Asian Journal of Information Technology 13 (5): 291-299, 2014
ISSN: 1682-3915

An Area Efficient Weighting Coefficient Generation Architecture for Polynomial Convolution Interpolation

[1]C. John Moses and [2]D. Selvathi
[1]Department of Electronics and Communication Engineering,
St. Xavier's Catholic College of Engineering, Nagercoil, Tamil Nadu, India
[2]Department of Electronics and Communication Engineering,
Mepco Schlenk Engineering College, Sivakasi, Tamil Nadu, India

Abstract: Interpolation is a technique which is used to enhance or reduce the size of digital images to correct spatial distortion. In convolution based interpolation scheme, the quality of the scaled image depends upon the order of convolution kernel. A better quality of interpolation can be achieved by using higher order models but it requires complex computation and heavy memory access time. Fast First Order Polynomial Convolution Interpolation Method is one of the efficient methods for scaling images still there exists complexity in generating weights. This study presents an area efficient Weighting Coefficient Generation (WCG) circuit for digital image scaling. In the proposed hardware architecture, the WCG circuit has decreased the hardware complexity substantially. It includes only nine arithmetic components which is much less than the number of components in the existing hardware architecture of WCG of Fast First Order Polynomial Convolution Interpolation (FFOPCI) algorithm. The computational burden is less when using the proposed architecture for generating the weighting coefficients in FFOPCI algorithm. Thus, high quality scaled images can be obtained with low computational burden when using the proposed architecture in FFOPCI algorithm.

Key words: Image scaling, interpolation, FPGA, weighting coefficient, simulink, system generator, image resizing, cubic convolution, keys interpolation, Bi-Cubic Interpolation

INTRODUCTION

Digital image scaling is the process of resizing a digital image based upon a scale factor. In recent years, digital display devices which use digital data in displaying images have become more and more popular (Tsai *et al*., 2011). Displaying digital images needs resizing the image which involves operations such as scaling up and scaling down operation.

Digital image scaling including scaling up and scaling down, has a variety of applications. Commonly, it is widely used in medical image processing (Chen, 2013a), image zooming (Chen *et al*., 2011), multimedia (Chen, 2013b) and computer graphics, etc. In many applications, from consumer electronics to medical imaging, it is desirable to improve the restructured image quality and processing performance of hardware implementation (Chen *et al*., 2011; Kim *et al*., 2003; Lin *et al*., 2007; Fry and Pusateri, 2009). Scaling is a non-trivial process that involves a trade-off between efficiency, smoothness and sharpness. As the size of an image is increased, the pixels which comprise the image become increasingly visible, making the image appear soft. Conversely, reducing an image will tend to enhance its smoothness and apparent sharpness. Usually the resized image will be lower in quality as the new pixels are found out based upon some prediction or approximation and also it need more computations.

Nowadays much different kind of interpolation kernels are used to reconstruct the image by using either down-sampling or up-sampling the image for improving the image quality (Madani *et al*., 2012). The algorithms used for resizing digital images are broadly classified as Non-Adaptive Interpolation algorithms and Adaptive Interpolation algorithms. In non-adaptive interpolation scheme, linear and fixed pattern of computation is applied in every pixel. This technique is fixed irrespective of the input image features and has low computational complexity.

In adaptive interpolation scheme, non-linear type of computation is applied based on sharp edges and smooth texture (Xiao *et al*., 2006; Shezaf *et al*., 2000; Hwang and Lee, 2004). This technique is computationally inefficient and expensive. The computational logic of an adaptive

Corresponding Author: C. John Moses, Department of Electronics and Communication Engineering,
St. Xavier's Catholic College of Engineering, Nagercoil, Tamil Nadu, India

image interpolation technique is mostly dependent upon the intrinsic image features and contents of the image whereas computational logic is complex (Lin *et al*., 2008a). As adaptive interpolation algorithms need much more hardware resources which are very expensive (Lin *et al*., 2010). Mostly Non-Adaptive Based algorithms are mainly used for real time applications. Convolution based interpolation scheme is the most commonly used non adaptive interpolation scheme for digital image scaling. In convolution based interpolation scheme the input image is multiplied with the convolution kernel to find the resized image. In scaling the image quality highly depends on the used interpolation technique (Lin *et al*., 2008b). The numerical accuracy and computational cost of interpolation algorithms are directly tied to the interpolation kernel. As a result, interpolation kernels are the target of design and analysis. So, a kernel of higher order and with simpler weighting coefficient circuit is chosen to generate high quality images with low computational complexity.

The degree of the kernel represents the order of the model. A better quality of interpolation is performed by using higher order models but it requires complex computation and heavy memory access time. A superior interpolation technique does not cause interpolated image distortion nor does it need complex computation (Tsai *et al*., 2011).

