An Efficient Edge Preserving Image Interpolation Algorithm

Abhinash Kumar Jha*, Ayush Kumar*, Gerald Schaefer† and Md. Atiqur Rahman Ahad‡
* The LNM Institute of Information Technology, Jaipur, India
† Department of Computer Science, Loughborough University, U.K.
‡ University of Dhaka, Dhaka, Bangladesh

Abstract—**Quality degradation and computational complexity are the major challenges for image interpolation algorithms. Advanced interpolation techniques achieve to preserve fine image details but typically suffer from lower computational efficiency, while simpler interpolation techniques lead to lower quality images. In this paper, we propose an edge preserving technique based on inverse gradient weights as well as pixel locations for interpolation. Experimental results confirm that the proposed algorithm exhibits better image quality compared to conventional algorithms. At the same time, our approach is shown to be faster than several advanced edge preserving interpolation algorithms.**

I. Introduction

Interpolation techniques comprise approaches designed to construct new data points from known data samples. In the case of image interpolation, data samples clearly correspond to image pixels, while interpolation is primarily employed for resampling, i.e. increasing or decreasing the image resolution. In the case of upsampling, the aim is thus to increase the per pixel density of an image and can be simply understood as the process of enlarging an image of low resolution to a high resolution version of the image.

A good interpolation technique should not only preserve fine image details such as edges, but should also be computationally efficient. The following criteria should be kept in the mind when judging an interpolation technique [1]:

- Geometric invariance: preserving the geometry and relative sizes of objects in the image;
- Contrast invariance: maintaining the luminance of objects and the overall contrast of the image;
- Noise: no noise or artefacts should be introduced;
- Edge preservation: preserving edges and boundaries, sharpening them where possible;
- Aliasing: no zagged or staircase edges should be introduced;
- Texture preservation: textured regions should not be blurred;
- Oversmoothing: no undesirable piecewise constant or blocky regions should be introduced;
- Application awareness: the produced results should be appropriate for the type of image and order of resolution (for example, they should appear realistic for photographic images, but typically require crisp edges and high contrast for medical images; if the interpolation is for general images, the method should be independent of the type of image);
- Sensitivity to parameters: should not be too sensitive to internal parameters that may vary from image to image;
- Computational efficiency: should not be computationally inefficient.

A variety of interpolation techniques have been introduced in the literature. Nearest neighbour interpolation [2], billinear interpolation [3], bibcubic interpolation [4] and spline interpolation [5] are some of the most well known and widely used algorithms for this purpose. Nearest neighbour interpolation involves the translation of known pixel values, while in billinear interpolation the average value of the nearest pixels on either side is used. Bicubic interpolation involves the translation of a weighted average of the nearest pixels, whereas spline interpolation involves a special type of piecewise polynomial as the interpolant. Unfortunately, while being of low computational complexity, these approaches suffer from insufficient preservation of spatial details in the image and consequently lower image quality.

In contrast, interpolation algorithms that aim to preserve edge structures are computationally more complex. Li and Orchard [6] proposed the edge directed interpolation algorithm (NEDI), in which missing pixels are interpolated based on the estimated covariance of the high resolution (HR) image from the covariance of the low resolution (LR) image. Ketan and Oscar [7] employed an autoregressive method using Gauss-Seidel optimisation that relies on both LR and HR pixels. Jaiswal and Jakhetiya [8] presented an algorithm based on downsampling and using least squares estimation. Jaiswal and Kumar's algorithm [9] is less complex and based on a set of predictors; however these predictors do not adapt well to all types of images. Oscar and Chan [10] suggested a content adaptive interpolation scheme which is computationally simple but fails to give sufficiently high image quality.

In this paper, we propose an image interpolation algorithm that is not only capable of preserving edges and leading to good image quality but is also computationally efficient.

II. Proposed algorithm

The pixels of the image I are classified into four categories based on their spatial locations: odd-odd, odd-even, even-odd and even-even as laid out in Fig. 1.

