Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 4, April 2013, pg.46 – 51

RESEARCH ARTICLE

Implementation of High-Performance Image Scaling Processor using VLSI

R.S. KARTHIC[1]

[1]Assistant Professor, Department of Electronics and Communication Engineering. PSNA College of Engineering and Technology, Dindigul, Tamilnadu, India

[1]*karthicrs@hotmail.com*

Abstract— In this paper, a less complexity, less memory requirement, and high performance algorithm is proposed for Very Large Scale Integration implementation of an image scaling processor. The anticipated image scaling algorithm consists of a clamp filter, spatial filter and a bilinear interpolation. The spatial and clamp filters are added as pre-filters for reducing the aliasing artifacts resulted by the bilinear interpolation. A T-model and inversed T-model convolution kernels are proposed to reduce the complexity of the design. Combined filter is replaced by a dynamic estimation unit to minimize the hardware cost. This architecture is targeted to produce 320MHz with 6.08-K gate counts. Compared with Previous methodologies, this work shows better performance with respect to cost and less complexity.

Key Terms: - Clamp filter; Image zooming; Dynamic estimation unit; Bilinear; Spatial filter; VLSI

I. INTRODUCTION

IMAGE scaling has been widely applied in the fields of digital imaging and mainly on Electronic based imaging devices. Image scaling is the process of scaling down the high – quality frames or pictures to fit small size LCD panel of electronic displays as digital PDA's are growing fast. Scaling algorithm can be classified into two types as polynomial-based and non-polynomial-based. Nearest neighbor algorithm is the uncomplicated polynomial algorithm, but resultant images are with full of aliasing artifacts. Bilinear interpolation algorithm [15] and Bi-cubic algorithm [14] are the other polynomial based methods widely used to target the pixels. For the past decade many non-polynomial high performance methods [1-3] have been proposed. The techniques like bilateral filter [2], interpolation [1], and autoregressive model [3]. These methods are used to boost the image quality by reducing the artifacts. These Image scaling algorithms are very complex to implement in VLSI. Thus, for fast, real time applications, less complexity based algorithms are necessary [5-9]. Area pixel model Winscale method is previously proposed for less complexity methods. Adding of sharpening spatial and clamp filers effectively improves the image quality with bilinear interpolation algorithm. By these cost of the hardware and memory also reduced.

© 2013, IJCSMC All Rights Reserved

II. PROPOSED SCALING ALGORITHM

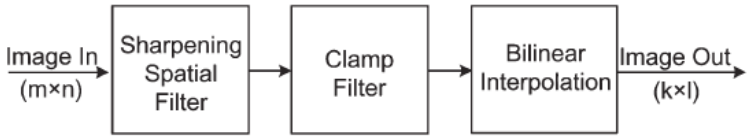

Fig. 1. Block diagram of the proposed scaling algorithm for image zooming.

Fig. 1 shows the block diagram of the proposed scaling algorithm. The sharpening spatial and clamp filters [5] act as pre- filters [4] to reduce blurring and aliasing artifacts produced by the bilinear interpolation. The input pixels of the original images are filtered by the spatial filter to remove associated noise and enhance the edges.

Unwanted discontinuous edges and boundaries are filtered by clamp filter. To conserve computing resource and memory buffer, these two filters are simplified and combined into a combined filter.