Fast First Order Polynomial Convolution Interpolation is one of the Non-Adaptive Interpolation algorithms for digital image scaling. The Kernel of Fast First-Order Polynomial Convolution Interpolation Method was developed with third-order approximation of normal interpolation kernel (Lin *et al*., 2008a). This scheme has the advantages of low operation complexity, weighting coefficient generation effort and hardware cost. Consequently, this architecture has solved the problem of blur and blocking effect and enhances the quality of image. Still there exists complexity in generating the weighting coefficients.

The two criteria which are essential to produce high quality images are the order of kernel and the complexity in generating weighting coefficients (Lin *et al*., 2008b). Increase in the order of kernel increases the quality of the scaled image, at the same time it increases the computational burden of generating weighting coefficients. So, mostly third order kernels are used to generate high quality scaled for processing digital image scaling for high Definition Television (HDTV) in real-time. By decreasing the number of components in the weighting coefficient circuit the computational complexity such as area and cost can be reduced. Field Programmable Gate Arrays (FPGAs) are widely used for rapid prototyping of Digital Signal Processing (DSP) Systems (Meijering *et al*., 1999). FPGA technology is used to improve performance while providing programmability and dynamic reconfigurability (Moses *et al*., 2011, 2010). FPGAs have millions of gates, reasonably on-chip memory and fast input output interface. Therefore, FPGAs can provide an easy and cost effective way to evaluate Image Processing algorithms from an implementation perspective. The Proposed algorithm is synthesized using Xilinx ISE for Xilinx Virtex-6-xc6vsx315t-3ff1156 FPGA.

The proposed architecture has decreased the hardware complexity substantially. The computational burden is less when using the proposed architecture for generating the weighting coefficients in FFOPCI algorithm. It includes only nine arithmetic components which is much less than the number of components in the existing hardware architecture of WCG for FFOPCI algorithm. Thus, the weighting coefficient circuit becomes simpler and an area efficient one. Thus high quality scaled images can be obtained with low computational complexity when using the proposed architecture in FFOPCI algorithm.

CONVOLUTION BASED INTERPOLATION

Convolution Based Interpolation (CBI) is one of the most generally used methods for digital image scaling process (Meijering and Unser, 2003; Miklos, 2007). The convolution based image interpolation performs image scaling by convolving the input image with interpolation kernel. The convolution process is used to reform a two dimensional s(x, y) from the discrete samples of the signal by using reconstruct filter with 2D continuous impulse response. Convolution based interpolation method applies constant convolution kemels to resample the entire image (Shezaf *et al*., 2000). According to Eq. 1, a real-valued N-dimensional discrete image s(k, l) is used to construct a real-valued continuous image s(x, y) can be constructed by means of interpolation, i.e.:

$$s(x, y) = s(k,l)*h(x) \quad (1)$$

Where:

s(x, y) = A continuous image
s(k, l) = A discrete image
h(x) = An interpolation kernel
* = The convolution

So, as to acquire a satisfactory reconstruction in conditions of together computational price and mathematical accuracy, it is needed to intend a kernel of fixed level that resembles the sinc-function to the extent that possible.

Features of convolution based interpolation kernel: For an accepted sampling grid through unit distance among the sample points, the regular piecewise nth order polynomial image interpolation kernel is defined by:

$$h(x)=\begin{cases} a_{ni}|x|^{n}+\ldots+a_{ni}|x|+a_{0i} & i\le|x|<i+1 \\ 0 & m\le|x| \end{cases} \quad (2)$$

where, i = 0, 1, ..., m-1, the relationship between m and is n = 2m-1. The (n + 1) m coefficients a_{ij} can be established by imposing constraints on the interpolation kernel h(x). As the samples of discrete image s(k, l) are the correct values of the novel image at the locations (k, l) on the specified sampling grid, the rate of the interpolant on those locations should also be identical to the sample values.

Supplementary constraints are obtained by requiring the interpolation kernel h(x) that consists of piecewise polynomials and to be continuous also if feasible have continuous derivatives on the transition points. These conditions can be interpreted into the following constraints: h(x) = 1 and h(x) = 0 for |x| = 1, 2, ..., m-1 and $h^{(l)}(x)$ should be continuous on |x| = 1, 2, ..., m where, the superscript (l) indicates the first derivative and the second constraint grasps for l = 0, 1, ... k. Where:

$$k=\begin{cases} 0 & n=1 \\ n-2 & n>1 \end{cases}$$

The value of k has to be adequately large in order to give up an adequate number of equations to facilitate be capable of answer for the unidentified coefficients.

