

# Design Methodology for Low Power Dissipation and Parametric Robustness through Output Quality Modulation: Application to Color Interpolation Filtering

N. Banerjee, G. Karakonstantis, J. H. Choi, C. Chakrabarti *Senior Member* and K. Roy *Fellow, IEEE*

**Abstract**—Power dissipation and robustness to process variation have conflicting design requirements. Scaling of voltage is associated with larger variations, while Vdd up-scaling or transistor up-sizing for parametric delay variation tolerance can be detrimental for power dissipation. However, for a class of signal processing systems effective trade-off can be achieved between Vdd scaling, variation tolerance and “output quality”. In this paper, we develop a novel low-power variation tolerant algorithm/architecture for color interpolation that allows a graceful degradation in the PSNR (Peak Signal to Noise Ratio) under aggressive voltage scaling as well as extreme process variations. This feature is achieved by exploiting the fact that all computations used in interpolating the pixel values do not equally contribute to PSNR improvement. In presence of Vdd-scaling and process variations, the architecture ensures that only the “less important computations” are affected by delay failures. We also propose a different sliding window size than the conventional one to improve interpolation performance by a factor of two with negligible overhead. Simulation results show that even at a scaled voltage of 77% of nominal value, our design provides reasonable image PSNR with 40% power savings. We also emulated the color interpolation design on an ALTERA DE2 board and observed the efficacy of our methodology with voltage scaling.

**Index Terms**—Low power design, Quality-Vdd tradeoffs, Color Interpolation, Process Variation Tolerance.

## I. INTRODUCTION

Present-day integrated circuits are expected to deliver high-quality/high-performance under ever-diminishing power budgets. Power-aware designs are necessary to prolong the battery lifetime of portable devices, to prevent excessive heat generation, which might result in device reliability problems and to reduce the cost associated with expensive cooling techniques. Due to quadratic dependence of power on voltage, supply voltage scaling has been investigated as an effective method [1, 2] to reduce power. However, supply scaling increases the delays in all computation paths and can result in incorrect or incomplete computation of certain paths (since delay is inversely

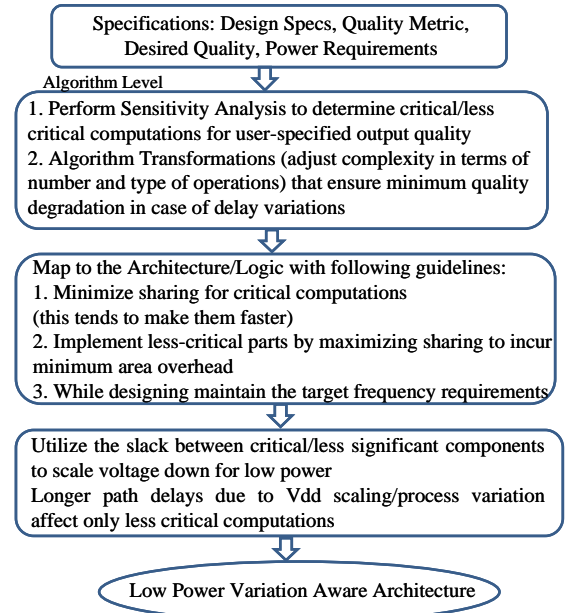


Fig. 1: The design methodology

proportional to voltage as a first order approximation). Therefore, applying supply voltage scaling randomly to any architecture can adversely affect the output, leading to lower manufacturing/parametric yield.

Besides power dissipation, process variations also pose major design concern with technology scaling. Studies have shown [3, 4, 5] that parameter variations create a delay spread of almost 30% for 70 nm process technology, leading to delay failures in some chips. Supply voltage can be scaled-up or logic gates can be up-sized to prevent delay failures and to achieve higher parametric yield. However, such techniques come at a cost of increased power and/or die area. Hence, meeting the contradictory requirements of high yield, low power and high quality is becoming exceedingly challenging in nanometer designs. Therefore, there is a need for scalable designs, where lower power and process variation tolerance can be achieved with minimal output quality degradation. Interestingly, we observed that for several DSP applications, all computations are not equally important in shaping the output response. For such systems, some computations are critical for determining the output quality, while others play a less important role. This

information can be exploited to provide the “right” trade-off between output quality, energy consumption (supply scaling) and parametric yield due to process variations.

The design approach advocated in this paper is shown in Fig. 1. After obtaining user specifications and deciding the appropriate algorithm, a sensitivity analysis determines the important/less-important components (important being those which contribute significantly to the output quality, less-important being the ones that affect the output quality to a lesser extent). Of course, this classification is only possible based on the quality requirements of the user. In the next step, the architecture is developed so that the critical components share minimal resources (this also tends to make them significantly faster). However, this comes at an increased cost of area. To circumvent this area overhead, we maximize sharing among the less-critical sections of the design, while maintaining the target frequency requirements. This maximal sharing for less-critical sections and minimal sharing for the critical sections of the architecture allows a delay slack to be available within these classified sections. This ensures delay discrepancies at nominal voltage/scaled voltages due to parametric variations or voltage scaling do not have any impact on the “important” computations. One other important feature of this design methodology is that we maintain the same operating frequency as that of nominal Vdd at scaled voltages. The resultant increase in delays affects only the less-important computations and has negligible effect on output quality. In this paper, we apply this design principle to color interpolation architecture.

Color interpolation is the most computationally demanding operation in the image color processing stage (Fig. 2). Since each pixel of a color image consists of 3 primary color components, red (R), green (G) and blue (B), imaging systems require 3 image sensors in order to capture a scene. However, most image sensors are housed in portable devices, whose lifetimes are limited by the battery lifetimes. To alleviate power requirements as well as reduce the manufacturing cost it is customary for such devices to capture an image with a single sensor. In this case, only one of the three primary colors is sampled at each pixel and subsequently stored in color filter arrays (CFA). In order to reconstruct a full color image, the missing color samples at each pixel need to be interpolated by a process called color interpolation. Also, since the color processing stage output directly impacts the post-processing stage results (Fig. 2), it is imperative that a reasonably high Peak Signal to Noise Ratio (PSNR) be maintained for interpolation stage output. The interpolation architecture, therefore, should simultaneously satisfy both low power and high PSNR. Though extensive research [4, 5, 6] has been conducted at the algorithmic level to improve the PSNR of color interpolation, very little attention has been given to generate low power designs with high parametric yield for the same. However, by

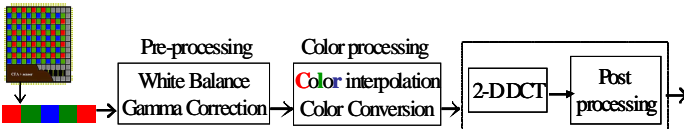


Fig. 2: A typical image processing chain for a digital camera

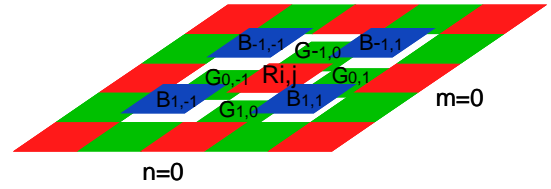


Fig. 3: Bayer CFA pattern

allowing “intelligent” tradeoffs between image quality and energy consumption, it is possible to develop energy-aware interpolation with high image PSNR. Our contributions in this paper can be summarized as follows:

- 1) Identify crucial/less-crucial computations for maintaining high output image PSNR in color interpolation.
- 2) Present an algorithm/architecture for color interpolation where graceful degradation in PSNR with Vdd-scaling (low power) under process variations can be obtained. In fact, unlike existing implementations [9], our architecture can operate under a scaled-supply voltage around 77% of nominal Vdd (for 65 nm CMOS technology) while maintaining “reasonable” PSNR. This feature is implemented with negligible overhead in terms of delay, area and power at the nominal voltage. To the best of our knowledge, this is the first time such scalable color interpolation has been proposed.
- 3) Based on the above architecture, we propose a different interpolation window-size (5X6) in contrast to the conventional one (5X5) to improve the speed of interpolation processing by approximately a factor of two with minimal hardware overhead.

The rest of the paper is organized as follows. The classification of important/less-important components in color interpolation is proposed in Section 2. Section 3 presents detailed mathematical analysis of output quality in the scalable color interpolation architecture. The architectural implementation details with the simulation results are presented in section 4. A different filtering window size for improved performance is also presented in this section. Section 5 explains the applicability to our technique to improve parametric yield. Section 6 concludes the paper.

## II. COLOR INTERPOLATION: PRINCIPLES AND DESIGN

In this section, we briefly mention the underlying principle of conventional interpolation schemes and identify the critical/less-critical components in color interpolation.

### A. Determination of critical components

The Bayer pattern [6] (Fig. 3) is the most frequently used CFA interpolation pattern and consists of R, G and B channels. The G channel represents the luminance component (brightness) of a scene, whereas R and B channels together form the chrominance component. The combination of luminance and chrominance components determines the image quality. Interestingly, the Bayer pattern contains twice as many G channels as compared to R and B channels.

Bilinear interpolation is the basic building block for all color interpolation schemes. In this method, the missing color values at each pixel are determined using the average of

adjacent pixels of the same color. For instance, if we want to interpolate the missing colors,  $G$  and  $B$  at the pixel  $R_{i,j}$ , we compute the average of neighboring  $G$  and  $B$  values (Fig. 3):

$$\hat{G}_{i,j} = \frac{1}{4} \sum_{(m,n) \in \{(0,-1),(0,1),(-1,0),(-1,0)\}} (G_{i+m,j+n}) \quad \hat{B}_{i,j} = \frac{1}{4} \sum_{(m,n) \in \{(-1,-1),(-1,1),(1,-1),(1,1)\}} (B_{i+m,j+n})$$

At a  $G$  pixel, the adjacent  $R$  and  $B$  values are interpolated as:

$$\hat{R}_{i,j} = \frac{1}{2} \sum_{(m,n) \in \{(0,-1),(0,1)\}} (R_{i+m,j+n}) \quad \hat{B}_{i,j} = \frac{1}{2} \sum_{(m,n) \in \{(-1,0),(1,0)\}} (B_{i+m,j+n})$$

Note that the locations of  $R$  and  $B$  pixels may be interchanged, but the computation remains the same. The bilinear method does not take into account the edge information and also neglects the use of other color channels. Although this method results in low PSNR, it is widely used due to its simplicity and can be identified as an integral part of all interpolation algorithms.