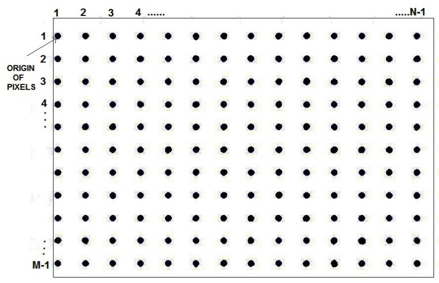

Fig. 1. Sign convention used in the paper.

The initial stage of the algorithm consists of filling the odd-odd locations of the HR image I which are directly translated from the LR image S as

$$I_{2i-1,2j-1} = M(S_{i,j}), \tag{1}$$

which is illustrated in Fig. 2. The remaining pixels are filled in the following two stages.

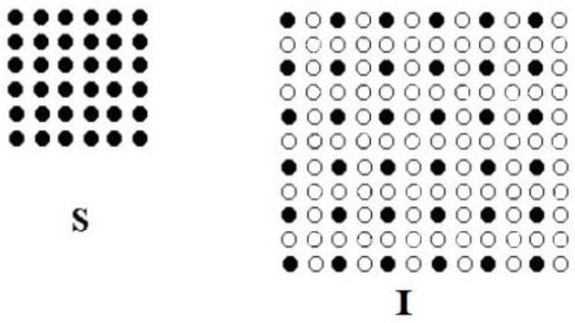

Fig. 2. Translation of odd-odd pixels from LR to HR image.

A. Prediction using inverse gradient weights

In this stage, prediction of pixels is performed so that the fine detailed information of the nearest 4 neighbours of the current spatial location L_C is included. Suppose (d_1^S, d_2^S) and (d_1^D, d_2^D) are the sum and difference respectively of the current diagonally opposite neighbouring pixels of $L_C(i, j)$ as shown in Fig 3.

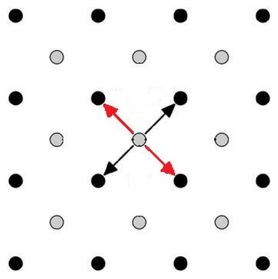

Fig. 3. Diagonal pixels taken in pair marked with the same colour.

Then, the filling of even-even locations is goverened by

$$I_{2i,2j} = \frac{1}{2}[Wd_1^S + (1-W)d_2^S], \tag{2}$$

where the weight W is the inverse gradient weight and is computed as

$$W = \begin{cases} 0.5 & \text{if } d_1^D = d_2^D = 0 \\ [1 + \frac{d_1^D}{d_2^D}]^{-1} & \text{if } d_1^D > d_2^D \\ [1 + \frac{d_2^D}{d_1^D}]^{-1} & \text{if } d_1^D \leq d_2^D \end{cases}. \tag{3}$$

As can be seen, the distribution of weights is defined so that equal weights (0.5) are assigned to spatial locations where no edges can be found and the majority of weights falls in the region which dominates the edges the most, resulting in preservation of edges.

B. Prediction using L_C

Prediction of the remaning pixels (odd-even and even-odd) is performed based on a weighted sum of neighbouring pixels as

$$\begin{aligned} I_{(i,j)} &= W_N(I_{i,j-1} + I_{i,j+1}) + \\ & W_F((I_{i-2,j-1} + I_{i-2,j+1} + I_{i+2,j-1} + I_{i+2,j+1}), \end{aligned} \tag{4}$$

where weights W_N and W_F are based on L_C and calculated as

$$W_N = \frac{1}{2}\frac{1}{1+\frac{2}{P}} \tag{5}$$

and

$$W_F = \frac{1}{2}\frac{\frac{1}{P}}{1+\frac{2}{P}}. \tag{6}$$

Here, P is a position factor, which is found to be different (and hence can be optimised) for different kinds of images. Fig. 4 shows the two weights as a function of P.

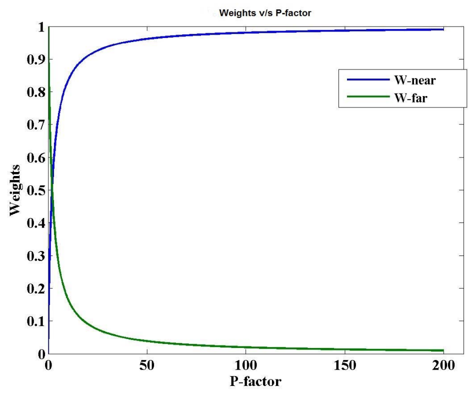

Fig. 4. Plot of W_N and W_F vs. P.