CUBIC CONVOLUTION INTERPOLATION

Cubic Convolution Method is a third degree interpolation scheme that practically fit approximates the notionally finest sinc interpolation function (Tsai *et al*., 2011). Since, it approximates the perfect sinc interpolation function the eminence of scaled image is superior to other convolution based interpolations. The kernel is collected of piecewise cubic polynomials described lying on subintervals (-2, -1), (-1, 0), (0, 1) and (1, 2). Exterior of this space, the kernel of the interpolation is zero. For obtaining the cubic convolution kernel, 8 linear equations to be solved with 7 unknown parameters as a result, the system has a free parameter that can be controlled by the user. The kernel is defined as:

$$K_C(s)=\begin{cases} (\alpha+2)|s|^{3}-(\alpha+3)|s|^{2}+1 & 0\le|s|<1 \\ \alpha|s|^{3}-5\alpha|s|^{2}+8\alpha|s|-4\alpha & 1\le|s|<2 \\ 0 & 2\le|s| \end{cases} \quad (3)$$

The value of α establishes the exactness of the kernel. A number of researches have been carried out to discover out the finest value of α. A few methods of them are given as:

- Keys Method
- Bi-Cubic Method
- Fast First-Order Polynomial Convolution Interpolation

Keys interpolation: Keys Image Interpolation Method is one of the widely used cubic convolution interpolations. The kernel of this interpolation is of third order. It is obtained by cubic convolution interpolation kernel. Because "α" the free parameter establishes the exactness of the kernel, through appropriately tuning this parameter lofty worth images can be obtained. The worth of scaled images is straightly attached up to the free parameter "α".

Keys established the constant α = -0.5 by means of a third order approximation. By deputing α = -0.5 in the Eq. 3, the Keys kernel is gained. The kernel of keys interpolation method is:

$$K_{keys}(s)=\begin{cases} 1.5|s|^{3}-2.5|s|^{2}+1 & 0\le|s|<1 \\ -0.5|s|^{3}+2.5|s|^{2}-4|s|+2 & 1\le|s|<2 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

The weights are produced by means of the above kernel. The effect achieved by this kernel is better than the nearest neighbor image interpolation and bi-linear image interpolations (Lin *et al*., 2010).

Bi-cubic convolution interpolation: The distinctive method of cubic convolution image interpolation group is bi-cubic convolution image interpolation (Nuno-Maganda and Arias-Estrada, 2005). Bi-cubic image interpolation utilizes the constant α = -1. Also, it matches the slope of the perfect sinc-function on s = 1.

Principally, the procedure of bi-cubic convolution interpolation wants the coordinates of 16 nearby basis pixels of every interpolated point (Miklos, 2007). The process of two dimensional image bi-cubic convolution interpolation needs the computation of 16 weighting coefficients produced along from 16 nearby pixels of a source image. It has fine image value but it wants difficult

calculation and crucial memory access times. By using α = -1 in Eq. 3, the bi-cubic convolution kernel can be obtained. The kernel of bi-cubic convolution scheme is:

$$K_C(s) = \begin{cases} |s|^3 + 2|s|^2 + 1 & 0 \le |s| < 1 \\ -|s|^3 - 5|s|^2 - 8|s| + 4 & 1 \le |s| < 2 \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

Nuno-Maganda decomposed the Bi-Cubic Convolution Interpolation Method as two 1D interpolations on a Xilinx Virtex II Pro FPGA for real-time image processing. It has good image value and less computational complexity than bi-cubic image interpolation at a stand still the kernel needs complex computation and intense memory access times (Miklos, 2007).

To develop the effectiveness of bi-cubic interpolation, Lin demonstrated a Very Large Scale Integration (VLSI) design of bi-cubic convolution interpolation for digital image processing (Lin *et al*., 2008b). Its architecture diminishes the computational complexity of producing weighting coefficients and the amount of memory access times. Though, the kernel of cubic convolution image interpolations still needs complex computations.

A better-quality interpolation scheme does not make interpolated image distortion, nor does it need complex computation. To defeat the above said shortcomings fast first-order polynomial convolution interpolation method was presented (Lin *et al*., 2010).

Fast first order polynomial convolution interpolation: Fast First Order Polynomial Convolution Interpolation (FFOPCI) is a high-performance image scaling method with high excellence and diminished calculation complexity of generating weights. The kernel of fast FFOPCI was made on third order interpolation kernel.

This scheme has the benefits of low operation complication which decreases the input data accesses and coefficient generating effort also hardware cost (Lin *et al*., 2010). The weighting coefficient generation circuit comprises only 12 arithmetic elements which is greatly less than other higher order models. Also, this design has solved the trouble of blur and blocking outcome and improves the quality of image.