Less -Complexity Sharpening Spatial and Clamp Filters

The sharpening spatial filter, a kind of high-pass filter, is used to reduce blurring artifacts and defined by a kernel to increase the intensity of a center pixel relative to its neighboring pixels. The clamp filter a kind of low-pass filter is a 2-D Gaussian spatial domain filter and composed of a convolution kernel array. It usually contains a single positive value at the center and is completely surrounded by ones. The clamp filter is used to reduce aliasing artifacts and smooth the unwanted discontinuous edges of the boundary regions. The sharpening spatial and clamp filters can be represented by convolution kernels. A larger size of convolution kernel will produce higher quality of images. However, a larger size of convolution filter will also demand more memory and hardware cost. For example, a 6×6 convolution filter demands at least a five-line-buffer memory and 36 arithmetic units, which is much more than the two-line-buffer memory and nine arithmetic units of a 3×3 convolution filter. In our previous work], each of the sharpening spatial and clamp filters was realized by a 2-D 3×3 convolution kernel as shown in Fig. 2(a). It demands at least a four-line-buffer memory for two 3×3 convolution filters. For example, if the image width is 1920 pixels, $4 \times 1920 \times 8$ bits of data should be buffered in memory as input for processing. To reduce the complexity of the 3×3 convolution kernel, a cross-model formed is used to replace the 3×3 convolution kernel. It successfully cuts down on four of nine parameters in the 3×3 convolution kernel. Furthermore, to decrease more complexity and memory requirement of the cross model convolution kernel, T-model and inversed T-model convolution kernels are proposed for realizing the sharpening spatial and clamp filters. The T-model convolution kernel is composed of the lower four parameters of the cross-model, and the inversed T-model convolution kernel is composed of the upper four parameters. In the proposed scaling algorithm, both the T-model and inversed T-model filters are used to improve the quality of the images simultaneously. The T-model or inversed T-model filter is simplified from the 3×3 convolution filter of the previous work, which not only efficiently reduces the complexity of the convolution filter but also greatly decreases the memory requirement from two to one line buffer for each convolution filter. The T-model and the inversed T-model provide the low-complexity and low memory- requirement convolution kernels for the sharpening spatial and clamp filters to integrate the VLSI chip of the proposed low-cost image scaling processor.

Combined Filter

In proposed scaling algorithm, the input image is filtered by a sharpening spatial filter and then filtered by a clamp spatial filter again. Although the sharpening spatial and clamp filters are simplified by T-models and inversed T-models, it still needs two line buffers to store input data or intermediate values for each T-model or inversed T-model filter. Thus, to be able to reduce more computing resource and memory requirement, sharpening spatial and clamp filters, which are formed by the T-model or inversed T-model, should be combined together into a combined filter as

© 2013, IJCSMC All Rights Reserved

$$P'_{(m,n)} = \left[P^*_{(m,n)} \begin{bmatrix} -1 & S & -1 \\ & -1 & \end{bmatrix} /(S-3)\right]^* \begin{bmatrix} 1 & C & 1 \\ & 1 & \end{bmatrix} /(C+3)$$

$$= P^*_{(m,n)} \begin{bmatrix} -1 & S-C & SC-2 & S-C & -1 \\ & -2 & S-C & -2 & \\ & & -1 & & \end{bmatrix} / [(S-3)\times(C+3)] \quad (1)$$

$$\approx P^*_{(m,n)} \begin{bmatrix} -1 & S-C & SC-2 & S-C & -1 \\ & -2 & S-C & -2 & \end{bmatrix} / [(S-3)\times(C+3)] \quad (2)$$

Where S and C are the sharp and clamp parameters and *P*(m,n) is the filtered result of the target pixel *P*(m,n) by the combined filter. A T-model sharpening spatial filter and a T-model clamp filter have been replaced by a combined T-model filter as shown in (1). To reduce the one-line-buffer memory, the only parameter in the third line, parameter −1 of *P*(m,n−2), is removed, and the weight of parameter −1 is added into the parameter S-C of *P*(m,n−1) by S-C-1 as shown in (2). The combined inversed T-model filter can be produced in the same way. In the new architecture of the combined filter, the two T-model or inversed T-model filters are combined into one combined T-model or inversed T-model filter. By this filter-combination technique, the demand of memory can be efficiently decreased from two to one line buffer, which greatly reduces memory access requirements for software systems or hardware memory costs for VLSI implementation.