To improve interpolation accuracy and obtain better image quality, more advanced techniques [6] have been developed that exploit the correlation among  $R$ ,  $G$ ,  $B$  channels or information about edge direction to adaptively interpolate each of the missing colors. In [10], missing colors are estimated by applying bilinear interpolation to the color difference domain  $G-R$  and  $G-B$  to obtain superior image quality. In edge sensing methods [6, 7], the interpolation is adjusted to estimate the pixels along edge directions and not across them. In [8], image quality is enhanced along vertical and horizontal edges only, considering first and even second order pixel differences. More complex sensing methods [11] consider a variable number of edges. Further sophisticated methods [10, 11] have been proposed that lead to PSNR improvement by using results from  $B$  and  $R$  interpolation to correct  $G$  interpolation and vice versa. However, these complex techniques either require excessive hardware for their implementations or additional software processing for practical realization. Therefore, attaining high image PSNR requires careful consideration of both edge sensing and pixel correlation while limiting design complexity.

An effective methodology for providing improved color interpolation is shown in Fig. 4. In this scheme, we divide the computations required for interpolation into two parts based on their impact on the output image quality. The first part consists of the bilinear component, and the second part consists of a correction term. This aspect is also evident from a quantitative analysis (Fig. 5). The correction term is designated as “gradient”. This gradient is directly used to enhance the quality of the estimated color value produced by the bilinear technique (Fig.5).

The gradient is expressed as a 2-D vector; it points in the direction of largest possible intensity increase at each pixel and its value corresponds to the rate of change in that direction. The bilinear component can also be represented by a 2-D vector based on the averaging weights and positions of the pixels. Therefore the bilinear and gradient components

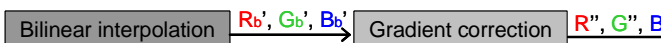


Fig. 4: The 2 parts of color interpolation

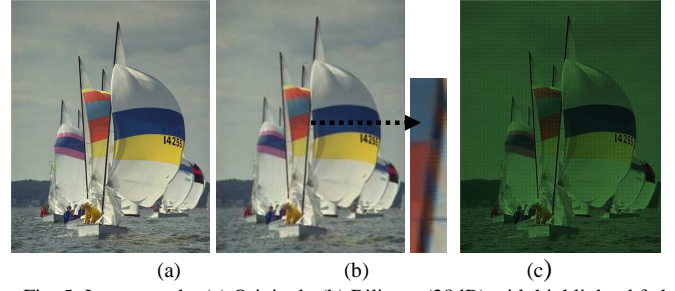


Fig. 5: Image results (a) Original, (b) Bilinear (28dB) with highlighted faded portions compared to original, (c) Gradient correction term (7.495dB)

can be jointly perceived as a 2-D “filter window”. The elements of both the 2-D vectors represent the filter coefficients and the number of elements determines the filter size. For instance, let us consider the 2-D filter shown in Fig. 6(a), where we estimate the missing  $G$  color value at the  $R$  pixel. The coefficient values shown in green color with weights  $\{2, 2, 2, 2\}$  represent the bilinear component, whereas the remaining 5 coefficients  $\{4, -1, -1, -1, -1\}$  express the gradient component of the filter. When such a filter acts on an interpolation region, the output is computed by the sum of products of the filter coefficients with the intensity levels of each pixel at that location contained within the filter window. This can be expressed by the simplified 2-D convolution given by:

$$\text{Filter Response(FR)} = \sum_{i=1}^m \sum_{j=1}^n w_{i,j} z_{i,j}$$

where,  $z$ 's are the intensity of the pixels within the filter window,  $w$ 's represent the filter coefficients and  $m, n$  are the filter window size in  $x, y$  directions. The size of the filter window can be varied (e.g. in Fig. 6(a), we have a filter size of 9). However, increased filter-window size results in a commensurate increase in the complexity.

The gradient calculation requires the computation of derivatives along an intensity direction. Therefore in regions of constant pixel intensities, the gradient should be equal to zero [12]. The filter output always determines the missing channels at a particular pixel position, centered at that pixel. These filter windows (e.g. Fig. 6) operate on the whole image by using a sliding window protocol. The “sliding window” or “interpolation window” size can be varied (e.g. 5X5, 7X7 etc.) depending on performance and filter size requirements.

### III. PROPOSED DESIGN METHODOLOGY FOR LOW POWER/PROCESS VARIATION

In this section, we first discuss our design approach (Fig.1) and then compute the gradual degradation of quality that is introduced due to voltage scaling and parametric variations.

The basic idea behind the proposed voltage-scalable architecture is to retain the bilinear part in all computations and vary the size and complexity of the gradient component to satisfy the low power requirements. At nominal  $V_{dd}$ , the filter windows and coefficients used in our design are similar to that proposed in [15] with 5x5 interpolation window. At nominal  $V_{dd}$ , our design yields high PSNR. As delays increase under scaled voltages and parameter discrepancies,

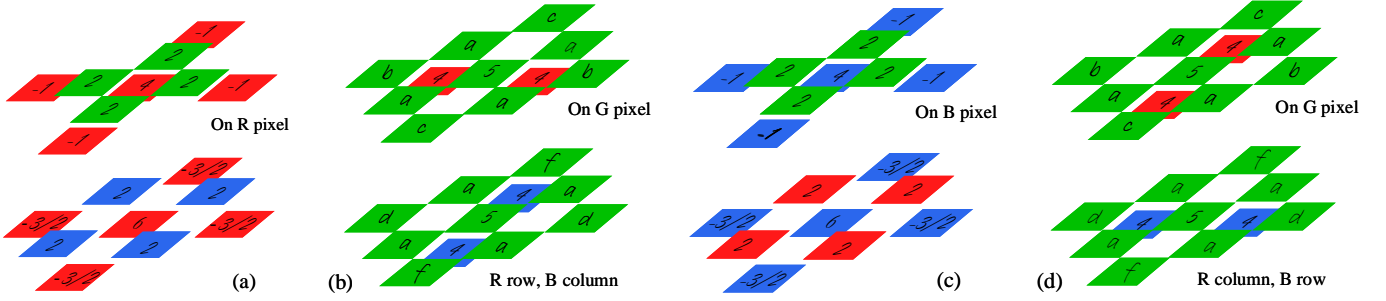


Fig. 6: Color interpolation of: (a) G and B at R pixel, (b) R and B at G pixel, (c) G and R at B pixel, (d) R and B at G pixel

we meet frequency targets by varying the values and the number of coefficients for evaluation of gradient component of each filter. The new filter coefficients are suboptimal in Wiener criterion [14, 15]; however, by performing a rigorous error analysis we attempt to minimize the error imposed after each alteration of the coefficient values/numbers. We also ensure that during such transformations, the coefficients which define the gradient always sum up to zero [12].

#### A. Nominal Vdd

Fig. 6 shows the different filter windows utilized for interpolation of missing colors at nominal Vdd. Fig 6(a), 6(b) shows the coefficients for estimation of G and B values at the R pixel and R and B values at G pixel, respectively, for a row consisting of R and G pixels (Fig. 2). Fig. 6(c) and 6(d) show filter coefficients for calculating G and R values at B pixel and R and B values at G pixels, respectively, for a row consisting of G and B pixels. For the filters shown, we distinguish between the computation of B and R for a G pixel on an R-G row to that of B and R computation of a G pixel on a G-B row, since a slightly different computation is required in each case. However, the same hardware can be used to determine both the computation sets just by switching two inputs for the second case. Similarly, the computation of G and B at the R pixel (Fig. 6(a)) and G and R at B pixel (Fig. 6(c)) are completely identical, with just the red and the blue pixels interchanged. The mathematical expressions of the filtering operation to determine G and B values at R pixel and R and B at G pixel locations at nominal supply voltages are given by Eqns 1, 2 and 3 in Table 1. The filters in Fig. 6(c) (G and R at the B pixel) can be expressed by the same set of above equations with only the R and B literals interchanged. The coefficient values of the corresponding filters are shown

in Fig. 6  $\{a=-1, b=-1, c=1/2, d=1/2, f=-1\}$ . Note that all computations are divided by 8 (shifted by 3 bits) to obtain correct estimated pixel values.

Prediction of missing colors at a pixel through color interpolation introduces some error in the processed image when compared to the original. We perform an error analysis at nominal Vdd by estimating the mean square error (MSE) which is defined as:

$$MSE = \frac{\sum_{i=1}^M \sum_{j=1}^N (P_{i,j} - P'_{i,j})^2}{MN}$$

Here  $P_{i,j}$  and  $P'_{i,j}$  are the pixel values in the original and reconstructed image, respectively. For one particular interpolation window, we need to minimize  $\Delta = |P_{i,j} - P'_{i,j}|$  to obtain a good reconstructed image. We evaluate the error value for two distinct cases:

1) **Homogeneous window**: The R, G and B pixels are highly correlated (no sudden edges) and a *homogenous* region of interpolation is available where pixels have similar intensity values as shown in Fig. 7(a). For instance, we assume that the homogeneous region of Fig. 7(a) consists of pixels with all high intensity values denoted by H. Note that choosing either all high or all low intensities over the same filter window yields identical results in error analysis.

2) **Window with Existent Edges**: The region of interpolation has edges in it. We consider a vertical edge in Fig. 7(b), where H and L denote high and low intensity components, respectively. Similar analysis can be conducted for other kinds of edges, such as horizontal or diagonal edges.

We perform the error computation at nominal voltage after applying filter of Fig. 7(a, b) (R value at G pixel) to the interpolation region. The results for homogeneous/edge cases are summarized in Table 1, where  $P_{i,j} = H_R$  (pixel intensity in original image) and  $P'_{i,j}$  is the reconstructed R channel by using filter shown in Fig. 6(b). For sake of brevity, we show the analysis for only one filter of Fig. 6(b). However, identical analysis can be easily performed for other filters shown in Fig. 6.