The FFOPCI Method estimates the ideal sinc-function in the distance [-2, 2]. Therefore, the novel kernel is gained also the polynomial of the novel kernel can be characterized as:

$$k_p(s) = \begin{cases} 1 - \left(1 + \frac{4\alpha}{9}\right)s & 0 \le |s| < \frac{1}{3} \\ \left(1 - \frac{2\alpha}{9}\right) - \left(1 - \frac{2\alpha}{9}\right)s & \frac{1}{3} \le |s| < 1 \\ -\frac{4\alpha}{9} + \frac{4\alpha}{9}s & 1 \le |s| < \frac{4}{3} \\ \frac{4\alpha}{9} - \frac{2\alpha}{9}s & \frac{4}{3} \le |s| < 1 \end{cases} \quad (6)$$

where, α is sharpness parameter. The excellence of the interpolated images is straightly attached to this sharpness parameter. By rightly tuning this parameter, elevated excellence images can be gained.

So, as to gain a best rate of sharpness parameter (α) that has enhanced image interpolation eminence, the addition of standard deviation is used to establish the rate of α, in the range of -1 and -0.5 whose range estimates an ideal sinc-function. The least standard deviation can be gained at α = -0.853. Therefore, the kernel of FFOPCI scheme for α = -0.853 is agreed as $k_{p-0.853}$:

$$k_{p-0.853}(s) = \begin{cases} 1 - 0.6209|s| & 0 \le |s| < \frac{1}{3} \\ 1.1896 - 1.1896|s| & \frac{1}{3} \le |s| < 1 \\ 0.3791 - 0.3791|s| & 1 \le |s| < \frac{4}{3} \\ -0.3791 + 0.1896|s| & \frac{4}{3} \le |s| < 1 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

HARDWARE ARCHITECTURE

The block diagram of the hardware architecture for digital image scaling is shown in Fig. 1 which includes coordinate calculation unit, memory bank, Weighting Coefficient Generator (WCG), vertical and horizontal interpolation units and virtual pixel buffer (Lin *et al*., 2010).

Coordinate calculation unit: The main blocks of coordinate calculation unit are interpolated coordinate accumulator, row/column address calculator and vertical and horizontal distance calculator.

Interpolated coordinate accumulator: The fast first order polynomial convolution interpolation requires 16 source pixels that around an interpolated point of a source image. The coordinate of the interpolated point is obtained in the

interpolation coordinate calculator. The row addresses and column addresses of its 16 neighboring source pixels then can be determined. Afterward, the values of the 16 source pixels in off-chip memory can be obtained and fast first order polynomial convolution interpolation can proceed.

Row/Column address calculator: In the circuit of row/column address calculator the operation of vertical or horizontal address orientation is controlled by the signal of vertical/horizontal. This signal determines the vertical (y_m) and the horizontal (x_n) coordinates. If the signal vertical/horizontal is vertical then the row addresses and the vertical interval s_v can be obtained. Otherwise the column addresses and the horizontal interval s_h can be found.

Vertical and horizontal distance calculator: To simplify the weighting coefficient generation, the distance in vertical direction and horizontal direction has to be found before the calculation of vertical weighting and horizontal coefficients. The vertical distance calculator calculates distance between the source pixel and the virtual interpolated pixel. Similarly, horizontal distance calculator calculates distance between the virtual interpolated pixel and the final interpolated pixel.

Memory bank: The image which has to be scaled is stored in memory bank. The pixels are retrieved from the memory bank based upon the generated row and column address. For every interpolation operation 16 source pixels that around an interpolated point of a source image is accessed from this bank.

Weighting coefficient generator: The most important computation in convolution based scaling is the calculation of interpolation weighting coefficients. The weighting coefficient generator as depicted in Fig. 2 is designed for producing vertical and horizontal weighting coefficients. The final outputs are $-v_i$, v_{i+1}, v_{i+2} and $-v_{i+3}$ if the control signal vertical/horizontal is vertical otherwise, the outputs are $-h_i$, h_{i+1}, h_{i+2} and $-h_{i+3}$.

Vertical and horizontal interpolation units: The vertical and horizontal interpolations have the same operation but they have to execute in parallel to accelerate scaling speed. The vertical interpolation unit performs interpolation in column wise manner to produce virtual pixel. The horizontal interpolation unit performs interpolation for these virtual pixels to produce interpolated pixel.

Virtual pixel buffer: The virtual pixels created from vertical interpolation are stored in the virtual pixel buffer as shown in Fig. 1 where to be accessed in the process of horizontal interpolation. In general, scale ratio will determine both input and output data rates of this buffer. While scaling up, each virtual pixel may be reused for the horizontal interpolation such that the data rate of vertical interpolator is less than that of the horizontal interpolator in a period of time. In the process of scaling up, the buffer will temporarily suspend the operation of vertical interpolation which cease the generation of virtual pixels while the number of virtual pixels is more than certain amount that required by horizontal interpolation. In the operation of scaling down, the horizontal interpolation will be halted if the buffer detects that the amount of valid virtual pixels in the virtual pixel buffer approximately approaches the quantity needed for horizontal interpolation until more virtual pixels are ready.