Through extensive experiments, it can be concluded that the majority of edges – and hence image quality – depends predominantly on these weights.

III. Experimental results

To measure both efficacy and efficiency of our algorithm, we carried out an extensive set of experiments. All calculations were carried out on system with an Intel Core^{TM} i3-3130M Processor (3M Cache) 2.6GHz processor. A grayscale image

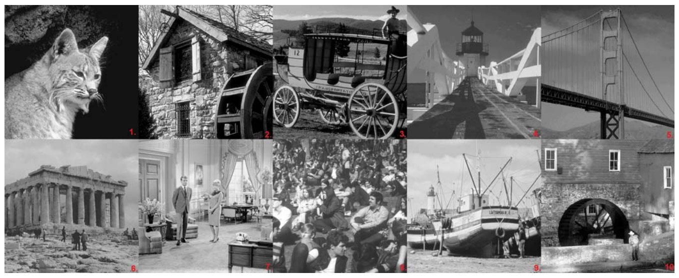
Fig. 5. The 10 test images used in the experiments.

TABLE I
PSNR COMPARISON OF DIFFERENT INTERPOLATION TECHNIQUES ON THE IMAGES OF FIG. 5.

image	bilinear	bicubic	CAI	CBID	CBII	proposed
1	29.34	29.22	30.87	30.92	30.89	31.33
2	22.57	22.49	24.11	24.49	24.37	24.92
3	25.91	25.81	27.20	27.96	27.85	28.41
4	32.00	31.98	34.82	34.50	34.57	34.96
5	32.49	32.43	33.82	33.90	33.89	34.35
6	29.45	29.40	31.14	31.06	30.97	31.51
7	26.94	26.75	28.67	28.00	28.78	29.20
8	28.48	28.49	31.62	31.54	31.32	32.15
9	27.92	27.73	30.29	30.23	30.17	30.67
10	26.53	26.41	28.13	28.12	28.01	28.20
average	28.16	28.07	30.07	30.07	30.08	**30.57**

database [11] comprising a total of 10 images, all of resolution 512×512, were used for the experiments as shown in Fig. 5.

In order to put our obtained results into context, we have implemented five other interpolation techniques: bilinear interpolation [3], bicubic interpolation [4], content adaptive interpolation (CAI) [10], context-based image dependent (CBID) interpolation [9], and context-based image independent (CBII) interpolation [9].

A visual comparision demonstrating edge preservation quality is given in Fig. 6, which shows an original image together with interpolated versions obtained by all tested methods followed by edge detection using a Canny edge detector. It can be confirmed that our proposed approach is capable of correctly preserving more edges when compared to the other techniques.

A visual comparison of resulting image quality can be found in Fig. 7, which shows a sample image interpolated using the different techniques. Again, it is apparent, that our proposed algorithm yields high image quality in comparison with other methods.

To obtain an objective measure of image quality, we employ the standard peak signal-to-noise ratio (PSNR) defined as

$$\text{PSNR}(O, I) = 10 \log_{10} \frac{255^2}{\text{MSE}(O, I)}, \qquad (7)$$

where $\text{MSE}(O, I)$ is the mean squared error between the original image O and I is the interpolated image I generated from the downsampled version S. At the same time, we measure the running times of the various algorithms.

PSNR results for all images in the dataset are given in Table I, while timing results are listed in Table II.

As can be seen from Table I, our algorithm works well in comparison to the other methods we implemented. In fact, over the 10 images, it is shown to give the highest PSNR. At the same time, while not quite matching the efficiency of bilinear or bicubic interpolation, it vastly outperforms the more complex techniques in terms of computational complexity. Thus, our approach is shown to provide a fast interpolation algorithm that is capable of achieving high image quality due to its edge preserving behaviour.

