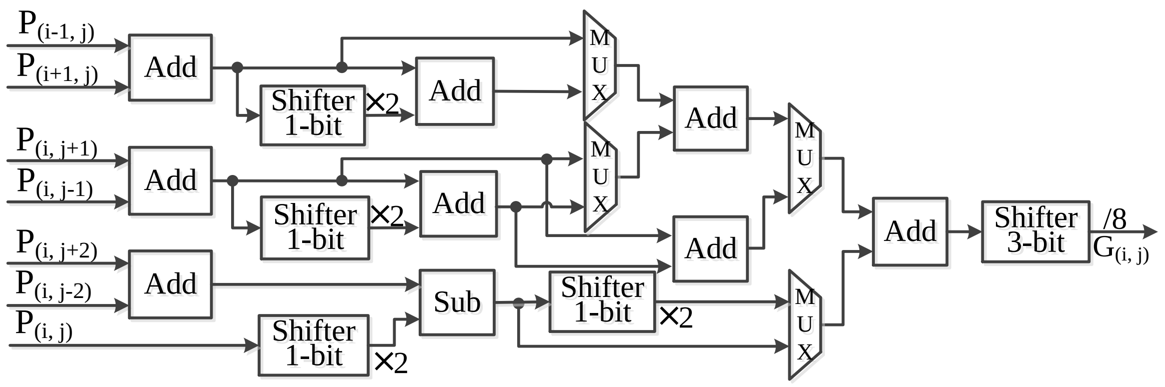


Fig. 9. Architecture of the green color interpolator (G interpolator)

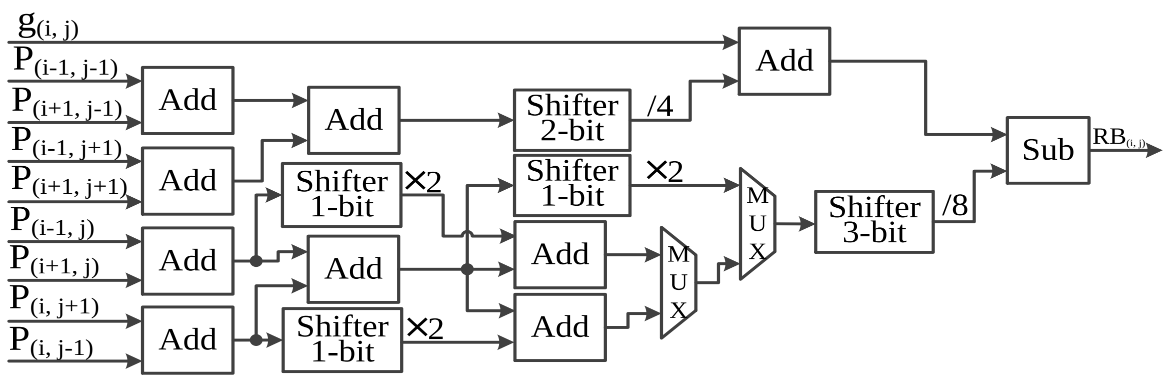


Fig. 10. Architecture of the red and blue colors interpolator model 1 (RB_M1 interpolator)

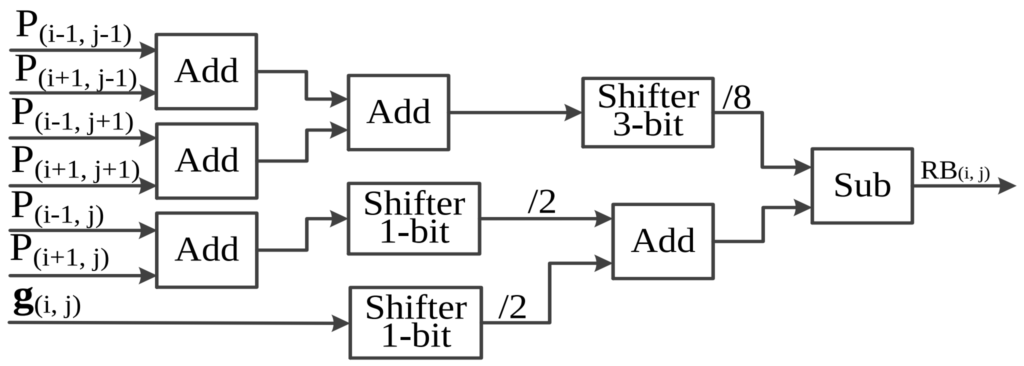


Fig. 11. Architecture of the red and blue colors interpolator model 2 (RB_M2 interpolator)