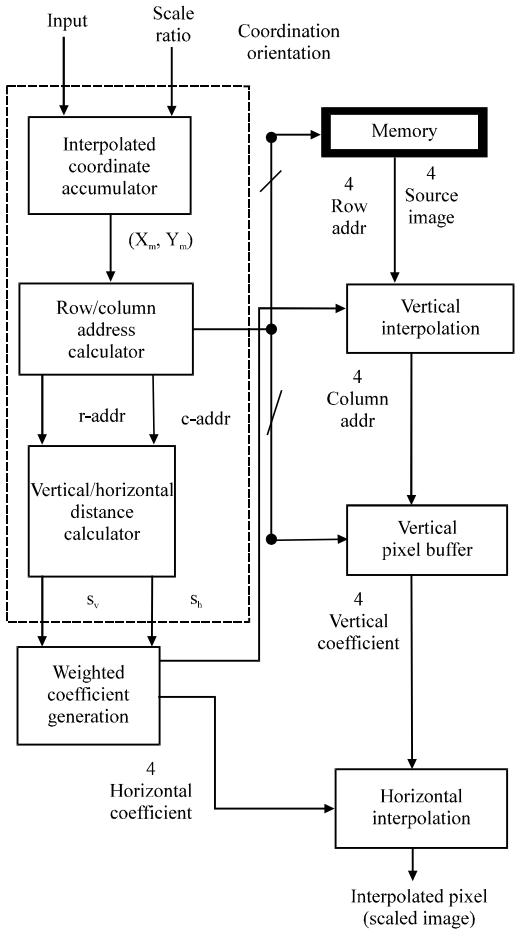

Fig. 1: The block diagram of the hardware architecture for digital image scaling

EXISTING WCG CIRCUIT FOR FFOPCI

In 1-dimensional image interpolation, vertical as well as horizontal weighting coefficients are required to be determined. The way of determining vertical weighting coefficients is indistinguishable to the way of acquiring horizontal weighting coefficients. In accordance with Fig. 1, equation can be modified to discover all the vertical weighting coefficients. The vertical weighting coefficients are established out by using vertical distance in Eq. 6 as:

$$
\begin{aligned}
v_i(1+s_v) &= \begin{cases} -0.375s_v & 0 \le s_v < \frac{1}{3} \\ -0.1875 + 0.1875s_v & \frac{1}{3} \le s_v < 1 \end{cases} \\
v_{i+1}(s_v) &= \begin{cases} 1 - 0.625s_v & 0 \le s_v < \frac{1}{3} \\ 1.1875 - 1.1875s_v & \frac{1}{3} \le s_v < 1 \end{cases} \\
v_{i+2}(1-s_v) &= \begin{cases} 1.1875s_v & 0 \le s_v < \frac{2}{3} \\ 0.375 - 0.625s_v & \frac{2}{3} \le s_v < 1 \end{cases} \\
v_{i+3}(2-s_v) &= \begin{cases} -0.1875s_v & 0 \le s_v < \frac{2}{3} \\ -0.375 + 0.375s_v & \frac{2}{3} \le s_v < 1 \end{cases}
\end{aligned} \tag{8}
$$

where, v_i, v_{i+1}, v_{i+2} and v_{i+3} are the vertical weighting coefficients of the consequent source pixels $A_{i,j}$, $A_{i+1,j}$, $A_{i+2,j}$ and $A_{i+3,j}$. s_v is the space between the source pixel $A_{i+1,j}$ and the virtual interpolated pixel Pj. Likewise, all the horizontal weighting coefficients can be established by Eq. 9:

$$
\begin{aligned}
h_j(1+s_h) &= \begin{cases} -0.375s_h & 0 \le s_h < \frac{1}{3} \\ -0.1875 + 0.1875s_h, & \frac{1}{3} \le s_h < 1 \end{cases} \\
h_{j+1}(s_h) &= \begin{cases} 1 - 0.625s_h & 0 \le s_h < \frac{1}{3} \\ 1.1875 - 1.1875s_h & \frac{1}{3} \le s_h < 1 \end{cases} \\
h_{j+2}(1-s_h) &= \begin{cases} 1.1875s_h & 0 \le s_h < \frac{2}{3} \\ 0.375 - 0.625s_h & \frac{2}{3} \le s_h < 1 \end{cases} \\
h_{j+3}(2-s_h) &= \begin{cases} -0.1875s_h & 0 \le s_h < \frac{2}{3} \\ -0.375 + 0.375s_h & \frac{2}{3} \le s_h < 1 \end{cases}
\end{aligned} \tag{9}
$$