#### B. Scaled Vdd

At scaled voltages, we alter the number/values of coefficients determining the gradient. In this subsection, we describe each of the filters utilized for estimation of missing pixels under two scaled voltages (Vdd1 and Vdd2) and analyze their impact by analytically evaluating the errors.

**Scaled Vdd1**: To design an architecture which meets the delay constraints at a reduced voltage, we reduce the number

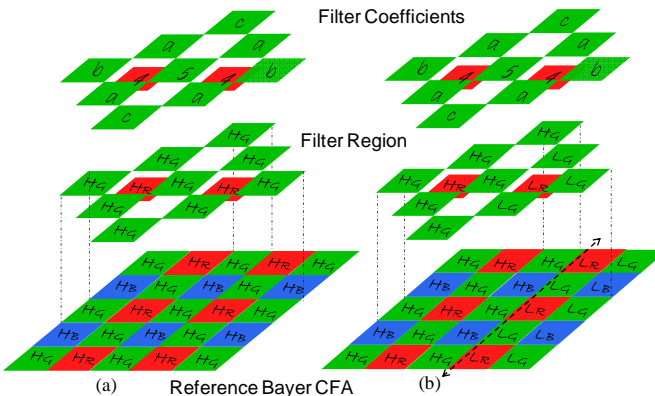


Fig. 7. (a) Homogeneous region, (b) Vertical edge

Table 1: Mathematical expressions for filter windows at different voltages with corresponding error computations

Voltage	Filter Window Computations	Error Computation (All error computations show R values at G pixel for illustration purposes)
Nominal Vdd	<p><u>Interpolation of G and B at R (Eq. 1, 2) / Interpolation of R and B at G (Eq. 3)</u></p> $\hat{G}_{i,j} = \left[ \underbrace{2 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (G_{i+m,j+n})}_{\text{Bilinear}} + 4(R_{i,j}) - \underbrace{\sum_{(m,n)=\{(-2,0),(2,0),(0,-2),(0,2)\}} (R_{i+m,j+n})}_{\text{Edge}} \right] / 8 \quad (1)$	$\Delta_{R,Nominal} =  P_{i,j,orig} - P_{i,j,nominal} $ $=  (8H_R - (8H_R + (5+2c+4a+2b)H_G)) / 8 $ $= (5+2c+4a+2b) H_G / 8$ <p><i>Substituting {a=-1, b=-1, c=1/2}</i></p> $\Delta_{R,Nominal} (\text{Homogeneous}) = 0$ <p><i>Considering edges in vertical direction</i></p> $\Delta_{R,Nominal}(\text{Edge}) =  (8H_R - (4(H_R + L_R) + (5+b+2a+2c)H_G + (b+2a)L_G)) / 8  =$ $= (4((L_G - L_R) - (H_G - H_R)) + (H_G - L_G)) / 8$ <p>Assuming <math>(L_G - L_R) \approx (H_G - H_R)</math> [4] (differences <math>L_G - L_R</math> and <math>H_G - H_R</math> have similar values over a small interpolation region)</p> $\Delta_{R,Nominal}(\text{Edge}) = (H_G - L_G) / 8$
	$\hat{B}_{i,j} = \left[ \underbrace{2 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (B_{i+m,j+n})}_{\text{Bilinear}} + 6(R_{i,j}) - \underbrace{3/2 \sum_{(m,n)=\{(-2,0),(2,0),(0,-2),(0,2)\}} (R_{i+m,j+n})}_{\text{Edge}} \right] / 8 \quad (2)$	
	$\hat{R}_{i,j} = \left[ \underbrace{-1 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (R_{i+m,j+n})}_{\text{Bilinear}} + 5(G_{i,j}) + 4 \sum_{(m,n)=\{(0,-1),(0,1)\}} (R_{i+m,j+n}) \right] / 8$ $- 1 \sum_{(m,n)=\{(-2,0),(2,0)\}} (R_{i+m,j+n}) + 0.5 \sum_{(m,n)=\{(0,-2),(0,2)\}} (R_{i+m,j+n}) \quad (3)$	
	$\hat{B}_{i,j} = \left[ \underbrace{-1 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (B_{i+m,j+n})}_{\text{Bilinear}} + 5(G_{i,j}) + 4 \sum_{(m,n)=\{(0,-1),(0,1)\}} (B_{i+m,j+n}) \right] / 8$ $+ 0.5 \sum_{(m,n)=\{(-2,0),(2,0)\}} (B_{i+m,j+n}) - 1 \sum_{(m,n)=\{(0,-2),(0,2)\}} (B_{i+m,j+n})$	
Scaled Vdd1	<p><u>Interpolation of G and B at R (Eq. 4, 5) / Interpolation of R and B at G (Eq. 6, 7)</u></p> $\hat{G}_{i,j} = \left[ \underbrace{2 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (G_{i+m,j+n})}_{\text{Bilinear}} + 2(R_{i,j}) - \sum_{(m,n)=\{(0,-2),(0,2)\}} (R_{i+m,j+n}) \right] / 8 \quad (4)$	$\Delta_{R,ScaledVdd1} =  P_{i,j,orig} - P_{i,j,ScaledVdd1} $ $=  (8H_R - (8H_R + (4+4a)H_G)) / 8 $ $= (4+4a) H_G / 8$ <p><i>Substituting {a=-1}</i></p> $\Delta_{R,ScaledVdd1} (\text{Homogeneous}) = 0$ $\Delta_{R,ScaledVdd1}(\text{Edge}) =  (8H_R - (4(H_R + L_R) + (4+2a)H_G + 2aL_G)) / 8  =$ $=  (4(H_R - L_R) - 2(H_G - L_G)) / 8 $ $=  (H_G - L_G) / 4 $
	$\hat{B}_{i,j} = \left[ \underbrace{2 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (B_{i+m,j+n})}_{\text{Bilinear}} + 2(R_{i,j}) - \sum_{(m,n)=\{(0,-2),(0,2)\}} (R_{i+m,j+n}) \right] / 8 \quad (5)$	
	$\hat{R}_{i,j} = \left[ 4G_{i,j} - \sum_{(m,n)=\{(-1,-1),(-1,1),(1,-1),(1,1)\}} (G_{i+m,j+n}) + \underbrace{(R_{i+m,j+n})}_{\text{Bilinear}} \right] / 8$ $\quad (m,n)=\{(0,-1),(0,1)\}$	
	$\hat{B}_{i,j} = \left[ 4G_{i,j} - \sum_{(m,n)=\{(-1,-1),(-1,1),(1,-1),(1,1)\}} (G_{i+m,j+n}) + \underbrace{(B_{i+m,j+n})}_{\text{Bilinear}} \right] / 8$ $\quad (m,n)=\{(-1,0),(-1,0)\}$	
Scaled Vdd2	<p><u>Interpolation of G and B at R (Eq. 8) / Interpolation of R and B at G (Eq. 9, 10)</u></p> $\hat{G}_{i,j} = \left[ \underbrace{2 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (G_{i+m,j+n})}_{\text{Bilinear}} \right] / 8 \quad \hat{B}_{i,j} = \left[ \underbrace{2 \sum_{(m,n)=\{(-1,-1),(1,-1),(-1,1),(1,1)\}} (B_{i+m,j+n})}_{\text{Bilinear}} \right] / 8 \quad (8)$	$\Delta_{R,ScaledVdd2} =  P_{i,j,orig} - P_{i,j,ScaledVdd2} $ $=  (8H_R - (8H_R + (4+4a)H_G)) / 8 $ $= (1+a) H_G / 8$ <p><i>Substituting {a=-1}</i></p> $\Delta_{R,ScaledVdd2} (\text{Homogeneous}) = 0$ $\Delta_{R,ScaledVdd2}(\text{Edge}) =  (8H_R - (4(H_R + L_R) + (H_G - L_G)) / 8  =  (4(H_R - L_R) - (H_G - L_G)) / 8 $ $=  3(H_G - L_G) / 8 $
	$\hat{R}_{i,j} = \left[ G_{i,j} - \sum_{(m,n)=\{(-1,-1)\}} (G_{i+m,j+n}) + \underbrace{(R_{i+m,j+n})}_{\text{Bilinear}} \right] / 8$ $\quad (m,n)=\{(0,-1),(0,1)\}$	
	$\hat{B}_{i,j} = \left[ G_{i,j} - \sum_{(m,n)=\{(1,1)\}} (G_{i+m,j+n}) + \underbrace{(B_{i+m,j+n})}_{\text{Bilinear}} \right] / 8$ $\quad (m,n)=\{(-1,0),(-1,0)\}$	
	$\hat{B}_{i,j} = \left[ G_{i,j} - \sum_{(m,n)=\{(1,1)\}} (G_{i+m,j+n}) + \underbrace{(B_{i+m,j+n})}_{\text{Bilinear}} \right] / 8 \quad (10)$	

of operations and hence the critical path of the design by changing the filter size from 9 to 7 (for G and B at R pixel), and from 11 to 7 (for R and B at G pixel). Similar filter size reduction holds for the other four cases. The new filters retain the bilinear component as in nominal Vdd case, and certain terms in the gradient component which result in minimal PSNR degradation are not computed. It should also be

observed that the remaining filter coefficients constituting the gradient component are altered to make their sum equal to zero. This alteration is dictated by minimal hardware overhead and will be explained in the next section. The mathematical expressions with the error computations are summarized in Table 1.

Scaled Vdd2: We further scale the supply voltage to a

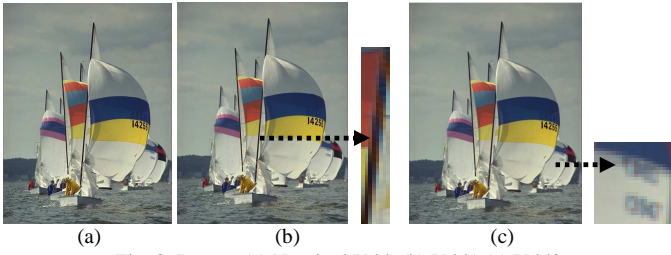


Fig. 8: Images (a) Nominal Vdd, (b) Vdd1 (c) Vdd2

value of 0.77 times the nominal supply voltage. The filters used in this case are optimized based on minimal hardware requirements discussed in the next section. At this voltage, we obtain only the bilinear component for G and B values at R pixel. On the other hand, the R and B values at G pixels produce bilinear and a minor gradient component. The mathematical expressions and error values for this case is also shown in Table 1.

