ADAPTIVE ADDER-BASED STEPWISE LINEAR INTERPOLATION

John Moses C
Department of Electronics and Communication Engineering,
Sreyas Institute of Engineering and Technology, Hyderabad, Telangana, 600068, India.

Abstract. Interpolation is the process of determining the values of a function at positions lying between its samples. Quality and complexity vary with different interpolation methods. A high quality adaptive edge-based algorithm is proposed by using linear interpolation. The proposed algorithm consists of a sigmoidal edge detector, clamp filter and linear interpolation. The clamp filter reduces the blurring effect produced by linear interpolation. Clamp filter performs based on the characteristics of edge pixels. The hardware architecture of edge-based stepwise linear interpolation (AABSI) interpolation algorithm is simulated by using MATLAB as well as MATLAB Simulink and Xilinx System generator. The experimental results show that the proposed AABSI outperforms the conventional adder-based stepwise linear interpolation (ABSI) in terms its peak signal to nose ratio (PSNR) and structural similarity index (SSIM).

Keywords: *Up-sampling, FPGA, scaling, clamp filte, edge detection.*

1 Introduction

Image interpolation is a process of proving high-resolution image from the low-resolution image [1]. The interpolated image is needed in various image processing applications like image compression, image zooming and image generation. So, image interpolation is an important process in image processing field [2]. Image zooming is broadly used to match images of independent sensors of various resolution by improving the resolution of the selected image [3]. Image up-sampling, also called single-image interpolation, and is an essential task for various applications in image processing and computer vision such as high definition television (HDTV), digital photographs and object recognition [4]. Interpolation is also useful for reducing the artifacts such as aliasing and blurring of any image geometric transformation [5]. Further, interpolation is used to remove impulse noise of images by considering noisy pixels of an image as missing pixels [6]. Additionally, mage interpolation is used for reproducing sharp textures and sharp edges of images, while reducing blocking and pixel blurring [7]. Thus, image interpolation is widely used in multimedia devices like media players, graphics renders and HDTV.

Image interpolation is a technique which is used for increasing number of pixels of any given image [8]. It is useful when the image is modified from one pixel stage to another pixel stage [9]. Image interpolation is based on image re-sampling. Re-sampling is a task of providing a new set of coordinate points from a discrete image that is considered at one set of coordinate points. The interpolation technique is utilized to find information of missed pixels in a given image depending on the information provided by the given pixels. For instance, in the nearest neighbour image interpolation, a new pixel is provided by utilizing the nearest neighbouring pixel of the given image [10]. The given pixel information is related to coordinates, density or colour [11]. Interpolation is typically carried out by convolving the given image with a weighting function kernel [12].

Image interpolation is typically grouped as non-adaptive interpolation and adaptive interpolation. The adaptive interpolation provides good results than non-adaptive interpolation [13]. The non-adaptive interpolation handles pixels as a sequence of raw intensity costs, and it doesn't utilize run-time conclusions on the system to recognize new pixels. But the adaptive interpolation categorizes pixels as either the areas of edge and non-edge. Based on the area of edges, diverse algorithms are utilized on old pixels to discover new pixels.

In recent years, numerous adaptive image interpolation techniques have been implemented for integrating the features of human visual structure [15]. Directional adaptive image interpolation technique identifies new pixel by approximating edge direction [16]. Wang & Ward [17] present an adaptive interpolation by using isophotes and by calculating average curvature by selecting edge pixels. Anisotropic interpolation [18] uses Gaussian filter to enhance edge pixels. Another edge-guided interpolation [19] uses directional filtering and data fusion. Bilateral filter [20] is also used to provide high-resolution pixels and this method operates at high speed. Mori et al. [21] presented a simple edge adaptive image interpolation. A modified edge-directed interpolation [22] uses bilinear algorithm to interpolate pixels at the smooth regions and utilizes co-variance-based interpolation algorithm for the pixels at the edge region. Modified Leung Malik filter bank [23] is also used to enhance edge information. A higher-order edge-directed interpolation [24] is used for data compression scheme. An adaptive edge-directed interpolation for display system [25] uses edge orientation to interpolate pixels. A displacement field-based interpolation [26] reconstructs sharp edges by preserving the distances of pixels on edges. Edge training set [27] is also used to enhance edge pixels. An edge-directed image superresolution [28] estimates a sharp high-resolution gradient field by using an adaptive selfinterpolation scheme. Choi & kim [29] present an SR algorithm with edge-based mapping kernels. Bilinear interpolation [30] uses sobel filter to enhance edge pixels. A fast edge-filtered image up-sampling [31] utilizes piecewise hyperbolic function to reduce spatial oscillation.

