Modified Cubic Convolution Interpolation for Low Computational Complexity

Young-Hyun Jun[1], Jong-Ho Yun, Myung-Ryul Choi
Dept. of EECI, Hanyang Univ., 1271, Sa-1 dong, Ansan, Kyeonggi-do, 425-791, Korea
Tel: +82-31-400-4036; E-mail: impword@asic.hanyang.ac.kr

Abstract

In this paper, we propose a modified cubic convolution interpolation for the enlargement or reduction of digital images using a pixel difference value. The proposed method has a low complexity: the number of multiplier of weighted value to calculate one pixel of a scaled image has seven less than that of cubic convolution interpolation has sixteen. We use the linear function of the cubic convolution and the difference pixel value for selecting interpolation methods. The proposed method is compared with the conventional one for the computational complexity and the image quality. The simulation results show that the proposed method has less computational complexity than one of the cubic convolution interpolation.

1. Introduction

The resolution of image sources and digital display device are various. When the resolution of a stream image generated by a PC is different from the screen resolution of a digital display device, an interpolation process is necessary. Interpolation has a wide range of application in image processing systems to allow users to vary the size of image interactively. It is required for resolution conversion to adapt to the characteristics of a particular displays device.

Interpolation generates a new pixel by analyzing the surrounding pixels and commonly is implemented by convolving an image with a small kernel for the weighting function. Popular methods of interpolation by convolution include nearest neighbor interpolation [1], bilinear interpolation[2], and cubic convolution interpolation[3]. The nearest neighbor interpolation is to assign the pixel closest to the newly generated address as the output pixel. It is a simple interpolation but visual blockiness, also known as the jaggies, can be seen in the output. The most widely used method is the bilinear interpolation which is regarded as the linear function. The newly generated pixel is a weighted sum of the four nearest pixels. Cubic convolution interpolation sharpens an image but requires more computational complexity.

This paper proposes a modified cubic convolution interpolation for low computational complexity. We use the linear function of the cubic convolution and the difference pixel value for selecting interpolation methods. In section 2, we review conventional interpolation. In section 3 and 4, the proposed method is described in detail. The conclusion is made in section 5.

2. Conventional Interpolation

2.1 Bilinear Interpolation

The bilinear interpolation determines the newly generated pixel from the weighted average of the four closest pixels to the specified input coordinates. Bilinear method uses three linear interpolations. The first linear interpolation evaluates the first interpolated value from the values at NW and NE. In the same way, linear interpolation at the second interpolated value evaluates from the values at SW and SE. The new pixel value is then linearly interpolated from the two values previously obtained.

Bilinear interpolation yields a smoother image than nearest neighbor interpolation. Because of the three linear interpolations per pixel, bilinear interpolation

[1] This work is supported by IT R&D Project funded by Korea Ministry of Information and Communications.

requires significantly more computations than nearest neighbor interpolation does.

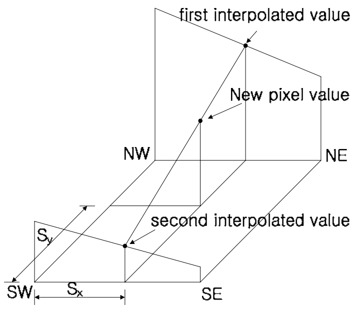


Figure 1 Physical meaning of the bilinear interpolation.

2.2 Cubic Convolution Interpolation

Nearest neighbor interpolation uses one pixel to generate a new pixel. Bilinear interpolation requires the four nearest pixels of input. Cubic convolution interpolation requires sixteen-nearest pixels to generate the output pixel.

Cubic convolution function has negative values, which brings up two important points. The first is that this function will sharpen more than nearest neighbor interpolation or bilinear interpolation. The second is that it is possible to output negative number. The output will need to be clipped not to output negative pixel values.

The 1-dimensional cubic convolution function is defined as equation (1).

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

Rifman[5] and Bernetein[6] have set a = -1, which casuse *f(x)* to have the same slope, -1, at x = 1 as the *sinc(x)* function.

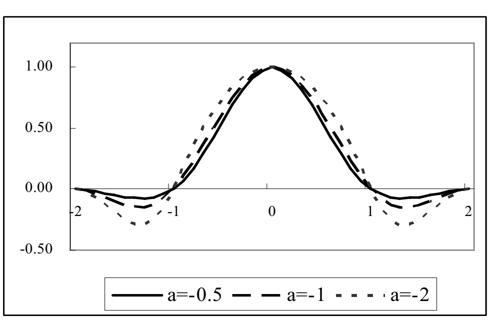


Figure 2 Cubic convolution interpolation function.

Because of three order function, this interpolation requires more computation than nearest neighbor or bilinear interpolation does.