Bilinear Interpolation

In the proposed scaling algorithm, the bilinear interpolation method is selected because of its characteristics with low complexity and high quality. The bilinear interpolation is an operation that performs a linear interpolation first in one direction and, then again, in the other direction. The output pixel P(k,l) can be calculated by the operations of the linear interpolation in both x- and y-directions with the four nearest neighbour pixels. We can easily find that the computing resources of the bilinear interpolation cost eight multiply, four subtract, and three addition operations. It costs a considerable chip area to implement a bilinear interpolator with eight multipliers and seven adders. Thus, an algebraic manipulation skill has been used to reduce the computing resources of the bilinear interpolation.

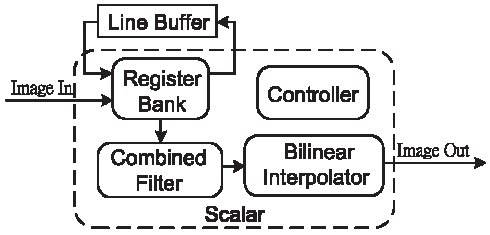

Fig.2. Block diagram of the VLSI architecture for proposed real-time image scaling processor

The original equation of bilinear interpolation is presented and the simplifying procedures of bilinear interpolation can be described. Since the function of dy × (P(m,n+1) − P(m,n)) + P(m,n) appears twice in, one of the two calculations for this algebraic function can be reduced by the characteristic of the executing direction in bilinear interpolation [15], the values of dy for all pixels that are selected on the vertical axis of n row equal to n + 1 row, and only the values of dx must be changed with the position of x. The result of the function "[P(m,n) + dy × (P(m,n+1) − P(m,n))]" can be replaced by the previous result of "[P(m+1,n) + dy × (P(m+1,n+1) − P(m+1i,n))]" as shown in (6). The simplifying procedures successfully reduce the computing resource from eight multiply, four subtract, and three add operations to two multiply, two subtract, and two add operations.

III. VLSI ARCHITECTURE

The proposed scaling algorithm consists of two combined pre-filters and one simplified bilinear interpolator. For VLSI implementation, the bilinear interpolator can directly obtain two input pixels.

Register Bank

In this brief, the combined filter is filtering to produce the target pixels of *P*_(m,n) and *P*_(m,n+1) by using ten source pixels. The register bank is designed with a one-line memory buffer, which is used to provide the ten

© 2013, IJCSMC All Rights Reserved

values for the immediate usage of the combined filter. Fig. 2 shows the architecture of the register bank with a structure of ten shift registers.

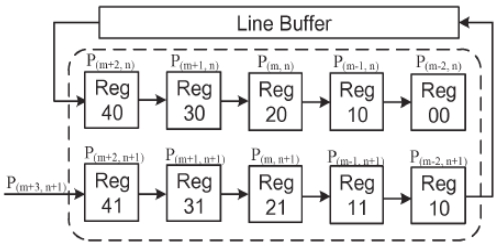

Fig.3. Architecture of the register bank

When the shifting control signal is produced from the controller, a new value of *P*(m+3,n) will be read into Reg41, and each value stored in other registers belonging to row *n* + 1 will be shifted right into the next register or line-buffer memory. The Reg40 reads a new value of *P*(m+2,n) from the line-buffer memory, and each value in other registers belonging to row n will be shifted right into the next register.

Combined Filter

The combined T-model or inversed T-model convolution function of the sharpening spatial and clamp filters had been discussed in Section II, and the equation is represented in (1). Fig. 4 shows the six-stage pipelined architecture of the combined filter and bilinear interpolator, which shortens the delay path to improve the performance by pipeline technology. The stages 1 and 2 in Fig. 4 show the computational scheduling of a T-model combined and an inversed T-model filter. The T-model or inversed T-model filter consists of three reconfigurable calculation units (RCUs), one multiplier–adder (MA), three adders (+), three subtracters (−), and three shifters (S). The hardware architecture of the T-model combined filter can be directly mapped with the convolution equation shown in (1). The values of the ten source pixels can be obtained from the register bank mentioned earlier.