The hardware architecture of various edge-based adaptive image interpolation schemes is implemented by using FPGAs in the past decade. An edge-oriented image scaling processor is presented by Chen et al. [32]. Bilinear with sharp and clamp filter is used to provide edge-enhanced interpolation [33] for real-time multimedia applications. Another bilinear with sharpening filter is also utilized to enhance images for the same multimedia applications [34]. Iterative linear interpolator [35] is developed by using iterative linear approximation. Another interpolator [36] is developed by using canny edge detection and Gaussian filtering. These methods are developed by using complex filtering techniques and interpolation schemes.

Linear interpolation requires only two pixels to calculate the interpolated pixel value. Linear interpolation is a first order polynomial with less complexity. Linear interpolation is frequently used in the reconstruction of signals [37]. In order to reduce the computation complexity and to achieve the high visual quality, linear interpolation is used [38]. To enhance the accuracy further, the image clamp filter is employed with the linear interpolation [39]. This work improves the quality of interpolated pixel by using edge detection and by using clamp filter with adder-based stepwise linear interpolation [37].

The rest of this paper is organized as follows. Section 2 explains ABSI algorithm. Section 3 explains the hardware architecture of proposed algorithm. Section 4 explains the clamp filter kernel. Section 5 gives a comparative analysis based on quantitative measures such as PSNR, SSIM and FPGA characteristics such as look-up tables (LUTs) and memory requirements. Section 6 gives the conclusion and scope of the research.

2 ABSI Algorithm

Linear interpolation is a first order polynomial algorithm. In the adder-based stepwise linear Interpolation (ABSI) method only simple operators that compare, shift and add are used [37]. By using ABSI the interpolated pixel values can be calculated using Equation (1)

$$y_k = x_1 + (x_2 - x_1) >> (d - d_1)$$
$$y_k = x_2 + (x_1 - x_2) >> (d - d_2) \quad (1)$$

The interpolated pixel value is calculated as Y_k.

The address and distance generator generate the row address and column address and distance required for interpolation. The coordinates of an interpolated pixel depends on scale ratio. The origin of the source image in source coordinate is (0.5, 0.5). The first interpolated pixel in scaled coordinates begins at (*1/scale ratio×0.5, 1/scale ratio×0.5*). The row address has to be found in-order to interpolate the row. The row address can be calculated by Equation (2).

$$raddr_0 = \begin{cases} (\text{integer of } y), & \text{fraction of } y >= 0.5 \\ (\text{integer of } y)-1, & \text{fraction of } y < 0.5 \end{cases} \quad (2)$$

$$raddr_k = raddr_0 + k, k=1,2,3$$

Similarly column address can be obtained by using the horizontal co-ordinate x. The column address is the orientation of pixels in the horizontal direction. The distance in the vertical direction and the horizontal direction is calculated using Equation (3)

$$\begin{aligned} vs &= y-(raddr+0.5) \\ hs &= x-(caddr+0.5) \end{aligned} \quad (3)$$

where *vs* and *hs* is the distance in the vertical and horizontal direction respectively[40].

3 Edge Detection and Filtering

A simple sigmoidal edge detecting model [34] is used for the edge detecting mechanism. The local characteristics of neighboring pixels around the target interpolated pixel P(k) can be evaluated by the neighboring pixels P(m+1), P(m). By using the sigmoidal edge detecting model, the asymmetry of the local characteristics neighboring around P(k) can be evaluated by using Equation (4).

$$A = |P(m+1) - P(m-1)| - |P(m+2) - P(m)| \quad (4)$$

Where A is asymmetry parameter and it is used to evaluate the asymmetry of the data in the neighborhood of P(k). Based on asymmetry of the edges, multiplexer allows either direct image pixels or filtered pixels as shown in Fig. 1. A 3x3 clamp filter [33] is used to perform this filter operation.

