Analysis of Interpolation Techniques for Image Scaling based on IET and its Variants

Nakul E
Student
Viswajyothi College of Engineering and Technology
Vazhakulam

Merlin Thomas
Assistant Professor
Viswajyothi College of Engineering and Technology
Vazhakulam

ABSTRACT
Image scaling is an important technique that is widely used in many image processing applications. Here the interpolation technique is based on the interpolation error theorem and a bilateral error-amender is used to interpolate the resampled pixels. Interpolation technique based on IET is compared with conventional interpolation techniques such as Bilinear and Nearest Neighbour interpolation. The 3 methods are compared for visual quality and hardware utilization. Interpolation technique based on IET shows better performance than the existing methods. It solved the blurring and checkerboard effects to an extent. The visual quality is observed using MATLAB. The design is synthesized using Xilinx ISE software. Simulation result is found using Modelsim software. The hardware implementation is done in Spartan 3E FPGA kit.

Keywords
Image Scaling,Interpolation,Very Large Scale Integration(VLSI).

1. INTRODUCTION
Image scaling is widely used in many fields, ranging from consumer electronics such as digital cameras, mobile phones, tablets, and display devices to medical imaging such as endoscopy [1], [2]. An image may be resampled to a finer matrix in order to improve its appearance for image display .This technique becomes indispensable when the resolution of an image differs from the screen resolution of a target display. Many interpolation methods have been proposed for image scaling [2]. These interpolation methods generally fall into two classes: spatial domain and frequency domain techniques.The most widely used spatial domain interpolation methods are the nearest neighbour and bilinear techniques [3],[4]. They afford the benefits of low complexity and ease of implementation, but leave the scaled image with checkerboard and blurring effect . Bicubic (BC) method has the benefit of excellent image quality [5]-[9]. Another study proposed a simple area-pixel interpolation method (Winscale) that uses an area-pixel model instead of the common point pixel model [10], [11].A low-cost, edge-oriented, area-pixel interpolation method appears in [12]. This method adopts a simple edge catching technique to preserve the edge features efficiently. Cha and Kim [13] proposed a new interpolation method, called the error-amended sharp edge (EASE) scheme. Another study proposed an isophote-oriented, orientation adaptive scaling method [14].Here interpolation error theorem and a bilateral error-amender is used to interpolate the resampled pixels [15]. The co-operation and hardware sharing techniques decrease the computing resources and hardware cost. The objective of this paper is to develop an interpolation technique based on IET and to prove that it provide much better visual quality compared to existing interpolation techniques like nearest neighbour and bilinear interpolation
The paper is organized as follows. Section 2 presents the image scaling techniques. The experimental results and comparison are analyzed in Section 3. Finally, the work is concluded in section 4.

2. IMAGE SCALING TECHNIQUES

2.1 Nearest Neighbour Algorithm
Nearest neighbour is the most basic and requires the least processing time of all the interpolation algorithms because it only considers one pixel, the closest one to the interpolated point. This has the effect of simply making each pixel bigger. Here the output pixel is assigned the value of the pixel that the point falls within. No other pixels are considered .It suffers from checkerboard effect.

2.2 Bilinear Interpolation
Bilinear interpolation considers the closest 2x2 neighborhood of known pixel values surrounding the unknown pixel. It then takes a weighted average of these 4 pixels to arrive at its final interpolated value. This results in much smoother looking images than nearest neighbour. It requires higher computation.

2.3 Interpolation based on IET
Let u(x) represents a multi-times continuously differentiable function on the bounded interval $[a, b]$. Let the interval be partitioned into $\{a = x_0 < x_1 < ... < x_N = b\}$, where an Nth-order polynomial P_N interpolates u at the nodal points of the partitioning and each partition is equidistant. Assume that $u^{(N+1)}(x)$ exists for each $x \sum [a, b]$.Then, for every $x \sum [a, b]$, there is a point $\varepsilon_x \sum [a, b]$, such that

$$U(x)\text{-}P_N(x) = \frac{u^{(N+1)\varepsilon_x}}{(N+1)!} \prod_{j=0}^{N}(x - xj) \qquad (1)$$