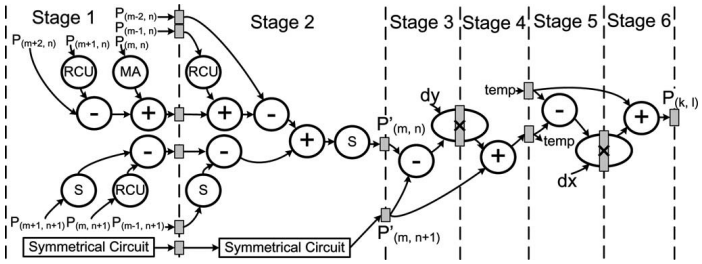

Fig.4. Computational scheduling of the proposed combined filter and simplified bilinear interpolator

The symmetrical circuit, as shown in stages 1 and 2 of Fig.3, is the inversed T-model combined filter designed for producing the filtered result of $p\_$ (m,n+1). Obviously, The T-model and the inversed T-model are used to obtain the values of $p\_$ (m,n) and $p\_{(m,n+1)}$ simultaneously. The architecture of this symmetrical circuit is a similar symmetrical structure of the T-model combined filter, as shown in stages 1 and 2 of Fig. 3. Both of the combined filter and symmetrical circuit consist of one MA and three RCUs. The MA can be implemented by a multiplier and an Adder. The RCU is designed for producing the calculation functions of (S-C) and (S-C-1) times of the source pixel value, which must be implemented with C and S parameters. The C and S parameters can be set by users according to the characteristics of the images.

TABLE I
PARAMETERS AND COMPUTING RESOURCE FOR RCU

Parameters	Values	Computing Resource
C	5, 13, 29	Add and Shift
S	7, 11, 19	Add and Shift
S-C	2, -6, -22, 6, -2, -18, 14, 6, -10	Add, Shift, and Sign
S-C-1	1, -7, -23, 5, -3, -19, 13, 5, -11	Add, Shift, and Sign

© 2013, IJCSMC All Rights Reserved

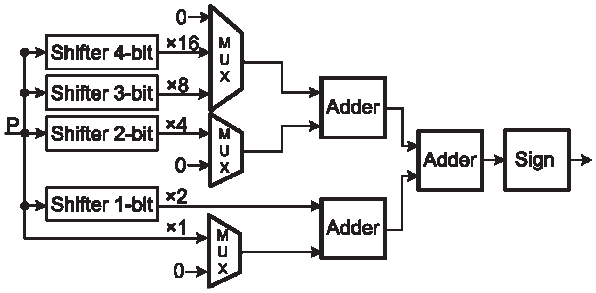

Fig.5. Architecture of the RCU

Table I lists the parameters and computing resource for the RCU. With the selected C and S values listed in Table I, the gain of the clamp or sharp convolution function is {8, 16, 32} or {4, 8, 16}, which can be eliminated by a shifter rather than a divider. Fig. 6 shows the architecture of the RCU. It consists of four shifters, three multiplexers (MUX), three adders, and one sign circuit. By this RCU design, the hardware cost of the combined filters can be efficiently reduced.

Bilinear Interpolator and Controller

In the previous discussion, the bilinear interpolation is simplified as shown in (5). The stages 3, 4, 5, and 6 in Fig. 4 show the four-stage pipelined architecture, and two-stage pipelined multipliers are used to shorten the delay path of the bilinear interpolator. The input values of P_ (m,n) and P_ (m,n+1) are obtained from the combined filter and symmetrical circuit. By the hardware sharing technique, as shown in (4), the temperature result of the function "P_ (m,n) + dy × (P_ (m,n+1) − P_ (m,n))" can be replaced by the previous result of

"P_(m+1,n) + dy × (P_(m+1,n+1)− P_(m+1i,n))." It also means that one multiplier and two adders can be successfully reduced by adding only one register. The controller is implemented by a finite-state machine circuit. It produces control signals to control the timing and pipeline stages of the register bank, combined filter, and bilinear interpolator.