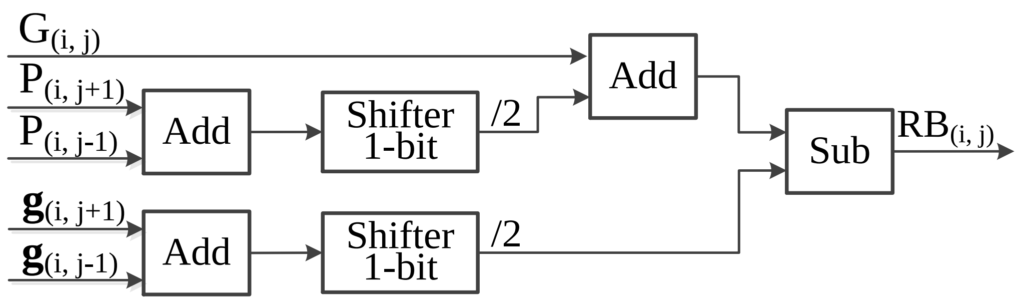


Fig. 12. Architecture of the red and blue colors interpolator model 3 (RB_M3 interpolator)

Since the result of the edge detector is stored in a register, the critical path in this design begins from the input of the G interpolator and ends in the output of the RB_M3. The critical path in this design has five adders, three shifters, and one substractor with only 5-ns and achieves an operating frequency of 200 MHz in TSMC 0.18-μm process.

D. Red and Blue Colors Interpolators

To analyze equations (7), (8), and (9), all input signals should be the same except the values of their parameters. Hence, the reconfigurable technique can be used to design the hardware architecture of the red and blue colors interpolator. The RB_M1 interpolator is shown in Fig. 10, which consists of nine adders, one subtractor, two multiplexers, and five shifters. This design can be reconfigured as the functions of Eq. (7), (8), or (9) according to the values of TD, DH, and DV during each processing cycle. Fig. 11 shows the VLSI architecture of the RB_M2 interpolator, which can be implemented from Eq. (10). The RB_M2 interpolator consists of five adders, one subtractor, and two shifters. Fig. 12 illustrates the VLSI architecture of the RB_M3 interpolator, which can be implemented using the Eq. (11). RB_M3 interpolator performs like RB_M2. It consists of three adders, one subtractor, and one shifter and has the lowest cost module among the four interpolators.

E. Controller

The controller is implemented by a finite state machine (FSM) sequential circuit. It provides control signals to the multiplexer for selecting input data for the interpolators and is capable of sending reconfigurable control signals for changing the architecture of the interpolators. Moreover, the controller must monitor its input and output data access with the memory to fit the performance of pixel-in and pixel-out. Finally, the proposed color interpolation processor achieves high performance and high throughput.

IV. Simulation Results and Chip Implementation

In order to compare the performance of the previous low-complexity color interpolation algorithms with this work, the Matlab (Mathworks, Natick, MA) tool was used to compute the CPSNR and S-CIELAB $\triangle E^*$ [23] values by the original golden images and the interpolated images. Fig. 13 shows the 24 photographs of Kodak data-set which are selected as benchmark images at the resolution of 768×512 with 24 bit-per-pixel. Initially, each testing image was processed as a Bayer color filter array [1] to create the CFA test images. Second, each CFA image was refined into full colors of 768×512 pixels for R, G, and B by the previous low-complexity algorithms bilinear [25], LHCI [20], CDSP

Fig. 13. Twenty-four reference images for testing.