This is called interpolation error theorem.Based on the interpolation error theorem, it is possible to calculate the linear interpolation error as follows. Assume that $P_N(x)$ mentioned in (1) is a low-complexity interpolation method, and let $N = 1.h$ be an interval length. There is a point $\in_x$ between xi and $xi + h$ such that

$u(x)-[u(xi)+\frac{u(xi+h)-u(xi)}{h}(x-xi)] = \frac{u"(\in x)}{2}(x-xi)[x-(xi+h)]$ (2)

where $P_1(x) = u(xi)+\frac{u(xi+h)-u(xi)}{h}(x-xi)$

For $x=xi+s\times h$, where $0 < s < 1$ and $s \in R$, note that

$$u\,(xi+s\times h - [(1-s)u(xi) + s \times u(xi+h)] = -u"(\frac{\in_{xi+s\times h}}{2})\ s(1-s)h^2 \quad (3)$$

for some $\in_{xi+s\times h} \in (x_i, x_{i+h})$. The second derivative term can be denoted as the linear interpolation error. Based on the interpolation error theorem and bilateral error-amender, this paper develops an efficient interpolation kernel for image scaling.

First, rewrite (3) and define the interpolation function as

$$u(xi+s\times h) = [(1-s)u(xi) + s \times u(xi+h)] - \frac{u"\varepsilon_{xi+s\times h)}}{2}s(1-s)h^2 \quad (4)$$

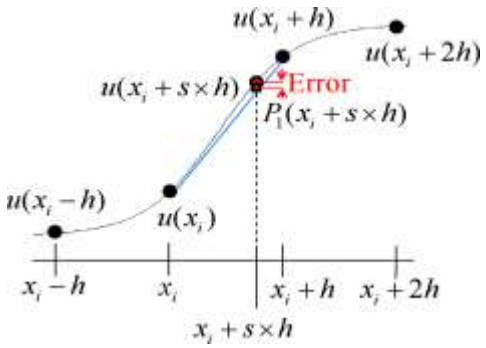

Figure 1: Example of linear interpolation error

For any interpolation in the bounded interval, let interval length $h = 1$ and $x_{i+s}\varepsilon[x_i, x_{i+1}]$ equation (4) indicates that

$$u(x_{i+s}) = [(1-s)u(x_i) + s \times u(x_{i+1})] - u"(\varepsilon_{x_{i+1}})s(1-s) \quad (5)$$

According to (5), only the value of the second derivative $(u"(\varepsilon_{x_{i+s}}))$ is unknown. The interpolation error theorem indicates that $u(x)$ is a real multi-times continuously differentiable function. If the function of the second derivative is a smooth curve, then $u"(\varepsilon_{x_{i+1}})$ can be interpolated using the interpolation errors of two nearest nodes. Figure 4 illustrates how to calculate the values of the second derivatives on x_i and x_{i+1}. Consider node x_{i+1} first. For $h = 2, s = 1/2$, which means $u(x_{i+1})$is the midpoint of $u(x_i)$ and $u(x_{i+2})$, it follows from (3) that

$$(x_{i+1}) - \frac{u(x_i)+u(x_{i+2})}{2} = -\frac{u"(\varepsilon_{x_{i+1}})}{2} = E^R \quad (6)$$

where $E^R$is an error-amender on the node x_{i+1}.Similarly, the same method can be applied to node x_i to obtain E^L, the error-amender on the node x_i, as follows:

$$u(x_i) - \frac{u(x_{i-1})+u(x_{i+1})}{2} = -u"(\varepsilon_{x_i}) = E^L \quad (7)$$

Thus, the bilateral error-amenders E^R and E^L can be used to interpolate $u"(\varepsilon_{x_{i+s}})$.As mentioned above, $u(x)$ is a real multi-times continuously differentiable function, and so is the second order of $u(x)$. According to the interpolation error theorem, the value of a continuously differentiable function can be interpolated using the interpolation errors of its two nearest nodes. This makes it possible to rewrite (5) in the second-order form as

$$u"(x_{i+s}) = (1-s)u"(x_i) + s \times u"(x_{i+1}) - \frac{u^{(4)}(\in_{x_{i+s}})}{2}s(1-s) \quad (8)$$

As Fig. 2 shows, the curve is smooth, so in most cases$u^{(4)}(\in_{x_{i+s}})$ is small enough to be neglected. Besides, some high complexity computation modules are required to