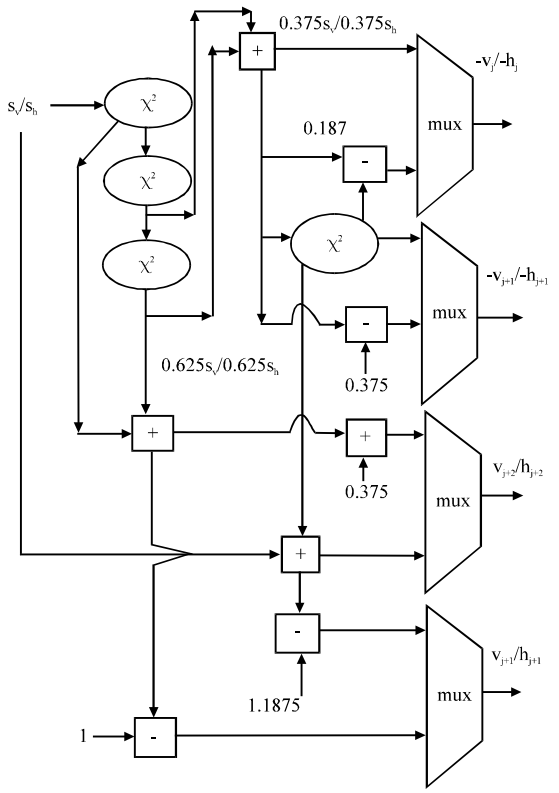

Fig. 2: Weighting coefficient generation circuit of FFOPCI

The existing hardware structural design for weighting coefficient generation is exposed in Fig. 2 (Lin *et al*., 2010). The existing WCG has 12 arithmetic elements. This structural design has answered the trouble of blur and blocking outcome and enhances the excellence of image. The mainly calculation attempt in FFOPCI is the computation of interpolation weighting coefficients. An area efficient architecture is proposed to overcome these computational efforts.

PROPOSED ARCHITECTURE

In Computational Comlexity Theory, arithmetic circuits are the standard model for computing the complexity in a logic circuit. Arithmetic circuits give the formal way for understanding the compexity of computing polynomials.

The existing hardware architecture of FFOPCI consists of 12 arithmetic circuits which increase the computational burden and memory requirements of generating weighting coefficients. So, simplification of the weighting coefficient circuit is needed to reduce the computational burden of generating the weighting coefficient.

The simplification is based on reducing the area utilized by the weighting coefficient circuit. Reduction in chip area reduces the memory requirementsAnd hardware cost. The existing weighting coefficient circuit of FFOPCI algorithm consists of four scalers, four adders and four sub tractors. The four scaling factors are 0.5, 0.25, 0.125 and 0.1875. The proposed architecture is shown in Fig. 3. In the proposed circuit, the scaling factor 0.125 was combined with 0.5 to produce the scaling factor 0.625 but the scaling factor 0.5 is used as such. Hence, the number of scalers used is reduced to three in this proposed architecture. By using the scaler 0.375, the adder which is used for generating the weight can be avoided. Thus, the number of adders used in the proposed method is also reduced to three. The number of subtractors used is four which is same as that of the exiting architecture.