Fig. 8 shows the image results for the original as well as the processed image at nominal and scaled Vdds. As the Vdd is scaled, the images become blurred and the artifacts are more pronounced. In Fig. 8(c), it is observed that in addition to the 8(b) artifacts, the letters (edges) become less visible as well.

### C. Comparison with existing algorithms

To verify the effectiveness of the filters proposed above (Table 1) we applied them on a set of 25 Kodak images and compared the results with the nominal Vdd case and other signal interpolation algorithms (signal correlation [10] and bilinear). In Table 2, the average PSNR for R, G, B channels and the whole images are shown. The filters at nominal Vdd produce better results than signal correlation. At Vdd1 due to error introduced by presence of edges, there is a quality deterioration of around 2 dB. For Vdd2, the magnitude of the error is higher due to further reduction of filter window size, degrading the image PSNR by 4 dB compared to nominal Vdd. We note that even at Vdd2, the proposed filters outperform bilinear in R and B channels due to the computation terms that are maintained (Table 1) for these color channels.

Regarding the design complexity we compare our method with a popular algorithm [10], where the color interpolation method using signal correlation required 20 additions to estimate 2 missing pixel values. Our method on the other hand requires 31 additions to compute 4 missing color channels. Therefore at nominal Vdd we provide similar PSNR at lower computational complexity.

## IV. IMPLEMENTATION OF SCALABLE ARCHITECTURE

In this section, we first propose a method of improving the performance of color interpolation by almost a factor of two with negligible area overhead. We then present our voltage-scalable variation-resilient architecture based on varying the filter size and number of coefficients to suit the scaled supply voltage and/or parametric variations discussed in the previous section.

### A. Architectural innovation for high throughput

The complete estimation of all the missing colors requires

Table 2: Average PSNR improvement in dB over bilinear interpolation for the 25 Kodak images for each color channel and the whole image

channel	Sig. Corr.	Nominal Vdd	Scaled Vdd1	Scaled Vdd2
R	3.6	4.36	2.59	0.62
G	3.24	3.24	1.41	0
B	3.09	3.09	1.95	0.29
image	3.41	3.89	2.37	0.41

the computation of the following: i) R and B at the G pixel ii) B and G at the R pixel and, iii) R and G at the B pixel. In conventional color interpolation, one pixel is targeted at a time and the interpolation of the other two channels is performed at that pixel. For instance, to interpolate G and B at the R pixel, the filters operate on a 5X5 pixel window size at any time instant and traverse the entire image. It is evident that the size of this interpolation region determines the speed of interpolation operation for the entire image. As shown in Section 2, the computation of R and B at the G pixel requires different implementation than computing G at the B or R pixels. Therefore, we need dedicated hardware for interpolation in each case. However, with an interpolation region of 5X5 even with dedicated hardware, we can interpolate the missing colors on only one pixel at a time.

If we assume a row-wise pixel traversal, then based on the arrangement of the CFA, the R and G pixels are traversed in one row and the B and G pixels are covered in the next. Even though the filtering of B and R for a G pixel on an R-G row differs slightly from that of B and R estimation at G pixel on a G-B row, as shown in Section II.A, it is possible to utilize the same hardware just by interchanging the R and B inputs depending on which row is being traversed. Therefore, hardware implementation of the filter windows shown in 6(a) and (b) are adequate to perform color interpolation of all missing pixels depending on the row it is operating on:

- i) B, G channels for R pixel and R, B channels for G pixel
- ii) R, B channels for G pixel and R, G channels for B pixel

With the help of these two observations, we propose a new sliding window for improving the throughput of color interpolation. The main idea here is to take advantage of the significant overlap between the filtering requirements of two adjacent pixels in a row and utilize a size of 5X6 pixels for the interpolation region (Fig. 9 (b)). For instance, consider a row consisting of R-G pixels and their corresponding filters in Fig. 6(a) and 6(b). Now if we utilize a 5X6 window it is possible to interpolate missing channels at two adjacent

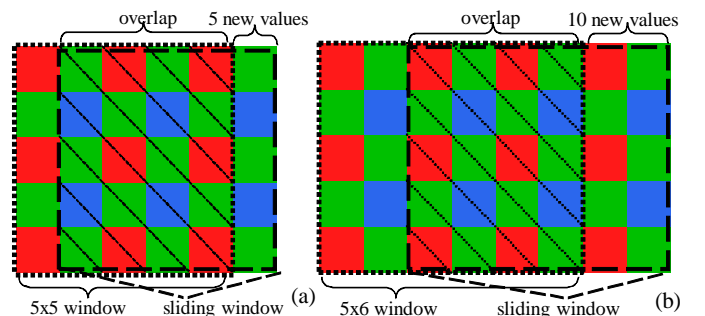


Fig. 9: Comparison of shifting windows for the (a) Conventional (5X5) (b) Proposed (5X6)

pixels in a row (either for R and G in R-G rows, or G and B in the G-B rows) at the same time without any hardware overhead (Fig. 9).

Also by using a 5X6 window, we can obtain 4 outputs (e.g. B and G at R, R and B at G) in each time period compared to only 2 for the 5X5 case. We can see from Fig. 9 that while the 5X5 window requires 5 additional pixel values every time it shifts along a row, the 5X6 windows (Fig. 9) shift 2 columns every time and require 10 additional pixels. Therefore, we obtain twice the throughput with this scheme with minimal additional requirements in hardware (reading only 5 extra pixels every time period).

### B. Scalable Architecture for color interpolation

To attain high parametric yield in presence of variations as well as to enable extreme low power operation under minimal quality degradation, it is essential to design a Vdd-scalable, variation-resilient architecture. Fig. 12 shows the scalable architecture developed for color interpolation. In this case, we have considered the primary inputs for estimating the G and B at R pixel (Fig. 6(a)) and R and B at G pixel (Fig. 6(b)) for the G-B row. Of course, same architecture can be utilized in the G-B case. We analyze the architecture under i) Vdd scaling and ii) parameter variations.

**Vdd scaling:** As shown in Fig. 10, our architecture provides maximum sharing among the 4 computations to optimize the total hardware requirements. For instance, while considering the G and B estimations at the R pixel, we find that the summations of the  $R_{i,j}$  (eqn. 1) terms are identical for both the cases only with different scaling factors. This aspect is utilized to effectively share the hardware and is highlighted in Fig. 10(a). Similarly, while calculating the missing pixels R and B at G, we find that the  $G_{i,j}$  terms are identical in both equations and can use the same hardware for computations

(highlighted in Fig. 10(b)). Moreover, as mentioned in Section II, we need to ensure that the bilinear component is not affected by Vdd scaling/process discrepancies. Therefore, in each of the missing pixel computations we compute the bilinear portion first. For instance, in case of G and B at R pixel, the bilinear component is computed just after 2 adders (Fig. 10(a)), whereas for R and B calculation at G pixel, this component just requires 1 adder to complete (Fig. 10(b)). To reduce the bit-width of the adders used in the computations, we use the shifted values for the input coefficients while implementing this architecture. For example, instead of dividing the final output by 8 (shifting 3 places) as shown in Section II, we consider each of the input coefficients divided by 8 (shifted by 3) in our design.

At nominal Vdd, the frequency of operation is determined by the maximum of the critical path delays for all the computations. It can be inferred from Fig. 10 that the critical path for this design is {four adders + one 3-input multiplexer + one 2-input multiplexer}, which is present at the output. In our architecture, we have maintained the same critical path-length for each computation used in interpolating the missing pixels. At nominal Vdd, both the voltage scaling signals are set to zero and the control signal of the 3-input multiplexer chooses the output from the last adder for each missing pixel computation. The first level of voltage scaling (Vdd1) is selected by setting the control signal  $V1=1$ . Under these circumstances, the size and complexity of the filters are reduced to satisfy timing requirements at minimal PSNR degradation. At this voltage, the computation of {four adders + two multiplexers} is no longer possible and the final adder in each computation has an invalid or incomplete result. The choice therefore remains to scale the voltage to an extent, where the computation of {three adders + two multiplexers (critical path)} is valid. In this case, the multiplexer chooses the output from the third adder in each of

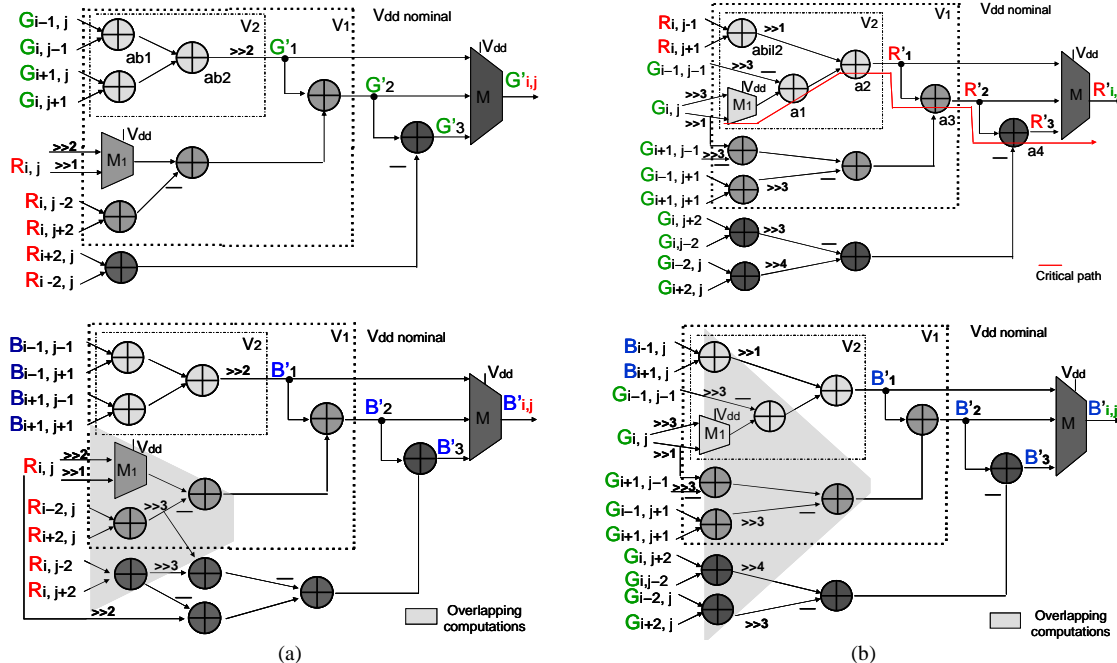


Fig. 10: Proposed Architecture for the interpolation of (a) G and B channels at R pixel, (b) R and B at G pixel