IV. CONCLUSIONS

In this paper, we have presented a computationally efficient edge preserving interpolation algorithm. Our proposed approach interpolates the image by prediction using inverse gradient weights as well as weights found experimently based on the current location within the image. A relationship between

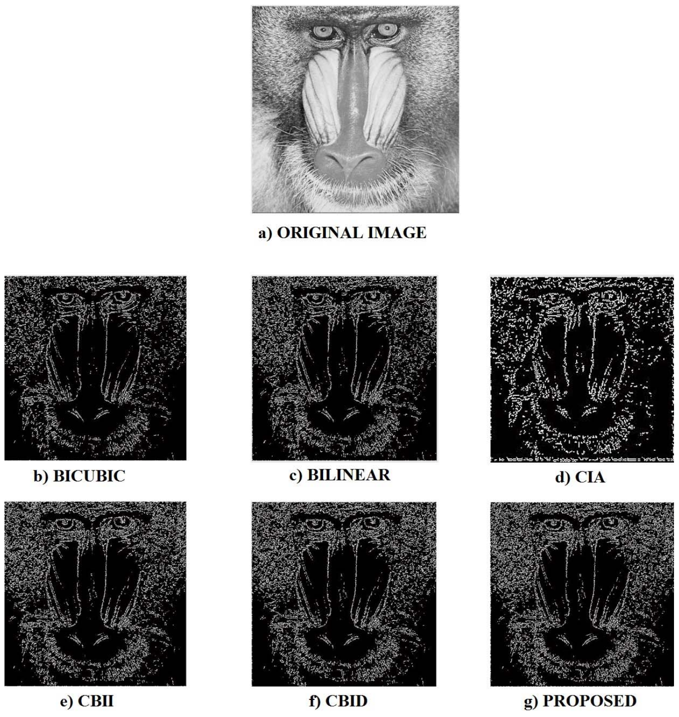

Fig. 6. Example demonstrating egde preservation of different interpolation methods.

TABLE II
RUNTIME COMPARISON [MS] OF DIFFERENT INTERPOLATION TECHNIQUES ON THE IMAGES OF FIG. 5.

image	bilinear	bicubic	CAI	CBID	CBII	proposed
1	80.52	88.75	804.34	531.11	1022.30	190.69
2	76.01	77.96	788.33	640.08	1013.37	209.47
3	78.64	84.27	758.86	527.89	898.25	198.84
4	89.32	80.93	750.39	503.98	892.16	193.80
5	83.87	86.73	766.56	507.80	929.98	202.17
6	86.12	87.96	716.35	509.73	930.21	220.24
7	74.91	87.35	729.08	535.11	920.03	199.92
8	84.13	83.88	751.64	522.25	939.81	177.33
9	82.40	83.03	723.48	496.44	918.74	205.53
10	75.93	86.17	748.23	513.42	952.16	216.43
average	81.19	84.70	753.73	528.78	941.70	201.44