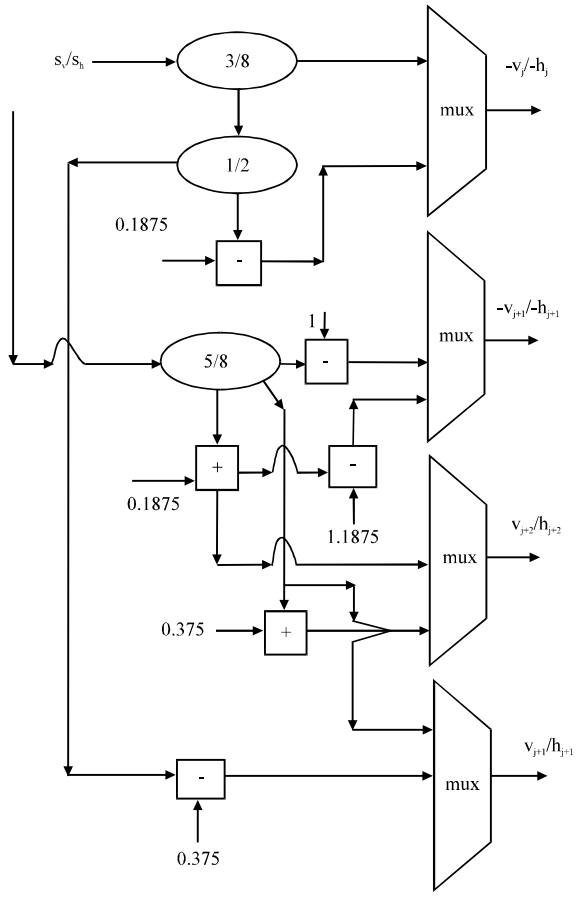

Fig. 3: Proposed weighting coefficient generation circuit of FFOPCI

IMPLEMENTATION RESULTS

The weighting coefficient generation circuit of Keys, Bi-Cubic, Fast First Order Polynomial algorithms and proposed architecture was simulated using Matlab Simulink tool and synthesized using Xilinx ISE Synthesis tool. The device chosen to synthesize the WCG circuit is virtex-6 xc6vsx315t-3ff1156. The WCG circuits are synthesized using Xilinx ISE 12.3. The design reports for the weighting coefficient generation circuit of Keys, Bi-Cubic, Fast First Order Polynomial algorithms includes the overall device utilization summary and total memory usage of weighting coefficient circuit.

For the generated Register Transfer Level (RTL) schematic of WCG circuit for keys, Bi-Cubic, Fast First Order Polynomial Convolution and the proposed method the synthesis report is found out and the area dependent parameters are compared. The synthesis report shows the results of the netlist-generation synthesis process. The netlist is generated using system generator and the device utilization summary and memory usage can be found out after synthesizing and implementing the design. Table 1 shows the number of arithmetic circuits used in weighting coefficient generation circuit for different scaling methods. The weighting coefficient generation circuit for the proposed method consists of only nine arithmetic circuits.

The number of hardware components required for computing the weighting coefficients very much less for the proposed method compared to Keys, Bi-cubic and Fast First Order Polynomial Convolution Interpolation Method. Using the proposed architecture in Fast First Order Polynomial Convolution Interpolation Method high quality scaled images with low computational complexity can be obtained. Results obtained from device utilization summary and synthesis report of weighting coefficient generation circuit for the existing and proposed method is shown in Table 2 and the performance is analyzed based on area dependent parameters.

Table 1: Number of arithmetic circuits used in weighting coefficient generation circuit of different types of interpolation methods

	WCG architecture			
Arithmetic circuits	Keys (Lin *et al*., 2008a)	Bi-Cubic (Lin *et al*., 2008b)	FFOPCI (Lin *et al*., 2010)	Proposed
Adders	8	8	4	2
Dividers	-	-	4	3
Multipliers	18	14	-	-
Subtractors	5	4	4	4
Total arithmetic circuits	31	26	12	9

Table 2: Comparison of area dependent parameters for various interpolation methods

	WCG architecture			
Parameters	Keys (Lin *et al*., 2008a)	Bi-Cubic (Lin *et al*., 2008b)	FFOPCI (Lin *et al*., 2010)	Proposed
No. of slice registers (out of 393600)	66 (1%)	17 (1%)	0 (0%)	0 (0%)
No. of slice LUTs (out of 196800)	341 (1%)	283 (1%)	132 (1%)	113 (1%)
No. of occupied slices (out of 49200)	99 (1%)	82 (1%)	36 (1%)	30 (1%)
Memory usage (kilo bytes)	274512	163324	164620	162800

The specifications such as device utilization summary and memory usage are being found out from the synthesis report. The area dependent parameters are less for the proposed architecture of WCG for FFOPCI algorithm.

CONCLUSION

In this study an area efficient weighting coefficient circuit for Fast First Order Polynomial Convolution Interpolation algorithm was presented. This architecture has the advantages of low operation complexity in weighting coefficient generation effort. In the proposed hardware architecture, the weighting coefficient generation circuit has decreased the hardware complexity significantly. The area dependent parameters of FPGA based design such as number of slice registers used, number Look Up Tables (LUTs), total number of occupied slices and memory usage are being found out from the synthesis Report and Compared with Keys, Bi-cubic and FFOPCI Methods. All area dependent parameters are less for the proposed architecture. Thus, the proposed WCG can be used to implement digital image scaling circuits with less area and low complexity.

REFERENCES

Chen, S.L., 2013a. VLSI implementation of a low-cost high-quality image scaling processor. IEEE Trans. Circ. Syst.: Express Briefs, 60: 31-35.

Chen, S.L., 2013b. VLSI implementation of an adaptive edge-enhanced image scalar for real-time multimedia applications. IEEE Trans. Circ. Syst. Video Technol., 23: 1510-1522.

Chen, S.L., H.Y. Huang and C.H. Luo, 2011. A low-cost high-quality adaptive scalar for real-time multimedia applications. IEEE Trans. Circ. Syst. Video Technol., 21: 1600-1611.