Table I
Comparisons of CPSNR values for the previous low-complexity color interpolation algorithms with this work

CPSNR	Bilinear [25]	LHCI [20]	CDSP [21]	ACDS [22]	This Work A	This Work B
K01	25.964	27.460	28.126	29.137	30.415	31.120
K02	31.908	30.359	33.974	34.890	35.471	35.754
K03	32.808	33.927	35.273	36.639	37.380	37.868
K04	32.613	31.361	36.010	35.826	36.704	37.334
K05	26.354	27.703	30.067	30.270	31.860	32.290
K06	27.163	28.548	29.279	30.516	31.760	32.460
K07	32.398	34.110	35.028	36.248	37.044	37.399
K08	23.466	25.381	26.236	26.735	27.476	28.518
K09	31.672	33.404	34.334	34.665	35.776	36.643
K10	31.568	33.610	34.951	34.920	36.085	36.909
K11	28.762	30.064	31.400	32.084	33.246	33.669
K12	32.159	34.318	34.116	35.502	36.392	37.020
K13	23.762	25.659	27.285	27.166	28.790	28.806
K14	28.536	29.436	31.278	32.059	33.057	33.396
K15	30.645	31.050	33.514	34.001	34.446	35.170
K16	30.325	31.289	32.060	33.758	34.862	35.400
K17	31.991	33.349	35.164	35.218	36.575	36.819
K18	27.867	29.343	31.435	30.944	32.439	32.291
K19	28.095	29.858	30.466	31.211	32.243	33.240
K20	29.972	31.542	32.732	33.025	34.538	34.891
K21	28.169	29.635	30.735	31.555	32.905	33.274
K22	30.224	31.712	32.938	33.304	34.310	34.824
K23	34.104	35.427	36.486	37.756	38.475	38.749
K24	26.581	28.486	30.059	29.747	31.337	31.206
Average	29.463	30.710	32.206	32.799	33.899	34.377

Note: This work A: without using edge-enhancement technique B: using edge-enhancement technique.

TABLE II
COMPARISONS OF S-CIELAB △E* FOR THE PREVIOUS LOW-COMPLEXITY COLOR INTERPOLATION ALGORITHMS WITH THIS WORK

S-CIELAB △ E*	Bilinear [25]	LHCI [20]	CDSP [21]	ACDS [22]	This Work A	This Work B
K01	15.4764	13.1446	12.1088	10.8331	9.5175	8.9043
K02	6.4805	7.9532	5.3123	4.9451	4.6994	4.5856
K03	4.9751	4.3892	3.7864	3.4266	3.1181	3.0051
K04	6.1287	6.9409	4.5557	4.4459	4.0854	3.9645
K05	13.3686	11.3626	9.0921	8.8395	7.4711	7.1299
K06	12.0246	10.0339	9.5108	8.0150	7.1670	6.6884
K07	5.3952	4.6653	4.2969	3.8035	3.5033	3.3687
K08	19.5483	15.9282	14.7800	13.5983	12.5724	11.2538
K09	6.2792	5.3728	4.9915	4.6626	4.2149	3.9750
K10	6.1834	5.2363	4.7885	4.5724	4.0740	3.8636
K11	9.5263	8.2366	7.3747	6.6437	5.9573	5.6879
K12	5.9777	5.0373	4.8916	4.2640	3.8805	3.6944
K13	19.6867	15.8251	13.5906	13.3518	11.4258	11.3018
K14	10.3701	9.1111	7.7200	7.0371	6.2689	6.0670
K15	6.6185	6.5566	4.9304	4.9197	4.5566	4.3887
K16	7.8501	6.7899	6.4098	5.3069	4.7840	4.5423
K17	6.0876	5.2335	4.5205	4.3854	3.8941	3.7777
K18	10.8607	9.2299	7.6817	7.7844	6.6812	6.6660
K19	10.1343	8.3779	7.7126	7.2201	6.5083	5.9883
K20	6.3236	5.2449	4.8413	4.6193	4.2124	4.0569
K21	9.9582	8.4423	7.5433	6.8475	6.0741	5.8542
K22	8.2565	7.0838	6.2805	6.0822	5.4286	5.2807
K23	4.1159	3.8566	3.2751	3.1023	2.9039	2.8534
K24	11.3546	9.2173	8.1010	7.9729	6.8588	6.7851
Average	9.2909	8.0529	7.0040	6.5283	5.8274	5.5701

[21], ACDS [22], and the two algorithms of this work. Finally, the CPSNR and S-CIELAB △E* values can be obtained from the original golden and the interpolated images.