IV. CONCLUSION

In this brief, a low-cost, low-memory-requirement, high quality, and high-performance VLSI architecture of the image scaling processor had been proposed. The filter combining, hardware sharing, and reconfigurable techniques had been used to reduce hardware cost. Relative to previous low-complexity VLSI scalar designs, this work achieves at least 36.5% reduction in gate counts and requires only one-line memory buffer.

REFERENCES

[1] H. Kim, Y. Cha, and S. Kim, "Curvature interpolation method for image zooming," IEEE Trans. Image Process., vol. 20, no. 7, pp. 1895–1903, Jul. 2011.

[2] J. W. Han, J. H. Kim, S. H. Cheon, J.O.Kim, and S. J. Ko, "Anovel image interpolation method using the bilateral filter," IEEE Trans. Consum.Electron., vol. 56, no. 1, pp. 175–181, Feb. 2010.

[3] X. Zhang and X.Wu, "Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation," IEEE Trans. Image Process., vol. 17, no. 6, pp. 887–896, Jun. 2008.

[4] F. Cardells-Tormo and J. Arnabat-Benedicto, "Flexible hardware-friendly digital architecture for 2-D separable convolution-based scaling," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 7, pp. 522–526, Jul. 2006.

[5] S. Ridella, S. Rovetta, and R. Zunino, "IAVQ-interval-arithmetic vector quantization for image compression," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 47, no. 12, pp. 1378–1390, Dec. 2000.

[6] S. Saponara, L. Fanucci, S. Marsi, G. Ramponi, D. Kammler, and E. M. Witte, "Application-specific instruction-set processor for Retinexlink image and video processing," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 54, no. 7, pp. 596–600, Jul. 2007.

[7] P. Y. Chen, C. C. Huang, Y. H. Shiau, and Y. T. Chen, "A VLSI implementation of barrel distortion correction for wide-angle camera images," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 56, no. 1, pp. 51–55, Jan. 2009.

[8] M. Fons, F. Fons, and E. Canto, "Fingerprint image processing acceleration through run-time reconfigurable hardware," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 57, no. 12, pp. 991–995, Dec. 2010.

[9] C. H. Kim, S. M. Seong, J. A. Lee, and L. S. Kim, "Winscale : An imagescaling algorithm using an area pixel model," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 6, pp. 549–553, Jun. 2003.

© 2013, IJCSMC All Rights Reserved

[10] C. C. Lin, Z. C. Wu, W. K. Tsai, M. H. Sheu, and H. K. Chiang, "The VLSI design of winscale for digital image scaling," in Proc. IEEEInt. Conf. Intell. Inf. Hiding Multimedia Signal Process., Nov. 2007, pp. 511–514.
[11] P. Y. Chen, C. Y. Lien, and C. P. Lu, "VLSI implementation of an edgeoriented image scaling processor," IEEE Trans. Very Large Scale Integr.(VLSI) Syst., vol. 17, no. 9, pp. 1275–1284, Sep. 2009.
[12] C. C. Lin, M. H. Sheu, H. K. Chiang, W. K. Tsai, and Z. C. Wu, "Real-time FPGA architecture of extended linear convolution for digital image scaling," in Proc. IEEE Int. Conf. Field-Program. Technol., 2008, pp. 381–384.
[13] C. C. Lin, M. H. Sheu, H. K. Chiang, C. Liaw, and Z. C. Wu, "The efficient VLSI design of BI-CUBIC convolution interpolation for digital image processing," in Proc. IEEE Int Conf. Circuits Syst., May 2008, pp. 480–483.
[14] S. L. Chen, H. Y. Huang, and C. H. Luo, "A low-cost high-quality adaptive scalar for real-time multimedia applications," IEEE Trans. Circuits Syst.Video Technol., vol. 21, no. 11, pp. 1600–1611, Nov. 2011.
[15] K. Jensen and D. Anastassiou, "Subpixel edge localization and the interpolation of still images," IEEE Trans. Image Process., vol. 4, no. 3, pp. 285–295, Mar. 1995.

© 2013, IJCSMC All Rights Reserved