Fry, J. and M. Pusateri, 2009. A comparison of bicubic and biquintic interpolators suitable for real-time hardware implementation. Proceedings of the Applied Imagery Pattern Recognition Workshop, October 14-16, 2009, Washington, DC., USA., pp: 1-5.

Hwang, J.W. and H.S. Lee, 2004. Adaptive image interpolation based on local gradient features. IEEE Signal Process. Lett., 11: 359-362.

Kim, C.H., S.M. Seong, J.A. Lee and L.S. Kim, 2003. Winscale: An image-scaling algorithm using an area pixel model. IEEE Trans. Circ. Syst. Video Technol., 13: 549-553.

Lin, C.C., M.H. Sheu, C. Liaw and H.K. Chiang, 2010. Fast first-order polynomials convolution interpolation for real-time digital image reconstruction. IEEE Trans. Circ. Syst. Video Technol., 20: 1260-1264.

Lin, C.C., M.H. Sheu, H.K. Chiang, Z.C. Wu, J.Y. Tu and C.H. Chen, 2008a. A low-cost VLSI design of extended linear interpolation for real time digital image processing. Proceedings of the International Conference on Embedded Software and Systems, July 29-31, 2008, Sichuan, China, pp: 196-202.

Lin, C.C., M.H. Sheu, H.K. Chiang, W.K. Tsai and Z.C. Wu, 2008b. Real-time FPGA architecture of extended linear convolution for digital image scaling. Proceedings of the International Conference on ICECE Technology, December 8-10, 2008, Taipei, Taiwan, pp: 381-384.

Lin, C.C., M.H. Sheu, H.K. Chiang, C. Liaw and Z.C. Wu, 2008c. The efficient VLSI design of BI-CUBIC convolution interpolation for digital image processing. Proceedings of the IEEE International Symposium on Circuits and Systems, May 18-21, 2008, Seattle, WA., USA., pp: 480-483.

Lin, C.C., Z.C. Wu, W.K. Tsai, M.H. Sheu and H.K. Chiang, 2007. The VLSI design of winscale for digital image scaling. Proceedings of the 3rd International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Volume 2, November 26-28, 2007, Kaohsiung, Taiwan, pp: 511-514.

Madani, R., A. Ayremlou, A. Amini and F. Marvasti, 2012. Optimized compact-support interpolation kernels. IEEE Trans. Signal Process., 60: 626-633.

Meijering, E.H. and M. Unser, 2003. A note on cubic convolution interpolation. IEEE Trans. Image Process., 12: 477-479.

Meijering, E.H., K.J. Zuiderveld and M.A. Viergever, 1999. Image reconstruction by convolution with symmetrical piecewise nth-order polynomial kernels. IEEE Trans. Image Process., 8: 192-201.

Miklos, P., 2007. Comparison of convolutional based interpolation techniques in digital image processing. Proceedings of the 5th International Symposium on Intelligent Systems and Informatics, August 24-25, 2007, Subotica, Serbia, pp: 87-90.

Moses, C.J., D. Selvathi and S.S. Rani, 2010. FPGA implementation of an efficient partial volume interpolation for medical image registration. Proceedings of the IEEE International Conference on Communication Control and Computing Technologies, October 7-9, 2010, Ramanathapuram, India, pp: 132-137.

Moses, C.J., D. Selvathi, J.P. Beena and S.S. Rani, 2011. FPGA accelerated partial volume interpolation. Proceedings of the International Conference on Emerging Trends in Electrical and Computer Technology, March 23-24, 2011, Tamil Nadu, India, pp: 816-819.

Nuno-Maganda, M.A. and M.O. Arias-Estrada, 2005. Real-time FPGA-based architecture for bicubic interpolation: An application for digital image scaling. Proceedings of the International Conference on Reconfigurable Computing and FPGAs, September 28-30, 2005, Puebla, Mexico, pp: 1-8.

Shezaf, N., H. Abramov-Segal, I. Sutskover and R. Bar-Sella, 2000. Adaptive low complexity algorithm for image zooming at fractional scaling ratio. Proceedings of the 21st IEEE Convention of the Electrical and Electronic Engineers in Israel, April 11-12, 2000, Tel-Aviv, Israel, pp: 253-256.

Tsai, C.S., H.H. Liu and M.C. Tsai, 2011. Design of a scan converter using the cubic convolution interpolation with Canny edge detection. Proceedings of the International Conference on Electric Information and Control Engineering, April 15-17, 2011, Wuhan, China, pp: 5813-5816.

Xiao, J., X. Zou, Z. Liu and X. Guo, 2006. Adaptive interpolation algorithm for real-time image resizing. Proceedings of the 1st International Conference on Innovative Computing, Information and Control, Volume 2, August 30-September 1, 2006, Beijing, China, pp: 221-224.