Table I lists the CPSNR values for each image of the twenty-four testing images interpolated by the previous low-complexity algorithms, and the proposed two algorithms(i.e. with and without edge-enhancement technique). Similarly, table II lists the S-CIELAB △E* results. The experimental results show that this work achieves better performance in terms of CPSNR and S-CIELAB △E* values regardless of the implementation of edge-enhancement technique. To compare the CPSNR and S-CIELAB △E* results in this work A and B, we can easily find that the edge-enhancement technique can improve the CPSNR by 0.478 dB and decrease the S-CIELAB △E* by 0.2573. Fig. 14 shows the first, second, third, fourth, and fifth column show the image results of original, LHCI [20], CDSP [21], ACDS [22] and this work B. The experimental results show that this work achieves better quality than the previous low-complexity algorithms.

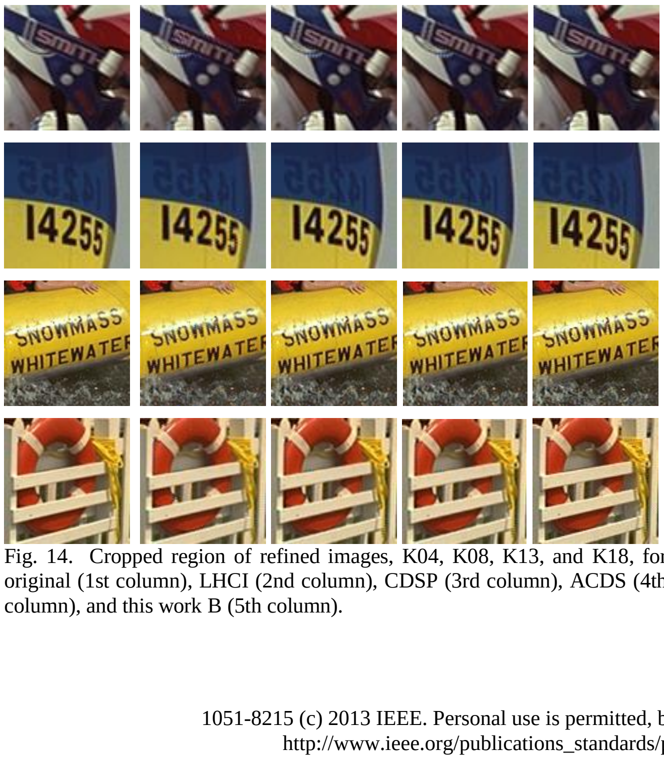


Fig. 14. Cropped region of refined images, K04, K08, K13, and K18, for original (1st column), LHCI (2nd column), CDSP (3rd column), ACDS (4th column), and this work B (5th column).

The proposed color interpolation processor was implemented using Verilog hardware description language (HDL) and synthesized using the electronic design automation (EDA) tool Design Vision based on TSMC 0.18 μm process standard cells. The layout was generated by auto placement and routing tool IC Compiler, in which the width is 251.86 μm and length is 255.05 μm. Furthermore, this design was also compiled by Quartus II, and evaluated using a FPGA emulation board with Altera FPGA EP2C70F896C6 core.

As shown in Table III, the two architectures of this work contain 3.6 K and 5.2 K gate counts and the chip areas are 45,158 μm^2 and 66,590 μm^2, respectively, synthesized in TSMC 0.18 μm CMOS process. The critical paths of these two architectures are 5 ns, which results in a clock frequency up to 200 MHz. The power consumptions of these two architectures are 3.42 mW and 4.66 mW, respectively, at 200 MHz with 1.8 V supply voltage. Furthermore, this work achieves 200 megapixels throughput in each color (i.e. R, G, and B) per second. This design is fast enough for real-time processing CFA image with a high definition (HD) of 1920×1080 resolution at 30 frames per second. The characteristics of high performance and throughput provide a well base for developing a real-time video system.