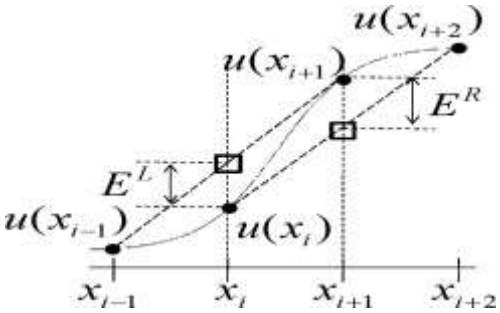

Figure 2: Example to obtain the second derivatives

calculate the fourth derivatives $(u^{(4)}(\in_{x_{i+s}}))$ and to implement it with a hardware circuit. Hence, we ignore the fourth derivatives to reduce the hardware cost. Use two nearest nodes $u"(\in_{x_i})$ and $u"(\in_{x_{i+1}})$ to estimate $u"(\in_{x_{i+1}})$ and rewrite (8) as $u(\in_{x_{i+s}})\approx$(1-s)u$(\in_{x_i}) + s \times u"(\in_{x_{i+1}})$.By replacing $u"(\in_{x_{i+s}})$ with $(1-s)u(\in_{x_i})$+s×u$(\in_{x_{i+1}})$,(5) can be rewritten as

$$u_{new}\,(1-s)u(x_i) + s \times u(x_{i+1}) - \frac{(1-s)u(\in_{x_i})+s\times u"(\in_{x_{i+1}})}{2}s(1-s) \quad (9)$$

The bilateral error-amenders E^R and E^L can then be used to replace the values of $u"(u_{x_i})/2$ and $u"(\in_{x_{i+1}})/2$, respectively.Equation (9) then shows that

$$u_{new} = (1-s)u(x_i) + s \times u(x_{i+1}) + \big((1-s)E^L + s \times E^R\big)s(1-s) \quad (10)$$

Note that$(1-s)u(x_i) + s \times u(x_{i+1})$ is a linear interpolation function. The proposed method adopts the concept of the edge weighted scheme to enhance edge features and modify as$u_{new} = (1-s)u(x_i) + s \times u(x_{i+1}) + \gamma[(1-s)E^L + s \times E^R]s(1-s)$ (11)

where γ is an edge-weighted coefficient.

Rewrite (11) as

$$u_{new} = (1-s)u(x_i) + s \times u \times [(1-s)E^L + s \times E^R]s(1-s)$$

$$= (1-s)u(x_i) + s \times u(x_{i+1}) + [(1-s)(2E^L) + s \times (2E^R)]s(1-s) \quad (12)$$

Equations (6) and (7) show that $2E^R$ and $2E^L$ can be given by

$$2E^R = 2u(x_{i+1}) - u(x_i) - u(x_{i+2}) \quad (13)$$

(13)

$$2E^R = 2u(x_i) - u(x_{i-1}) - u(x_{i+1}) \quad (14)$$

Use equation (13) and (14) to replace $2E^R$ and $2E^L$ in (12) respectively. Rearranging this equation leads to

$$u_{new} = -(1-s)^2 s \times u(x_{i-1}) + [(1-s) + 2(1-s)^2 s - (1-s)s^2] \times u(x_i) + [s + 2(1-s)s^2 - (1-s)^2 s] \times u(x_{i+1}) - (1-s)s^2 \times (x_{i+2}) \quad (15)$$

Next separate weights from (15) and define the convolution interpolation kernel as

$$C_o(s) = -(1-s)^2 s$$

$$C_1(s) = (1-s) + 2(1-s)^2 s - (1-s)s^2$$

$$C_2(s) = s + 2(1-s)s^2 - (1-s)^2 s$$

$$C_3(s) = -(1-s)s^2 \quad (16)$$

Finally the interpolated luminance values $u(x_{i+s})$ can be given by

$$u(x_{i+s}) \approx u_{new} = C_0(s) \times u(x_{i-1}) + C_1(s) \times u(x_i) + C_2(s) \times u(x_{i+1}) + C_3(s) \times u(x_{i+2}) \quad (17)$$