the computations as shown in Fig. 10. At the same time, it should be noted that the outputs of the third adder need to conform to voltage equations and implement the filters shown in Table 1 to obtain the output with the highest obtainable PSNR. The other requirement that needs to be ensured is that the gradient components in this implementation add to a zero value as per the filter requirements from Table 1. We elucidate this further with the help of an example. Let us consider the computation of G at the B pixel in Fig. 10(a). We observe that when the fourth adder is not computed, the bilinear portion is not affected. However, part of the gradient is not computed. Therefore, to obtain a zero value for the gradient under voltage scaling the values of the coefficients for the computation of gradient part are  $\{2,-1,-1\}$  instead of  $\{4,-1,-1,-1,-1\}$ . At the architectural level, this compensation is achieved with the help of the multiplexer shown in Fig. 10(a). The voltage scaling signal ( $V1='1'$ ) controls this multiplexer, and alters the coefficient value of  $R_{ij}$  from  $0.5 * R_{ij}(4/8)$  to  $0.25 * R_{ij}(2/8)$  to make the gradient sum to zero at even a scaled Vdd. Interestingly, only two 8-bit multiplexers at the inputs suffices this entire gradient compensation across all scaled voltages. As shown in Fig. 10(a) and (b), one multiplexer is shared by G and B interpolation at R, and the second multiplexer is shared by the R and B computations at the G pixel.

Further voltage scaling (Vdd2) is obtained by setting the value of the second voltage controller  $V2 = '1'$ . At this voltage, the computation of  $\{\text{three adders} + \text{two multiplexers}\}$  is no longer possible within the specified delay margin. Therefore, the multiplexer chooses the output that is available from the second adder (Fig. 10(a) and (b)). The delay in this case is  $\{\text{two adders} + \text{two multiplexers}\}$  delay, and it dictates the minimum voltage to which the design operates correctly. Under these conditions, the hardware is required to satisfy the relevant equations and implement the filters of Table 1. In this case, the output obtained for the G and B pixels is only the bilinear output. For the R and B colors at the G pixel, we obtain the bilinear value (output of first adder) and a small gradient. Again, the input multiplexer satisfies the zero gradient requirements. The extra hardware required to incorporate the scaled Vdd/process-aware option for all the four outputs, are the (8-bit) 3-input multiplexers at the output of the design and two 2-input multiplexers at the input side. The multiplexers at the input side allow selection of the different coefficient values that are required at each voltage level. These multiplexers contribute to an overall area overhead of 2% and power overhead of 3% at nominal Vdd. Of course, other intelligent circuit techniques might be employed to further optimize the area requirements. Techniques like Vdd/ground gating of the last stage of logic in the adders which serve as inputs to 3-input multiplexers (at each output) can help eliminate the requirement of the 3-input multiplexers and further reduce the area overhead. The area and power results reported in this paper however consider the overhead due to the multiplexers at the output.

We implemented the proposed architecture in Verilog and obtained a synthesized Verilog structural netlist from Synopsys Design Compiler. This Verilog netlist is then

Table 3: Power consumption and PSNR for sails image for various Vdd values

Method	Nominal Vdd		Vdd1=0.89V		Vdd2=0.77V	
	P (V) mW	PSNR dB	P (V) mW	PSNR dB	P (V) mW	PSNR dB
Prop1	13.2	38.15	10.6	35.25	7.9	32.9

converted into a Hspice file. The Hspice files were subsequently simulated using IBM 65nm technology using 1000 patterns and the average power consumption is determined. The delays of the critical paths and the area estimates consist of active transistor areas. We compare the PSNR of the image for nominal and scaled Vdds in Table 3.

**Process Variation tolerance:** We investigated the impact of process variation on our architecture at nominal and scaled voltages. As mentioned in Section 1, inter-die variations affect these circuits more than intra-die variations. At nominal voltages, parameter variations might affect the delays in computations of some chips, which are at the slower process corners. Under such a scenario, the output of the 4<sup>th</sup> adder (Fig. 10) would be incomplete or invalid. However, with the help of an adaptive sensing circuit (Fig. 11) or sensor [16], we can easily detect the process corner and correspondingly set the first Vdd control signal V1 to '1'. This will slightly degrade the quality of the image at nominal Vdd, but will provide a valid output. On the other hand, at scaled Vdds (say Vdd1) because of parametric variations, the computation of the 3<sup>rd</sup> adder might not be complete. Under such conditions, we still have an option to further set  $V2='1'$  and obtain a valid output at the expense of quality of the image.

The adaptive compensation circuit for controlling the multiplexer-enable signals of the color interpolation architecture based on Vdd/process corner is shown in Fig. 11. The mux-ing in the architecture prevents incomplete and/or incorrect computations to propagate to the final output. It is necessary, however, to correctly detect the path delay failures under process variation at nominal or scaled Vdds. The voltage comparators C1 and C2 compare the current supply voltage value with the reference voltage Vref1 and Vref2 to produce the logic value '1' if the supply voltage is equal to or lower than the reference voltage. The reference voltage Vref1 is the scaled voltage at which 3 adders fail to compute in the critical path under worst case process variations. On the other hand Vref2 is the worst case scaled voltage at which 2 adders failed. A process detector circuit provides signals indicating the process corner and a pre-calibrated lookup table contains information about whether a particular path gets affected at a given voltage and process corner. The lookup table is calibrated at design time through extensive Hspice simulations. We store the details of how many adders are properly computed at which voltage and process corners in this lookup table.

The process detection signal can be 1-bit to indicate a slow or fast/nominal corners or it can be 2 or more bits to indicate slow-slow (SS), fast-slow (FS), fast-fast (FF) etc. The lookup table receives this information and generates the corresponding mux-control logic signals. This ensures that

the final values in the computations do not carry any incomplete/erroneous values and correct computation output of 2-adder critical path (bilinear component) propagates to the output. Since the process detectors and voltage comparators are usually present on-chip/on-wafer, the area/power overhead associated with adaptive quality tuning with voltage scaling and/or parameter discrepancies is extremely low. Interestingly, there is no delay overhead due to the control signals of these multiplexers during runtime, since the lookup table statically provides the control signals way in advance.

### C. FPGA Deployment

To prove the effectiveness of our voltage scalable design, we mapped the color interpolation architecture onto an Altera Cyclone II FPGA on a DE-2 board [18]. Fig. 12 illustrates the block diagram of the complete system. The design consists of the color interpolation unit, SRAM/SDRAM controllers and the VGA control block. The user has the option to turn on/off on-board switches and buttons to select operation at the nominal, Vdd1 or Vdd2 voltages (as described in sections III and IV). The system also displays the corresponding PSNR and power values at the on-board LCD display.

The color interpolation processing occurs in the following manner: The Bayer pattern image data is initially stored in the SRAM (Fig. 12). The size of the raw image is 640x480 pixels (8-bits per pixel) occupying a space of 307 KB on the SRAM. The color interpolation unit implements all the filter windows described in Section III. This unit reads the whole Bayer image from the SRAM in 5x6 pixel window sizes (as mentioned in Section IV). It then computes the interpolated values at the nominal or scaled voltages depending on the switch settings set by the user and stores them in the SDRAM through the write FIFO of the SDRAM control unit. The SDRAM control unit consists of a PLL and performs the synchronization of the input data to the SDRAM clock. Since each pixel is represented in 8-bit format and the outputs have 24 bits per pixel (R, G, and B) the total space required for storing the image on the SDRAM is 1200KB. The next step is to display the computed image data from the SDRAM to the VGA monitor. For this, the SDRAM controller processes the read requests from Display controller and reads out the requested R, G and B pixel values from the SDRAM. The display controller, in turn, outputs 2D-pixel locations (x and

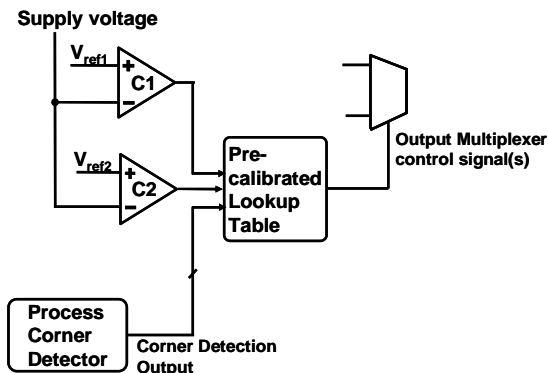


Fig. 11. Adaptive delay compensation circuit

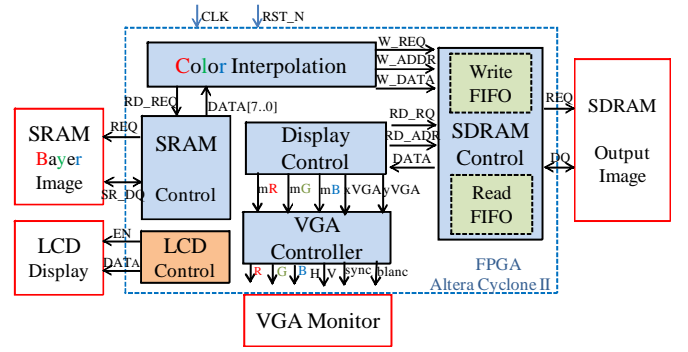


Fig. 12. Detailed block diagram of FPGA implementation of color interpolation algorithm