Table III also lists the comparisons of previous low complexity color interpolation designs and this work. It illustrates the CPSNR value, process, gate counts, operating frequency, power, core area, line-buffer memory, memory

TABLE III
COMPARISONS OF PREVIOUS LOW COMPLEXITY COLOR INTERPOLATION DESIGNS WITH THIS WORK

	LHCI [20]	CDSP [21]	ACDS [22]	This Work	
				A	B
CPSNR	30.7	32.2	32.8	33.9	34.4
Process	0.35 μm	0.35 μm	0.18 μm	0.18 μm	0.18μm
Gate Counts (excluding memory)	10 K	26 K	5.6 K	3.6 K	5.2K
Frequency	40 MHz	50 MHz	200 MHz	200 MHz	200 MHz
Power	200 mW	56 mW	N/A	3.42 mW	4.66 mW
Core Area (μm^2)	7 M	5 M	670 K	45.2 K	64.2 K
Memory (bit)	2 lines (16 Kbit)	1 frame (5.9 Mbit)	2 lines (30 Kbit)	2 lines (30 Kbit)	2 lines (30 Kbit)
Memory Area (μm^2)	78.4 K	15.8 M	20.7 K	20.7 K	20.7 K
Throughput (pixel/s)	40 M	50 M	200 M	200 M	200 M
Quality	VGA (1024×760)	VGA (1024×760)	HD (1920×1080)	HD (1920×1080)	HD (1920×1080)
Normalized Core Area	1.92	5	1.08	0.69	1

Note: The normalized core area is normalized by the NAND-equivalent gate counts

area, throughput, normalized core area, and normalized area of the two previous low complexity implementations and the two architectures presented in this work. The previous designs were implemented in 0.35 μm CMOS process. To be able to quantify the hardware cost objectively, the NAND-equivalent gate count was selected as a comparison standard for normalized area. The gate count (excluding the line buffer) in this work B is 5.2 k, which achieves reduction of at least 48 %, 80 %, or 8% than the previous design LHCI [20], CDSP [21], or ACDS [22].

As mentioned above, the proposed color interpolation processor design reduces at least 8 % gate counts, saves at least 91.7 % power consumption, and improves 1.6 dB averaging quality than the previous low-complexity designs. Comparing the two architectures in this work, the design B costs 31 % core area more than the design A due to an extra edge detector and some combinational circuit of the controller. However, it improves the quality by over 0.5 dB.

V. Conclusion

In this paper, a novel color interpolation algorithm is proposed to develop a low-cost, low-power, high performance, and high quality color interpolation processor for real-time video applications. An anisotropic weighting model, an edge detector, Laplacian and sharpening filters have been used to reduce the memory requirement and improve the quality of the images.

References

[1] B. E. Bayer, “Color Imaging Array,” U.S. patent 3 971 065, Jul. 1976.

[2] H. A. Chang, and H. H. Chen, “Stochastic color interpolation for digital cameras,” *IEEE Transaction on Circuits and Systems for Video Technology*, Vol. 17, no. 8, pp. 964-973, Aug. 2007.

[3] R. Lukac, K. N. Plataniotis, and D. H.atzinakos, “Color image zooming on the Bayer pattern,” *IEEE Transaction on Circuits and Systems for Video Technology*, Vol. 15, no. 11, pp. 1475-1492, Nov. 2005.

[4] S. C. Pei, and I. K. Tam, “Effective color interpolation in CCD color filter arrays using signal correlation,” *IEEE Transaction on Circuits and Systems for Video Technology*, Vol. 13, no. 6, pp. 503-513, Jun. 2003.

[5] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, “Color plane interpolation using alternating projections,” *IEEE Trans. Image Process.*, vol. 11, no. 9, pp. 997–1013, Sep. 2002.

[6] X. Li, “Demosaicing by successive approximation,” *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 370–379, Mar. 2005.

[7] K. Hirakawa and T. W. Parks, “Adaptive homogeneity-directed demosaicing algorithm,” *IEEE Trans. Image Process.*,vol. 14, no. 3, pp. 360–369, Mar. 2005.

[8] D. Menon, S. Andriani, and G. Calvagno, “Demosaicing with directional filtering and a posteriori decision,” *IEEE Trans. Image Process.*, vol. 16, no. 1, pp. 132–141, Jan. 2007.