Although (17) represents a 1-D interpolation, it can be applied to 2-D interpolation through the horizontal and vertical directions, respectively. The circuit first interpolates the four intermediate values through the vertical direction and interpolates the target value through the horizontal direction. Denote the luminance values of coordinate (x, y) at the source, vertically interpolated image, horizontally interpolated value as $s_{x,y}, V_{x,y}, H_{x,y}$ respectively.Since the proposed method uses (17) for interpolation, the first intermediate value through vertical direction can be calculated as

$$V_{x-1,y} = C_0(s) \times A_{x-1,y-1} + C_1(s) \times A_{x-1,y} + C_2(s) \times A_{x-1,y+1} + C_3(s) \times A_{x-1,y+2} \quad (18)$$

where s is the current distance in the vertical direction, and $A_{x-1,y-1}$, $A_{x-1,y}, A_{x-1,y+1}, A_{x-1,y+2}$ are the four neighboring values in the vertical direction.In the experiment s was taken as 0.5. Three other intermediate values, $V_{x,y}$, $V_{x+1,y}, I_{x+2}, y$ can be calculated in the same way using the different neighbouring values. The four intermediate values above can then be used to interpolate the horizontal value $H_{x,y}$, through the horizontal direction. According to (17), the value e of the pixel in the horizontal direction is

$$H_{x,y} = C_0(s) \times A_{x-1,y} + C_1(s) \times A_{x,y} + C_2(s) \times A_{x+1,y} + C_3(s)A_{x+2,y} \quad (19)$$

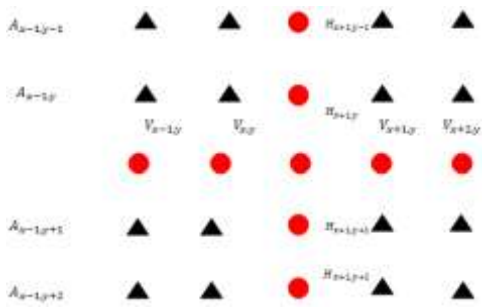

Figure 3: 4x 4 images after vertical and horizontal interpolation

Figure 4 shows a block diagram of the proposed architecture, which includes three main blocks: weight generator, vertical interpolator, horizontal interpolator. To decrease the cost of VLSI realization, the proposed design adopts the cooperation and hardware sharing techniques to implement WG and HI.The WG receives the value s and generates the corresponding four weights, $C_0(s)$ to $C_3(s)$. Equation (16) shows that the values of both $C_0(s)$ and $C_3(s)$ are negative or zero. Utilizing data dependency and adopting the hardware sharing technique makes it possible to minimize multiplication or subtraction operations and shorten the delay path of computation required to calculate the four weights. Hence, the WG generates the four weights in the sequence as

$$C_3(s) = [(1-s)s]s$$

$$C_0(s) = [(1-s)s - C_3]$$

$$C_1(s) = (1-s) + 2C_0 - C_3$$

$$C_2(s) = s + 2C_3 - C_0 \quad (20)$$

The purpose of the VI is to generate the intermediate values using the four weights and values obtained from the WG.The horizontal interpolator calculates the intermediate values in the horizontal direction.

3.EXPERIMENTAL RESULTS

The interpolation technique based on IET is compared with bilinear and nearest neighbour interpolation. The 3 methods are compared using MATLAB software for visual quality. Interpolation technique based on IET solves checkerboard effects .It also reduces the blurring effect The 3 methods are designed using Verilog HDL. The design is synthesized using Xilinx ISE software. Simulation result is found using Modelsim software. The hardware implementation is done in Spartan 3E FPGA kit. Figure 5 shows the simulation results of interpolation based on IET, nearest neighbour and bilinear interpolation .The simulation results show that interpolation based on IET removes checkerboard effect. The blurring effect is also reduced. Table I shows the hardware complexities of 3 methods. It shows that the hardware complexity of interpolation based on IET is higher than that of nearest neighbour but much lesser than that of bilinear interpolation.