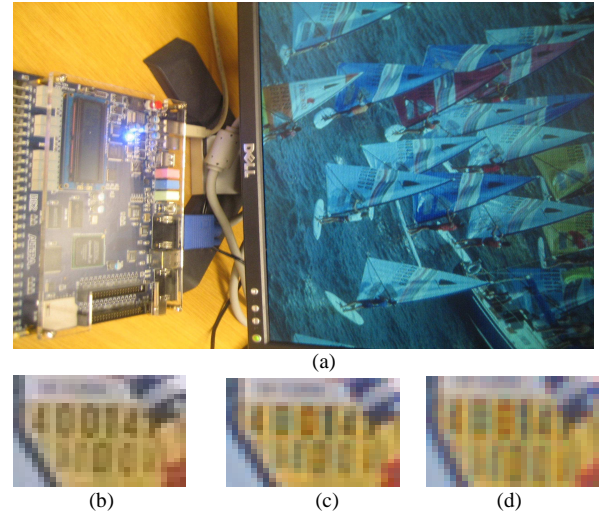


Fig. 13 (a) Altera DE2 FPGA setup with VGA monitor for display. Highlighted parts depict the gradual image degradation as seen in the VGA output, (b) Nominal Vdd: PSNR=34.75dB, (c) Vdd1: PSNR=32.15dB, (d) Vdd2: PSNR=29.31dB

y locations) with the corresponding pixel values and provides these inputs to the VGA controller. The VGA controller generates standard VGA sync signals - VGA\_HS and VGA\_VS (horizontal and vertical synchronization signals for VGA PLL) to enable display of the pixel values on the external VGA monitor. The LCD control block is responsible for displaying the PSNR values depending on the mode of operation. A 50 MHz clock that is generated on board serves as the input clock for the SRAM, SDRAM and the color interpolation blocks. This is synchronized with the 27 MHz clock used for the VGA controller/PLL. VGA controller can generate also a 13.5 MHz in case is required for generation of VGA sync and blank signals.

For the FPGA deployment, we have implemented the entire color interpolation design methodology in Verilog with a 5x6 sliding window operating on the image stored in the SRAM. In our Verilog implementations, we have designed multiplexers (as shown in the architectural implementation of Section IV-A) so that the output that can be obtained based on the computations of 4 (nominal Vdd), 3 (Vdd1) and 2 (Vdd2) adders. The on-board switches of the Altera DE2 board were used to implement the control

switches for the multiplexer depending on whether the output that was computed at 2, 3 and 4 adder levels. The FPGA setup is shown in Fig. 13. For sake of brevity, we have highlighted parts of the image which depict the deterioration of image colors/clarity. As observed in Fig. 13(b), (c) and (d) only a detailed depiction shows a slight degradation in the image quality from nominal to the scaled voltage implementations (even though the power consumption has reduced significantly). Also the PSNR values in Fig. 13 provide quantitative values of output quality with voltage scaling.

## V. PROPOSED METHODOLOGY AND EFFECT ON MANUFACTURING PARAMETRIC YIELD

The proposed architecture is resilient under parameter variations and hence, significantly improves the manufacturing yield of color interpolation chips. As mentioned in Section 1, if the effect of parameter variations is not explicitly taken into account in the design of these architectures, they are likely to fail, severely degrading the parametric yield. Higher yield can be obtained by increasing supply voltage or upsizing the transistors both of which creates more power overhead. For our architecture, we consider two distinct cases:

i) At nominal supply voltages, the output PSNR of the image of color interpolation chips (at the worst case process corner) are minimally affected as only the “gradient” values have errors in computation, ii) At worst case process corners under scaled voltages, we still obtain a graceful degradation of image quality since gradient values are affected at first; of course, beyond a certain voltage quality degrades rapidly since the bilinear computations experience delay failures.

It is evident that the proposed platform generates color interpolation designs that are tailored to provide 100% manufacturing yield under parameter discrepancies by allowing trade-offs between quality and yield. To evaluate the yield improvement, we assumed a threshold voltage spread of 30% for the transistors and computed the delay distributions of the critical paths for 2, 3 and 4 adder levels for the color interpolation architecture by using an SSTA algorithm [17]. Under parametric variations at nominal Vdd, the design with 4 adders in its critical path might get affected. However the last adder stage computes only the gradient and

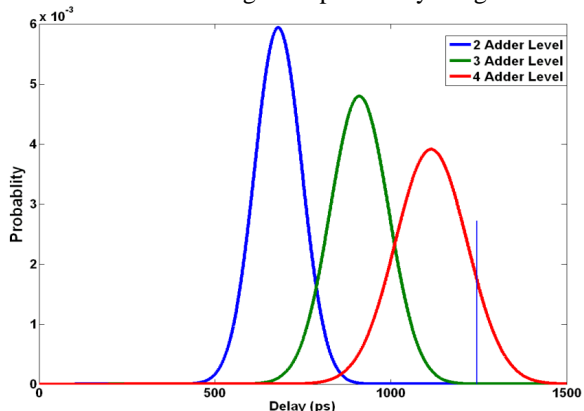


Fig. 14: Path Delay Distribution of 2, 3 and 4 Adder Levels based on their critical paths

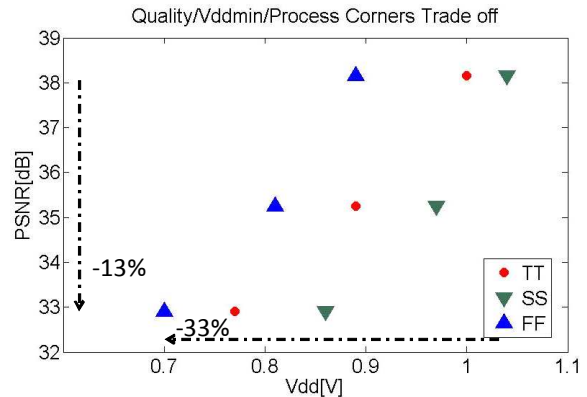


Fig. 15: Trade off Quality/Vdd/Process corners in 65nm IBM technology

by constraining the bilinear computation to 2 adders, we can maintain a fairly high PSNR. This allows us to achieve 100% manufacturing yield for color interpolation designs with minor quality degradation (Fig. 14).

Next we analyze our design to determine how it scales under various process corners. Fig 15 presents the trade off between quality (in terms of PSNR) and minimum voltage required for correct operation of our design at various process corners (SS, FF, TT-typical) in 65nm IBM process technology. For instance, if the chip operates at SS corner then in order to obtain a desired quality (say 38dB) the design has to operate at 1.05V. However, if we can tolerate some quality degradation (say up to 35dB) then the design can be scaled down and operated at 0.97V. For TT and FF process corners, the voltage can be scaled to 0.89V and 0.81V respectively. If we are interested more in low power operation, then the design can be scaled down up to 0.7V in case of FF corner, resulting in 13dB PSNR degradation and almost 50% power savings compared to TT corner operation at nominal voltage.

In comparison, the color interpolation architecture in [9] consists of 2 stages and produces the output by computing the product of the weighted edge-direction with the local gain. Such architectures are not scalable under delay variations. In fact, even at nominal Vdd at the slow corners (SS corner) the design completely fails. Since the bilinear computations are not given priority in this technique, the computational errors at the first stage propagate to the second computational stage leading to a drastic degradation of output quality. This is also true for other architectures [10] where the bilinear component of the image cannot be easily identified and separated out from the filter windows. In contrast, our architecture provides an opportunity to operate under severe process variations both at nominal voltage and scaled voltages. For instance, if the chip operates at SS corner at nominal Vdd (1V), 4 adders fail to complete their computation due to increased delays. Under this scenario, it is not possible to obtain the desired quality of 38dB (4 adder output PSNR) as shown in Fig. 15. In order to maintain a 100% yield without increasing power consumption, we can instead operate at 0.97V (SS corner with 3 adders computed) with minor output quality degradation. In that case, the output quality deteriorates by 2dB, which is preferable to the highly degraded output in conventional architectures.

## VI. CONCLUSION

We have developed a novel design methodology where “intelligent” trade-offs between power, quality and error resiliency is utilized to simultaneously achieve low energy and variation-tolerance, while maintaining a reasonably high quality response. The design methodology has been applied to digital signal processing (in particular, to a color interpolation filter). The basic concept behind this approach is to constrain the “more contributive coefficients” to initial computation stages, and the less important ones to later stages of computation. Even at scaled V<sub>dd</sub> we obtain a very graceful degradation of output quality in presence of parametric disparities.