[9] L. Zhang and X. Wu, “Color demosaicking via directional linear minimum mean square-error estimation,” *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2167–2177, Dec. 2005.

[10] S. H. Yun, J. H. Kim, and S. Kim, “Color interpolation by expanding a gradient method,” *IEEE Trans. Consumer Electronics*, vol. 54, no. 4, pp. 1531–1539, Nov. 2008.

[11] L. Chang, and Y. P. Tan, “Effective use of spatial and spectral correlations for color filter array demosaicking,” *IEEE Trans. Consumer Electronics*, vol. 50, no. 1, pp. 355–365, Feb. 2004.

[12] C. Y. Su, and Y. S. Lin, “Colour interpolation using wavelet-based classifiers,” *Electronics Letters*, vol. 43, no. 12 pp. 667–669, June 2007.

[13] D. D. Mmuresan and T. W. Parks, “Demosaicing using optimal recovery,” *IEEE Trans. Image Process.*, vol. 14, no. 2, pp. 267-278, Feb. 2005.

[14] D. Alleysson, S. Süsstrunk, and J. Hérault, “Linear demosaicing inspired by the human visual system,” *IEEE Trans. Image Process.*, vol. 14, no. 4, pp. 439–449, Apr. 2005.

[15] N. X. Lian, L. Chang, Y. P. Tan, and V. Zagorodnov, “Adaptive filtering for color filter array demosaicking,” *IEEE Trans. Image Process.*, vol. 16, no. 10, pp. 2515–2525, Oct. 2007.

[16] K.-H. Chung and Y.-H. Chan, “Color demosaicing using variance of color differences,” *IEEE Trans. Image Process.*, vol. 15, no. 10, pp. 2944–2955, Oct. 2006.

[17] J. Mairal, M. Elad, and G. Sapiro, “Sparse representation for color image restoration,” *IEEE Trans. Image Process.*, vol. 17, no. 1, pp. 53–69, Jan. 2008.

[18] D. Menon and G. Calvagno, “Regularization Approaches to Demosaicking,” *IEEE Trans. Image Process.*, vol. 18, no. 10, pp. 2209 - 2220, Oct. 2009.

[19] D. Doswald, J. Hafliger, P. Blessing, N. Felber, P. Niederer, and W. Fichtner, “A 30-frames/s megapixel real-time CMOS image processor,” *IEEE Journal of Solid-State Circuits*, Vol. 35, no. 11, pp. 1732-1743, Nov. 2000.

[20] S. C. Hsia, M. H. Chen, and P. S. Tsai, “VLSI implementation of low-power high-quality color interpolation processor for CCD camera,” *IEEE Transaction on Very Large Scale Integration (VLSI) Systems*, Vol. 14, no. 4, pp. 361-369, Apr. 2006.

[21] S. C. Hsia, and P. S. Tsai, “VLSI implementation of camera digital signal processor for document projection system,” *in Proc. IEEE Int. Conf. Signal Processing System (ICSPS)*, Jul. 2010, pp. 657–660.

[22] Y. H. Shiau, P. Y. Chen, and C. W. Chang, “An area-efficient color demosaicking scheme for VLSI architecture,” *International Journal of Innovative Computing, Information and Control*, Vol.7, No.4, pp.1739-1752, Apr. 2011.

[23] K. L. Chung, W. J. Yang, W. M. Yan, and C. C. Wang, “Demosaicing of color filter array captured images using gradient edge detection masks and adaptive heterogeneity-projection,” *IEEE Trans. Image Process*, vol. 17, no. 12, pp. 2356–2367, Dec. 2008.

[24] J. Dubois, D. Ginhac, M. Paindavoine, and B. Heyrman, “A 10000 fps CMOS sensor with massively parallel image processing,” *IEEE Journal of Solid-State Circuits*, Vol. 43, no. 3, pp. 706-717, Mar. 2008.

[25] K. Jensen and D. Anastassiou, “Subpixel edge localization and the interpolation of still images,” *IEEE Trans. Image Process.*, vol. 4, no. 3, pp. 285–295, Mar. 1995.