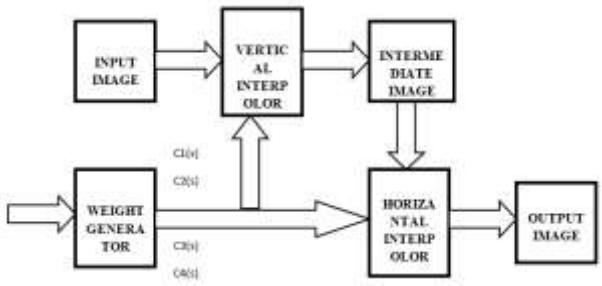

Figure 4: Block diagram of interpolation based on IET

4. CONCLUSION

(a) (b)

(c) (d)

Figure 5: Lena image scaled to 2x a) original b) based on IET c) nearest neighbour d) bilinear

This interpolation method solved the checkerboard effect. The blurring effect is also reduced. Experimental results demonstrate that the hardware complexity of interpolation based on IET is lesser than bilinear interpolation and higher than that of nearest neighbour interpolation. The cooperation and hardware sharing techniques in this design greatly reduced hardware requirements. Table I gives a comparison of hardware complexity of interpolation based on IET with bilinear and nearest neighbour interpolation.

Table 1.Comparison of Hardware Complexities Of Interpolation Techniques

	Based on IET	**Bilinear**	**Nearest**
No. of Slices	1851	3245	1564
4 i/p LUTs	1949	3213	1554
No of Equivalent Gates	30938	45038	26035
Delay(ns)	14.461	18.310	7.505

5. ACKNOWLEDGEMENT

The authors would like to thank Prof. Jose P Varghese and Mrs Smitha Cyriac of Electronics and Communication Department, Viswajyothi College of Engineering and Technology, Vazhakulam, India for their contributions to this work.

6. REFERANCES

[1] Chien-Chuan Huang and Pei-Yin Chen," A Novel Interpolation Chip for Real-Time Multimedia Applications", IEEE trans on circuits and systems for video technology, VOL. 22, no. 10, Oct 2012

[2] R. C. Gonzalez and R. E. Woods,1992, Digital Image Processing. Reading,MA: Addison-Wesley.

[3] T. M. Lehmann, C. Gonner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," IEEE Trans. Med. Imag., vol.18, no. 11, 1049–1075, Nov. 1999.

[4] J. A. Parker, R. V. Kenyon, and D. E. Troxel, "Comparison of interpolation methods for image resampling," IEEE Trans. Med. Imag., vol. MI-2, no. 3, 31–39, Sep. 1983.

[5] H. Hou and H. C. Andrews, "Cubic splines for image interpolation and digital filtering," IEEE Trans. Acoust., Speech Signal Proc., vol. ASSP-26, no. 6, 508–517, Dec. 1978.

[6] R. G. Keys, "Cubic convolution interpolation for digital image processing," IEEE Trans. Acou., Speech, Signal Process., vol. ASSP-29, no. 6, 1153–1160, Dec. 1981.

[7] C. C. Lin, M. H. Sheu, H. K. Chiang, W. K. Tsai, and Z. C. Wu, "Realtime FPGA architecture of extended linear convolution for digital image scaling," in Proc. IEEE Int. Conf. Field-Programmable Technol., Dec. 2008, 381–384.

[8] C. C. Lin, M. H. Sheu, H. K. Chiang, C. L. 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, 480–483.

[9] L. J. Wang, W. S. Hsieh, and T. K. Truong, "A fast computation of 2-D cubic-spline interpolation," IEEE Signal Process. Lett., vol. 11, no. 9, . 768–771, Sep. 2004.

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

[11] 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. IEEE Int. Conf Intell. Inform. Hiding Multimedia Signal Process., Nov. 2007, 511–514.

[12] P.-Y. Chen, C.-Y. Lien, and C.-P. Lu, "VLSI implementation of an edge oriented image scaling processor," IEEE Trans. Very Large Scale Integr. Syst., vol. 17, no. 9, 1275–1284, Sep. 2009.

[13] Y. Cha and S. Kim, "The error-amended sharp edge (EASE) scheme for image zooming," IEEE Trans. Image Process., vol. 16, no. 6, 1496–1505, Jun. 2007.

[14] Q. Wang and R. K. Ward, "A new orientation-adaptive interpolation method," IEEE Trans. Image Process., vol. 16, no. 4, 889–900, Apr. 2007.

[15] J. F. Epperson,2007, An Introduction to Numerical Methods and Analysis. New York: Wiley.