3. Proposed Interpolation Algorithm

In this paper, we reduce the number of a pixel to generate a new pixel using image analysis and the linear equation of cubic convolution function.

Cubic convolution interpolation has three order function for the coefficient of pixels. We use the linear function to calculate the coefficient of a pixel. Figure 3 shows linear cubic convolution function.

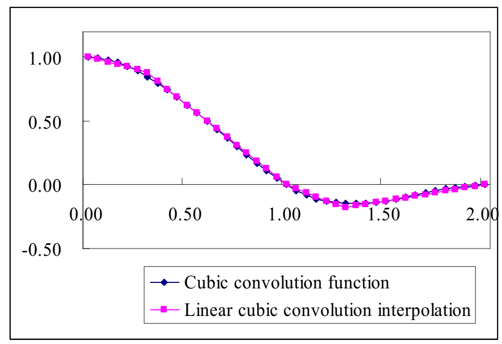


Figure 3 Linear cubic convolution function.

$$f(x)=\begin{cases}-0.375|x|+1 & 0\le|x|<0.25\\ -1.25|x|+1.25 & 0.25\le|x|<1\\ -0.625|x|+0.625 & 1\le|x|<1.25\\ 0.25|x|-0.5 & 1.25\le|x|<2\end{cases} \quad (2)$$

We use equation (2) to reduce the computation of coefficient, where one multiplier and one adder are employed.

Neighbor pixel values of the destination point are close. Figure 4 shows a difference of the neighbor pixel values.

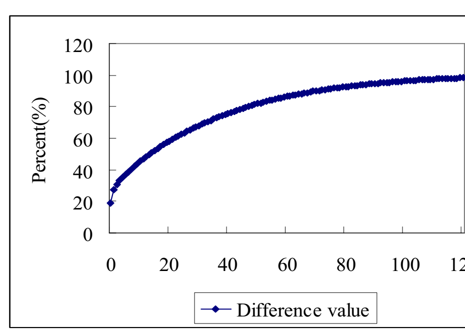


Figure 4 A difference value of the neighbor pixel.

$$Diff(x_k) = abs[a(x_k) - a(x_{k+1})] + abs[a(x_k) - a(x_{k-1})]/2 + abs[a(x_{k+2}) - a(x_{k+1})]/2 \quad (3)$$

We apply two interpolation methods to decrease an unnecessary computation. One interpolation method is selected by the threshold value using a neighbor pixel difference value. Figure 5 is RMSE (Root Mean Square Error) of up-after-down scale and down-after-up scale. We determine a threshold value as 30.

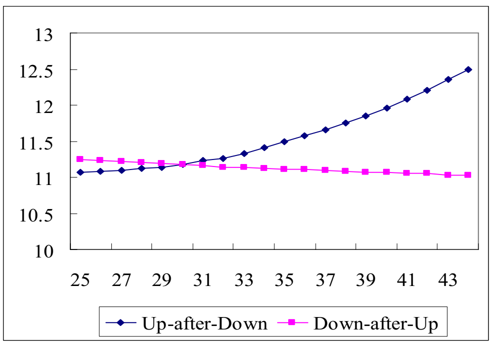


Figure 5 RMSE of up-after-down scale and down-after-up scale.

We determine interpolation algorithm by the threshold value.

$$f(x_k) = \begin{cases} bilinear(x_k) & diff(x_k) < threshold \\ linear\,cubic(x_k) & otherwise. \end{cases} \quad (4)$$

Figure 6 show the processing procedure of proposed method.

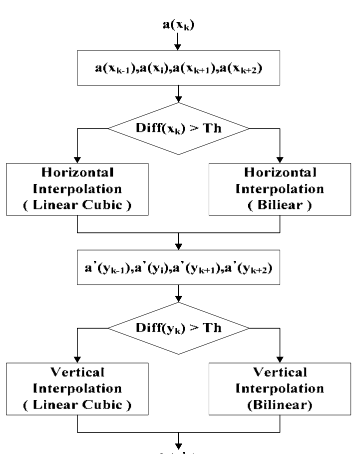


Figure 6 The processing procedure of proposed method.

As shown in Figure 7, a zoneplate image is used to compare a frequency response of proposed method with conventional interpolation methods.

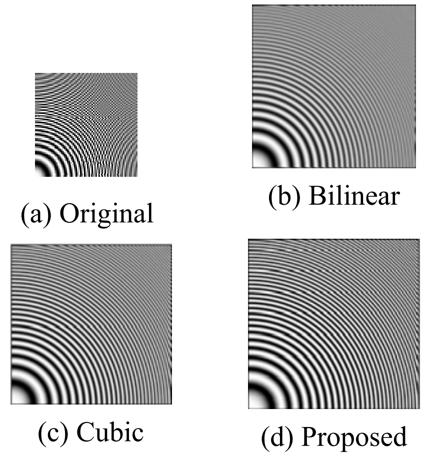


Figure 7 Zoneplate image up(1.6 times) scale.