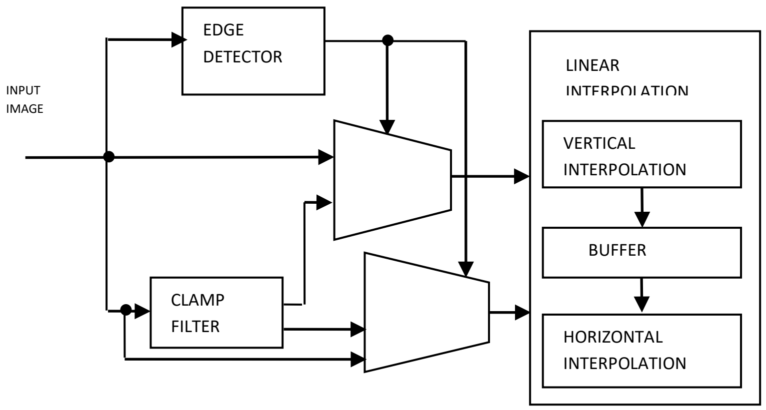

Fig. 1. Block diagram of edge-based stepwise linear interpolation (AABSI)

4 Experimental Results

Quantitative measure is the figure of merit used for the evaluation of image processing technique. Both PSNR and SSIM [34] are measured for different

interpolation algorithms using MATLAB. In order to get these measures, the input image is first down sampled by the factor of 0.5 and up sampled by the factor of 2. For testing the quality of the image scaling algorithms, the Kodak images with the size of 768x512 pixels are collected from the Kodak database. The experimental results show that the proposed AABSI with clamp filter and edge detection technique achieves better quantitative measure than ABSI method [37] and filter-based ABSI [39]. Table 1 shows the quantitative analysis measurements such as PSNR and SSIM of various Kodak images for varying scaling ratios by using different interpolation methods such as AABSI, ABSI with clamp filter [39] and ABSI [37] algorithm.

Table 1. PSNR(dB) and SSIM comparison

Images	**ABSI [37]**		**ABSI with clamp filter [39]**		**AABSI [Proposed]**	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Kodim01	17.22	0.5742	17.45	0.5880	17.80	0.6087
Kodim02	23.58	0.7409	23.85	0.7534	24.10	0.7640
Kodim03	23.83	0.9283	24.12	0.9324	24.38	0.9360
Kodim04	21.63	0.8638	21.87	0.8703	22.08	0.8755

Kodim05	16.20	0.6450	16.51	0.6660	16.77	0.6786
Kodim06	18.72	0.8654	19.09	0.8760	19.44	0.8846
Kodim07	18.77	0.7131	19.06	0.7315	19.32	0.7438
Kodim08	13.86	0.6253	14.07	0.6392	14.34	0.6543
Kodim09	19.72	0.7667	20.00	0.7790	20.48	0.7962
Kodim10	21.26	0.8289	21.46	0.8367	21.73	0.8453
Kodim11	19.66	0.8005	19.95	0.8111	20.22	0.8205
Kodim12	21.21	0.8795	21.43	0.8848	21.67	0.8904
Kodim13	15.96	0.6928	16.26	0.7091	16.59	0.7244
Kodim14	18.50	0.7984	18.80	0.8111	19.08	0.8209
Kodim15	20.09	0.9513	20.28	0.9532	20.43	0.9547
Kodim16	22.61	0.9006	23.02	0.9090	23.35	0.9151
Kodim17	21.04	0.8891	21.26	0.8947	21.49	0.8995
Kodim18	18.73	0.6757	19.01	0.6939	19.29	0.7078
Kodim19	17.90	0.7623	18.25	0.7781	18.54	0.7901
Kodim20	19.69	0.9546	19.84	0.9561	20.09	0.9585
Kodim21	18.30	0.7469	18.64	0.7624	18.93	0.7740
Kodim22	21.04	0.8650	21.26	0.8710	21.51	0.8774
Kodim23	22.23	0.9322	22.53	0.9368	22.75	0.9397
Kodim24	17.82	0.8159	18.15	0.8279	18.47	0.8377
Average	19.43	0.7955	19.84	0.8113	20.12	0.8207

As shown in Table I, the average PSNR value of the AABSI algorithm is higher than the clamp filter-based ABSI [39] and ABSI [37] and the average SSIM value of AABSI) algorithm is higher than clamp filter-based ABSI [39] and ABSI [37] algorithm.

The various device utilization factors are estimated by synthesizing the existing and proposed algorithms using Xilinx ISE 14.5. From the synthesis report the number of look-up tables (LUTs), slice registers, input/output buffers (IOBs), time and memory requirements used in both ABSI with filter and AABSI are obtained. Table 2 shows the various very large scale integrated circuit (VLSI) design characteristics of the filter-based ABSI [39] and the proposed EBSI algorithms.