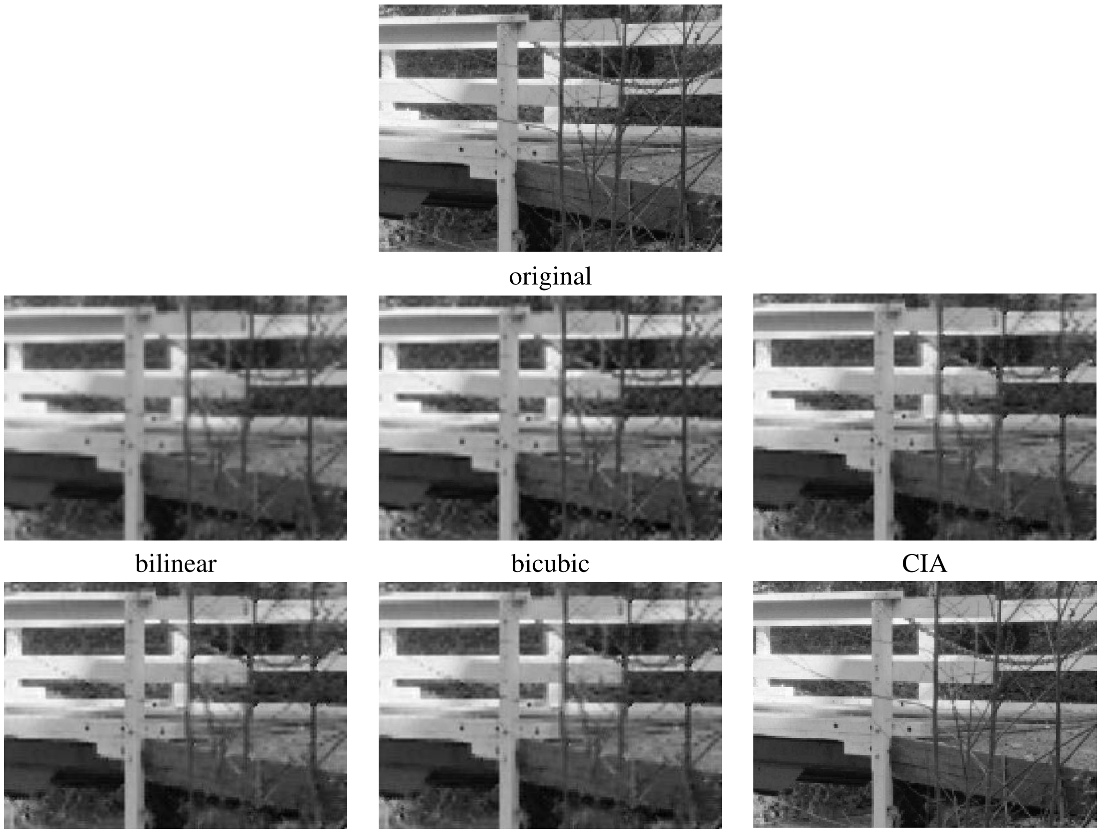

Fig. 7. Comparison of sample image interpolated by different interpolation methods.

the predicted pixel and its surrounding is confirmed, and our algorithm if found to yield not only improved image quality since preserving fine edges but also to be computationally more efficient compared to other, more complex, methods.

REFERENCES

[1] T. Wittman, "Mathematical techniques for image interpolation," *Report Submitted for Completion of Mathematics Department Oral Exam, Department of Mathematics, University of Minnesota, USA*, 2005.

[2] R. Franke, "Scattered data interpolation: Tests of some methods," *Mathematics of computation*, vol. 38, no. 157, pp. 181–200, 1982.

[3] J. Allebach and P. W. Wong, "Edge-directed interpolation," in *Image Processing, 1996. Proceedings., International Conference on*, vol. 3. IEEE, 1996, pp. 707–710.

[4] F. N. Fritsch and R. E. Carlson, "Monotone piecewise cubic interpolation," *SIAM Journal on Numerical Analysis*, vol. 17, no. 2, pp. 238–246, 1980.

[5] H. S. Hou and H. Andrews, "Cubic splines for image interpolation and digital filtering," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 26, no. 6, pp. 508–517, 1978.

[6] X. Li and M. T. Orchard, "New edge-directed interpolation," *Image Processing, IEEE Transactions on*, vol. 10, no. 10, pp. 1521–1527, 2001.

[7] K. Tang, O. C. Au, L. Fang, Z. Yu, and Y. Guo, "Image interpolation using autoregressive model and gauss-seidel optimization," in *Image and Graphics (ICIG), 2011 Sixth International Conference on*. IEEE, 2011, pp. 66–69.

[8] S. P. Jaiswal, V. Jakhetiya, and A. K. Tiwari, "An efficient image interpolation algorithm based upon the switching and self learned characteristics for natural images," in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 861–864.

[9] S. Prasad Jaiswal, V. Jakhetiya, A. Kumar, and A. K. Tiwari, "A low complex context adaptive image interpolation algorithm for real-time applications," in *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*. IEEE, 2012, pp. 969–972.

[10] T.-W. Chan, O. C. Au, T.-S. Chong, and W.-S. Chau, "A novel content-adaptive interpolation," in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*. IEEE, 2005, pp. 6260–6263.

[11] Decsai.ugr.es., "Dataset of standard 512x512 grayscale test images," http://decsai.ugr.es/cvg/CG/base.htm.