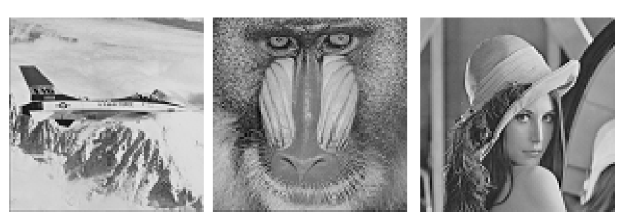
Table 1 shows the number of operations of comparative algorithms for the operation per one pixel. The number of operations is the number of multiplier and adder per one pixel.

Table 1 Comparison of operations per one pixel.

Factor / Algorithm	Weighing factor	Neighbor pixel operations	Memory operations
Bilinear	Mul : 3 Add : 2	Mul : 4 Add : 3	Read : 4 Write : 1
Cubic	Mul : 16 Add : 36	Mul : 16 Add : 15	Read : 16 Write : 1
Proposed	Mul : 7 Add : 13	Mul : 8 Add : 7	Read : 8 Write : 1

4. Simulation Results

We compared proposed interpolation to conventional interpolations in terms of image quality using RMSE. We used six images shown in Figure.8



(a) (b) (c)

(d) (e) (f)

(a) F16 (b) Baboon (c) Lena

(d) Bike (e) Background (f) Text

Figure 8 Six images for RMSE comparisons

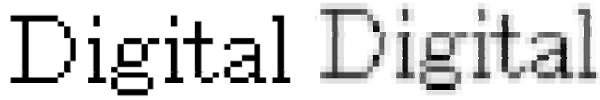
The image quality of comparative interpolations is shown in Table 2 and Table 3.

Table 2 Comparison of Up(1.6 times)-after-Down(1/1.6 times) RMSE .

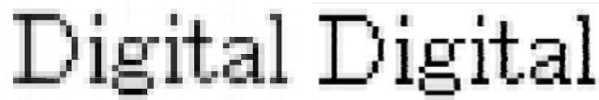
Algorithm / Test image	Bilinear	Cubic	Proposed
F16	2.16	1.16	2.05
Baboon	6.73	3.57	4.85
Lena	2.10	1.14	1.95
Bike	7.18	3.60	3.57
Background	9.71	5.81	5.61
Text	12.99	9.84	6.45

Table 3 Comparison of Down(1/1.6 times)-after-Up(1.6 times) RMSE .

Algorithm / Test image	Bilinear	Cubic	Proposed
F16	5.05	4.03	4.12
Baboon	14.87	13.92	14.41
Lena	4.68	3.98	4.06
Bike	16.02	14.30	12.81
Background	22.54	20.70	20.14
Text	29.74	27.09	26.11



(a) Original (b) bilinear



(c) Cubic convolution (d) Proposed

Figure 9 Up(1.6 times)-after-Down(1/1.6 times) test image which has characters "Digital".

5. Conclusion

In this paper, we propose a modified cubic convolution interpolation which has less computation complexity than conventional interpolation. In order to decrease the computation complexity, we use the linear function of the cubic convolution function and the difference pixel value for selecting interpolation methods. Simulation results show that the proposed method provides less computation complexity than one of the conventional interpolation. The proposed method will be applied to display systems which have the different resolution of image sources and digital display devices.

6. References

[1] R. Crane, A Simplified Approach to Image Processing, Prentice-Hall, 1997.

[2] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Prentice-Hall, 1992.

[3] R. G. Keys, "Cubic Convolution Interpolation for Digital Image Processing," IEEE Trans. ASSP, Vol. ASSP-29, pp. 1153-1160, 1981.

[4] Y. Bai and H. Zhuang, "On the Comparison of Bilinear, Cubic Spline, and Fuzzy Interpolation Techniques for Robotic Position Measurements," IEEE Trans. Inst. Meas., Vol. 54, No. 6, pp. 2281-2288, 2005.

[5] S. S. Rifman, "Digital Rectiflcation of ERTS Multispectral Imagery," Pmc. Symposium Signifiant Results Obtained from ERTS-1 (NASA SP-327), I, Sec. B, pp. 1131-1142, 1973.

[6] R. Bernstein, "Digital Image Processing of Earth observation sensor data," IBA4 J. Res. Dewel., MI. 20, pp. 40-57, 1976.

[7] C. H. Kim, S. M. Seong, J. A. Lee, L. S. Kim, "Winscale : An Image-Scaling Algorithm Using an Area Pixel Model," IEEE Trans. Circuit and System, Vol. 13, No. 6, pp. 549-553, 2003.