Table 2. Comparison of FPGA parameters

Parameters	**AABSI [proposed]**	**Clamp filter based ABSI [39]**
No .of slice register	191	191
No .of LUTs	269	242
No .of Occupied slices	118	102
No. of LUT flipflop pair	393	345
IOBs	147	121
Power(W)	0.285	0.271
Combinational path delay(ns)	1.152	0.798
CPU time(s)	12.46	14.07
Elapsed time(s)	12.00	14.00

From Table 2, it is observed that edge-based ABSI (AABSI) requires 269 LUTs and 393 LUT flipflop pairs and it requires 118 slices. Thus, AABSI requires a few more number of LUTs than fitter-based ABSI [39], however it provides higher quality.

4 Conclusion

This work proposes an adaptive adder-based stepwise linear interpolation technique to develop a high quality interpolated image. The proposed method achieves 6.44% improvement in PSNR and 1.15% improvements in SSIM by comparing with clamp filter-based ABSI. The quality of the proposed stepwise linear interpolation is also higher than the conventional ABSI. Further, this work is to be improved by using edge orientation to achieve better quality. So the proposed stepwise linear interpolation is very suitable for generating high-resolution image from the low-resolution version of

the same image. Furthermore, this work to be enhanced in terms low-complexity by using low-complexity convolution kernel of the pre-filter.

References

1. Wang H, Gao X, Zhang K and Li J, "Single image super-resolution using active-sampling Gaussian process regression", IEEE Transaction on Image Processing, vol. 25, no. 2, pp. 935-948, 2016.
2. Kim H, Cha Y and Kim S, "Curvature interpolation method for image zooming", IEEE Transactions on Image Processing, vol. 20, no. 7, pp. 1895-1903, 2011.
3. Arcelli C, Brancati N, Frucci, M, Ramella G and Baja GSD, "A fully automatic one-scan adaptive zooming algorithm for color images", Journal of Signal Processing, Elsevier, vol. 91, no. 1, pp. 61-71, 2011.
4. Zhao Y, Wang R, Wang W and Gao W, "High resolution localstructure-constrained image up-sampling", IEEE Transaction on Image processing , vol. 24, no. 11, pp. 4394-4407, 2015.
5. Delibasis K.K and Kechriniotis A, "A new formula for bivariatehermite interpolation on variable step grids and its application to image interpolation", IEEE Transactions on Image Processing, vol. 23, no. 7, pp. 2892-2904, 2014.
6. Kalyoncu C, Toygar O and Demirel H, "Interpolation based impulse noise removal", IET Image Processing, vol. 7, no. 8, pp. 777-785, 2013.
7. Liu Y.N, Lin Y.C, Huang Y.L and Chien S.Y, "Algorithm and architecture design of high-quality video up-scaling using database free texture synthesis", IEEE Transaction on Circuits and Systems for Video Technology, vol. 24, no. 7, pp. 1221-1234, 2011.
8. Oliver R and Hanqiang C, "Nearest neighbor value interpolation", Proceedings of international Journal of Advanced Computer Science and Applications', vol. 3, no. 4, 2012, pp. 1-6.
9. Liu X, Zhao D, Zhou, J Gao, W and Sun, H, 'Image interpolation via graph-based Bayesian label propagation", IEEE Transactions on Image Processing, vol. 23, no. 3, pp. 1084-1096, 2014.
10. Sharma S and Walia R, "Zooming digital images using modal interpolation", International Journal of Application or Innovation in Engineering and Management, vol. 2, no. 5, pp. 305-310, 2013.
11. Roy R, Pal M and Gulati, T, "Zooming digital images using interpolation techniques", International Journal of Application or Innovation in Engineering and Management, vol. 2, no. 4, pp. 34-45, 2013.
12. Shi J and Reichenbach S.E., "Image interpolation by two-dimensional parametric cubic convolution", IEEE Transactions on Image Processing, vol. 15, no. 7, pp. 1857-1870, 2006.
13. Liu X, Zhao R.X, Ma S, Gao W & Sun H, "Image interpolation via regularized local linear regression", IEEE Transactions on Image Processing, vol. 20, no. 12, pp. 3455-349, 2014.
14. Szydzik T, Callico G.M and Nunez, A, "Efficient FPGA implementation of a high-quality super-resolution algorithm with real-time Performance", IEEE Transactions on Consumer Electronics, vol. 57, no. 2, pp. 664-673, 2011.
15. Arandiga F, Donat R and Mulet P, 'Adaptive interpolation of images', Journal of Signal Processing, Elsevier, vol. 83, no. 2, pp. 459- 464, 2003.