## REFERENCES

- [1] J. Rabaey et al., “Digital Integrated Circuits”, Prentice Hall, 2002
- [2] K. Roy, S.C. Prasad, *Low-Power CMOS VLSI Circuit Design*, Wiley Interscience Publications, New York, 2000.
- [3] S.Borkar et al., “Parameter variations and impact on circuits and microarchitecture”, DAC 2003, pp. 338–342.
- [4] S. Borkar, “Designing Reliable Systems from Unreliable Components: The Challenges of transistor Variability and Degradation,” *IEEE Micro*, vol. 25, no. 6, Nov./Dec. 2005, pp. 10-16.
- [5] M. Eisele et al., “The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits,” *IEEE Transactions on VLSI Systems*, Dec. 1997
- [6] B. K. Gunturk et al., “Demosaicking: color filter array interpolation,” *IEEE Signal Process. Mag.* 22(1), 44–54, Jan. 2005.
- [7] R. Ramanath et al., “Demosaicking methods for Bayer color arrays”, *J. Electronic Imaging*, vol. 11, pp. 306–315, July 2002.
- [8] J. E. Adams Jr., “Design of practical color filter array interpolation algorithms for digital cameras”, *SPIE*, vol. 3028, pp. 117–125, Feb. 1997.
- [9] S. C. Hsia, et al., “VLSI Implementation of low-power high-quality color interpolation processor for CCD camera”, vol 14, issue 4, pp. 361-369, April 2006.
- [10] S.C. Pei, et al., “Effective color interpolation in CCD color filter array using signal correlation”, *Proc. ICIP*, pp. 488–491, Sept. 2000.
- [11] E. Chang, et al., “Color filter array recovery using a threshold-based variable number of gradients”, *Proc. SPIE*, vol. 3650, pp. 36– 43, Jan.1999.
- [12] R. Kimmel, “Demosaicing: image reconstruction from color CCD samples”, *Trans. on Image Processing*, vol. 8, pp. 1221–1228, Sept. 1999.
- [13] B. K. Gunturk et al., “Color plane interpolation using alternating projections”, *IEEE Trans. on Image Processing*, vol. 11, pp. 997– 1013, Sept. 2002.
- [14] A. Bovik Editor, “*Handbook of Image & Video Processing*”, ISBN 0-12-119790-5, Academic Press, San Diego, 2000.
- [15] Malvar, H.S. et al., “High-quality linear interpolation for demosaicing of Bayer-patterned color images”, *ICASSP*, vol 3, pp. 485-488, May 2004.
- [16] C.H. Kim et al., “On-die CMOS leakage current sensor for measuring process variation in sub-90nm generations”, *Symp. of VLSI Circuits*, 2004, pp.250-251.
- [17]K. Kang et al., “Statistical Timing Analysis using Levelized Covariance Propagation,” *DATE* 2005, pp. 764-769.
- [18] DE-2: Development and Education Board, User Manual, Altera, 2006.
- [19] G. Karakonstantis et al., “Design methodology to trade off power, output quality and error resiliency: application to color interpolation filtering. *ICCAD* 2007: pp. 199-204

We thank the anonymous reviewers for the comments which have significantly helped us improve the quality of the paper.

#### Reviewers' and Associate Editor's Comments

=====

#### Recommendation

-----

Resubmit after Minor Revision for Review

#### Comments to the Author

-----

*The reviewers in general liked the work, however have expressed a few concerns that could be quickly addressed through a minor revision. One reviewer has a concern that the paper has redundancy that can be removed and paper can be shortened to a brief. However, our recommendation is to give the authors a chance to address the concerns and revise the paper and the decision about regular or brief is deferred to the next round of review. Please add a list of responses to the reviewer comments to help quicker review.*

We have addressed the concerns of the reviewers and significantly modified the contents of the paper as per their instructions. However, even after removal of the redundant portions of the paper as per the request of one of the reviewers we still feel that the technical contribution of the paper is large enough (as evident from the shortened modified version) for it to be considered as a regular paper. We also had to increase some sections with more details on simulation results to address the concerns of the other reviewers.

#### *Key references that must be included:*

We have included several key references.

[2] K. Roy, S.C. Prasad, Low-Power CMOS VLSI Circuit Design, Wiley Interscience Publications, New York, 2000.

[4] S. Borkar, "Designing Reliable Systems from Unreliable Components: The Challenges of transistor Variability and Degradation," IEEE Micro, vol. 25, no. 6, Nov./Dec. 2005, pp. 10-16.

[5] M. Eisele et al., "The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits," IEEE Transactions on VLSI Systems, Dec. 1997.

#### Review Number 1.

\*\*\*\*\*

#### Comments to the Author

*This work presents a design framework that trades off power and performance while accounting for inter-die process variations. The paper is well structured and is technically sound.*

*Although the idea of making use of the fact that some computations in DSP applications are non-critical and hence can endure performance losses due to voltage scaling, is not new – the interesting part is including process variations into the picture and applying it to color interpolation filtering.*

*The authors do a good job describing the methodology and results. Perhaps the only two concerns I have are:*

*1. Since this is a TCAD paper, the authors need to include more CAD and circuits paper that discuss the trade-off between variability, power, and performance. They need to cite recent works that talk about this (i.e., delay variation is inversely proportional to Vdd, and design implications). Most of the references are - understandably – signal processing.*

*Answer:* Some references were included in the first draft which talks about variability, power and performance. Among them were the references [1], [2]. However, based on the reviewer's suggestion, we have included several other key reference papers which talk about variability, power and performance as well. The following references have been added:

[2] K. Roy, S.C. Prasad, *Low-Power CMOS VLSI Circuit Design*, Wiley Interscience Publications, New York, 2000.

[4] S. Borkar, "Designing Reliable Systems from Unreliable Components: The Challenges of transistor Variability and Degradation," *IEEE Micro*, vol. 25, no. 6, Nov./Dec. 2005, pp. 10-16.

[5] M. Eisele et al., "The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits," *IEEE Transactions on VLSI Systems*, Dec. 1997.

*2. I think there is a lot of overlap between this work and what the authors have published at ICCAD 2007. Differences should be clearer. Nevertheless, I do think that this would be a good journal paper.*

*Answer:* We have significantly increased the contents of this contribution compared to our ICCAD 2007 version. Three completely new sections have been added to the paper. Besides that the explanations and error computations in each section have been significantly improved. The differences with the ICCAD 2007 paper are listed below.

- 1) Section IV.C, which mentions the details of our FPGA implementation along with the display results, is a completely new addition to this paper.
- 2) Section V, which talks about the effect of manufacturing yield with our design methodology, is also a new addition to the paper. We have added more results and discussed the optimization options that can be explored in terms of trade-offs between quality, power and yield in this section. Extensive simulations in Hspice in 65nm and 90nm IBM process technology were performed. We have analyzed reference architectures under process variations.
- 3) We have included a new section (section IV.B) on how the process sensing can be developed with minimal overhead for this architecture.
- 4) Section IV.A details how the proposed high throughput architecture helps improve performance of color interpolation by almost 2X without any increase in hardware resources. We have only mentioned this briefly in the conference version.
- 5) In Section II, we have explained in detail the underlying concepts of color interpolation that help us design our low power/variation tolerance methodology. Significant additions to the error computation with voltage scaling have been added to Section III.
- 6) The quality-cosmetics of the paper have been improved. We have redrawn several figures to enhance their clarity (Fig. 1, 7, 9 and 12). Moreover we have magnified the artifacts in the image results (Fig. 5, 8, 13) in order to make clearer any quality degradation.
- 7) Redundant parts in writing and figures 8, 9 and 12 were removed.
- 8) We have added several key references.

Review Number 2.

\*\*\*\*\*

Comments to the Author

*Comments to the Author*

-----

*The paper describes a design methodology to trade-off power and process parameter variations against quality. The methodology has been applied to the design of a color interpolation filter and results are shown after mapping the flow into a Altera FPGA.*

*The work revolves around the idea of letting "less important computations" to undergo timing failure while "important computations" work properly which is achieved by allowing less sharing (and hence less MUX delays) to provide adequate slack to withstand timing variations.*

*Overall the paper is well written and the ideas are interesting and seem to be effective in hardware designs of image/video processing applications. A few points needs to be clarified though and may help in improving the manuscript:*

*1) When you are doing voltage scaling are putting the entire chip in the scaled Vdd or only parts of it? In that case is the overhead due to level converters also accounted for? The authors would also like to mention details about how they are achieving this in the Altera FPGA.*

*Answer:* In this paper, we have considered color interpolation as a standalone system. For such a system, there is no need of any level converter as the voltage of the entire system would be dropped based on the output image quality/power requirements.

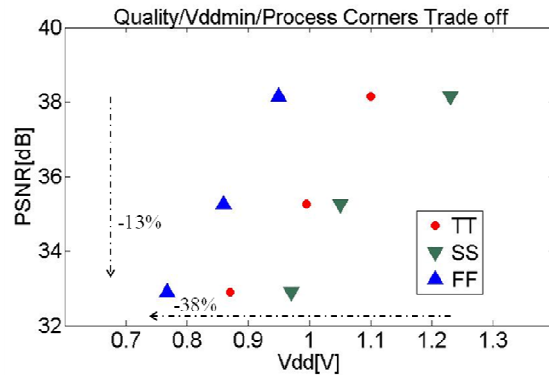


Fig. 1: Trade off Quality/Vdd/Process corners in 90nm IBM technology

For the FPGA deployment, we have implemented the entire color interpolation design methodology in Verilog. In our Verilog implementations, we have designed muxes (as shown in the architectural implementation) so that the output that can be obtained based on the computations of 4 (nominal), 3(scaled voltage 1) and 2(scaled voltage 2) adders. The 2 adder output computations compute only the bilinear component of the image. The on-board switches in the Altera DE2 board were used to implement the control switches for the multiplexer control depending on whether the output that was computed at 2, 3 and 4 adder levels. Therefore, these on-board switches could simulate the voltage scaling (namely nominal, Vdd1 and Vdd2) on the FPGA board and produce the corresponding output images on the VGA display. We have included more details about this implementation in the FPGA section.

The more detailed approach as to the particular voltages and process corners where the 4, 3 and 2 adder levels were operable was obtained by our detailed Hspice results and has been shown in Section V.

2) *The handling of process variations is a bit sketchy. The authors mention they employ sensors on chip to detect process corners and scale the voltage but details are missing. The authors need to illustrate and explain this clearly. While performing evaluations are you also taking into account the overhead due to these sensors?*

*Answer:* We have developed an adaptive compensation circuit for controlling the mux-enable signals of the color interpolation architecture and these signals choose the correct output based on Vdd/process corner. We have provided the circuit and the brief description of the compensation methodology in Section IV.B (process variation)

3) *Evaluations are performed on Kodak 25 images. The authors would like to mention why they chose such a benchmark. Are they typical for image processing systems deployed in commercial applications? Performing experiments on other image benchmark sets will also help for completeness of evaluations.*

*Answer:* The 25 Kodak images are from a standard benchmark suite that is frequently used in image processing and compression applications. Other color interpolation papers also use this standard image set.

4) *Since the work focusses on the power/yield/quality tradeoff it would be nice if the authors could plot the pareto space for such an optimization. It will help in understanding how well the method can scale with decreasing image quality.*

*Answer:* Based on our Hspice variability simulations, we have provided more detailed graphs so that the optimization space for the power/yield trade-offs can be analyzed in Section V. We analyzed our architecture in 90nm IBM technology and the design space that was obtained has been presented in figure 1. Similar information can be extracted regarding the minimum/maximum Vdd operable points in case of the plot described in section V for 65nm technology. In Fig. 1, for the FF corner, the design can be scaled up to 0.76V resulting in 13dB PSNR degradation but substantial power savings. Such graphs can be used in conjunction with the process detector circuit (section IV) in order to ensure 100% yield and acceptable quality under any delay variations due to process variations and even Vdd scaling. Note also that there is a spread in operable Vdd values of 38% and 33% for 90nm and 65nm technology respectively.

5) *The authors use various features of the color interpolation algorithm for further improving throughput. The quality of writing while explaining these have suffered a little and should be rectified.*

*Answer:* As per the reviewer's advice, we have re-written significant parts of the paper to make it more readable. Section IV.A has been modified.

6) *Some of the figures are not well drawn. Redrawing with a better pitch will create a better draft and improve the cosmetics of the paper.*

*Answer:* As per the reviewer's suggestion, we have redrawn several of the figures to make them look better. We have redrawn Fig. 1, 7, 9 and 12 and removed Fig. 8, 9 and 12. Moreover we have magnified the artifacts in the image results (Fig. 5, 8, 13).

*Review Number 3.*

\*\*\*\*\*

*Comments to the Author*

-----

*Dear authors,*

*the idea you propose is in principle a good one, but the paper needs to be shortened and modified.*

*Major issue 1: Basically, you suggest to cope with inter die by reducing the voltage for the slowest designs (by reducing the quality). Many authors successfully proved, that doing the opposite will be very beneficial for the yield. I think, that your idea may be useful to implement a low power mode, but not to cope with variation. Maybe I am wrong, with this opinion, so you need to show me with a meaningful evaluation:*

*Answer:* In this context, we need to mention how quality and yield here are related.

The term "yield" is conventionally used to indicate the number of good operating chips at nominal voltage. In conventional design methodologies, where primary importance is given to area/power optimization, if there are delay variations due to voltage scaling/slow corners in the design then the incorrect/incomplete computations of the design are propagated to the final outputs, causing them to deteriorate significantly, thereby severely degrading design yield.

In our methodology, we first identify the crucial/less-crucial components of a system (in terms of contribution towards the output values). Then we design the system in such a manner such that the critical components can be computed faster than the less critical ones. In the context of color interpolation, we perform this by ensuring that the bilinear components be computed earlier than the gradient ones. Such a technique ensures that if any delay variations occur, only the gradient components are affected in their computations. Of course, such a voltage has a lower bound below which even the critical components get affected and we obtain erroneous output. Muxes at the output ensure which computation is correct at a particular voltage and process corner. More details of the sensing mechanism etc. have been provided in Section IV.

Our "yield" remains high even under voltage scaling/process variations because we can provide outputs at slightly lower quality. Consider the case of our color interpolation design. The ideal output can only be obtained if the four adders are fully computed. However, if at nominal V<sub>dd</sub> (at the slow corner say), due to increased delays the complete computation of the 4 adders is not possible; our architecture ensures that we can still have a functional chip where the correct computation till 3 adders occurs. We obtain a slightly degraded image output (since we lose some portion of the gradient information); however, our chip continues to work unlike conventional system designs which fail drastically under such delay variations. Therefore, we are able to provide 100% yield at different process corners even at scaled voltages by sacrificing some amount of quality.

***We do not reduce the voltage for the slowest designs to improve our yield. We provide an architecture which allows us to continue to adapt to such variations in delay and operate correctly by modulating (decreasing) the output image quality at scaled voltages even under process variations. This is obtained by disregarding some part of the total computation which is not so critical to the output quality. However, as shown in Section V the lowest voltage at which you can operate at the slow corners is still always higher than the voltages you can operate at fast corners (this is intuitive as well).***

*Compare A) a reference design w/o variation awareness B) your design C) a version of the reference design using V<sub>dd</sub> scaling (with V<sub>1</sub>>V<sub>nom</sub>>V<sub>2</sub>) against variation.*

*Under the assumption of a threshold voltage variation (inter die), analyse the spread in CP delay and total power (dynamic and static).*

*Answer:* As mentioned in the paper, this is the first attempt to design a color interpolation system where simultaneous low power and process tolerance can be obtained by identifying critical/less-critical computations and prioritizing the critical ones over the less-critical ones. However, for the satisfaction of the reviewer we have analyzed our design through extensive HSPICE simulations to determine how it scales under various process corners and the trade off between Quality/Vdd/process corners is described in section V. We analyzed also our architecture in 90nm IBM technology and a similar design space has been obtained presented in Fig. 1 of this response letter. We also analyzed the computations of other color interpolation architectures [9] where conventional design principles have been adopted. This technique not only requires complex operations, such as multiplication and divisions but also it is evident that bilinear computations cannot be separated out from the filter windows. Moreover, the technique is implemented with a pipelined architecture. The critical path in this architecture consists of a multiplier and a divisor. Furthermore, in the first stages of pipeline the averages and local mean of each color channel is calculated which have to be used in the next stage in which the essential interpolation of each missing color is taking place. All the above characteristics lead to a non scalable architecture. Not only some outputs are not correctly computed under delay errors, there is no additional provision to remove the incorrect computations which affect the outputs drastically. Besides, any errors taking place in the first stage due to delay failures propagate to the next computation stages resulting in a highly degraded output. This is also true for other conventional architectures such as the one in [10] where also the bilinear interpolation cannot be easily identified and separated out from the filter windows. In contrast, our architecture provides an opportunity to not only operate under severe process variations at nominal voltage, but also under scaled voltages. For instance, if the chip operates at SS corner at nominal Vdd (1V), 4 adders fail to complete their computation due to increased delays. Under this scenario, it is not possible to obtain the desired quality of 38dB (4 adder output PSNR) as shown in Fig. 15. In order to maintain a 100% yield without increasing power consumption, we can instead operate at 0.97V (SS corner with 3 adders computed) with minor output quality degradation. In that case, the output quality deteriorates by 2dB, which is preferable to the highly degraded output in conventional architectures. We have reported several Pareto plots in Section V to highlight the opportunities of trade-offs between power and output quality. The plots also give us details about the spread in power values at different voltages/process corners.

*Major issue 2: Your work is way too long. I already got your idea after reading the introduction. My suggestion: Use the introduction for an introduction of the background (10 lines on process variation, and less than 1 page on the entire Bayer-to-RGB conversion). Then describe the basic idea on 1-2 pages and the implementation on 1-2 pages. After that, discuss the analytical and numerical errors and the power, quality and yield measurements.*

*Answer:* As per the reviewer's suggestion, we have significantly modified the contents of this paper. We have shortened several parts of the paper, especially the introduction and chapter II and III. We have removed Fig. 8 and 9, which showed the filter windows at scaled voltages (of the original transcript). We have also removed Fig. 12 from the Section IV.A. We have, however, added new areas of explanation and details (process sensing, FPGA implementation and trade off results) as well. However, some explanations/figures in the error computation are retained because we feel it improves the readability of the paper. Also we strongly believe that even after significant shortening of details there is enough technical contribution to consider this as a regular paper.

*Minor issues in order of occurrences:*

*Chap I) Introduction already presents the entire idea.*

*Answer:* We have reduced the size of the introduction and sections II and III (by almost 1.5 pages). However, we wanted to present the overall concept in the introduction itself for enhanced clarity. We provide the intricate details of the implementation in the following sections.

*Chap II) Needs to be much shorter*

*Answer:* We have reduced the contents of Chapter II and III as well.

*Fig6,7,8,... These figures are absolutely redundant to the respective equations. Use not more than one of these RGB Figures*

*Answer:* We have removed some figures and retained others for purposes of clarity. We have removed Fig. 8 and 9, which showed the filter windows at scaled voltages (of the original transcript). We have retained Fig. 7 to explain the homogeneous and edge concepts. We have significantly cut down on the writing in Section III and condensed the entire filtering and error analysis in a single table. We have also removed Fig. 12 from Section IV.A.

Equation 3) Reformat (wrap inside a bracket, but an unwrapped denominator)

Answer: We have reformatted Equation 3.

Page 6 First paragraph: Using the word 'frequency' here is absolutely irritating. Use a better terminology here (e.g. amplitude)

Answer: We have altered the “frequency” term by using the term intensity.

Page 6 error computation & Figure 7: Your error computation would be more meaningful, if you assume, that the pixels follow a narrow Gaussian distribution with the mean  $H$  and  $L$ . Avoid all the intermediate steps. (E.g. The user surely knows, that  $5+2*1/2-4-2=0$ ). explain the error computation once, and then give all errors in one table

Answer: We have made the error computation section more concise by retaining some relevant portions and condensing the other portions in a table.

Fig 5,10,15: I can not see any difference: Enlarge a small part or compute the error picture, or just leave away the pictures

Answer: It is necessary to keep some of these figures to illustrate the concepts. However, as per the reviewer’s suggestion, we have highlighted relevant parts of the figures (instead of the whole figures) to illustrate the quality degradation. In the revised version of the paper the figure numbers were changed; Figure 10 became Figure 8 and Figure 15 became Figure 13.



Figure 5b

We can observe artifacts on the images. As the Vdd is scaled, the images become blurred and the artifacts are more pronounced. In Fig. 10(d), it is observed that in addition to the 10(c) artifacts, the letters (edges) become less visible as well.



Figure 10c

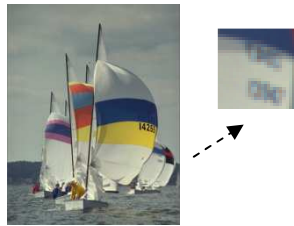


Figure 10d

Fig. 15 tries to highlight the deterioration of colors and image clarity, which become more blurred while artifacts increase. Below details of the obtained images at scaled Vdd values are shown.



Figure 15b



Figure 15c



Figure 15d

Page 9: Critical path is not longer in the area saving version? Than explain, where the 2-input multiplexer is used then.

Answer: As the reviewer has noted the critical path in our design is slightly longer than the base design because of the mux-insertions. But we have noted in our Hspice simulations that this increase is very nominal.