An Adaptive Distance-based Edge Preserving Interpolation Algorithm for Natural Images

Abhinash Kumar Jha*, Ayush Kumar†, Gerald Schaefer‡ and Md. Atiqur Rahman Ahad§
* The LNM Institute of Information Technology, Jaipur, India
† Visual Analytics and Imaging Lab, Stony Brook University, New York, USA
‡ Department of Computer Science, Loughborough University, Loughborough, U.K.
§ Department of Electrical and Electronic Engineering, University of Dhaka, Dhaka, Bangladesh

***Abstract*—In this paper, we propose a new image interpolation method using adaptive weights based on inverse gradients and distances from the pixels used in prediction. Since the weights are based on different spatial locations of the pixels, this allows preservation of important edge information and hence to prevent extensive blurring across edges in the upsampling process. Experimental results on a large test image set show that the proposed algorithm gives better performance compared to conventional algorithms.**

***Keywords*—Image resolution; interpolation; upsampling; edge preservation; adaptive weights.**

I. Introduction

Interpolation is an important image processing technique to change the resolution of a digital image, based essentially on estimating values at unknown image locations using known data. Interpolation is commonly employed for various applications such as de-mosaicing, image rotation, and image enlargement where a high resolution image is generated from a lower resolution counterpart.

When employed for the latter, the main objective of an interpolation technique is to increase the pixel density per unit area of an image in order to obtain an higher resolution interpolated image version from a low resolution one [1]. If, as illustrated in Fig. 1, we have a discrete sequence $f(x_k)$ of length N, and this sequence is filtered and downsampled by a factor of 2, we get another sequence $g(x_n)$ of length $N/2$. The interpolation process aims at estimating a sequence $l(x_k)$ of length N, which is as close as possible to the original sequence $f(x_k)$.

Various interpolation algorithms have been introduced in the literature. Amongst the most well known and most commonly applied ones are nearest neighbour interpolation [2], bilinear interpolation [3], bicubic interpolation [1], and spline interpolation [4]. In the nearest neighbour method, the value of a new pixel is taken as the translated value nearest to it. In bilinear interpolation, the interpolated value is the weighted average of the one-translated values on either side, while bicubic methods use the interpolated values as the weighted average of two-translated values on either side. While these methods are used in many applications due to their low computational complexity, they lack the ability to preserve spatial information appropriately since they fail to work properly near edge structures, resulting in blurred image artifacts.

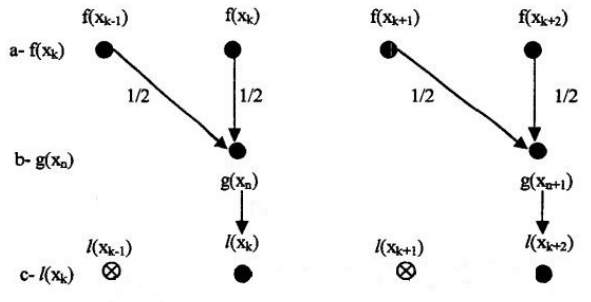

Fig. 1. Signal downsampling and interpolation.

Interpolation algorithms that aim to preserve edge structures in an image are computationally more complex. Li and Orchard [5] proposed an 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. Tang *et al.* [6] employed an autoregressive method using Gauss-Seidel optimisation that relies on both LR and HR pixels. Jaiswal *et al.* [7] presented an algorithm based on downsampling of an image and using least squares estimation. Kumar *et al.*'s approach [8] is less complex and based on a set of predictors, however these do not adapt well to all types of images. Chan *et al.* [9] suggested a content adaptive interpolation scheme which is computationally simple but fails to give sufficiently high image quality, while Jha *et al.* [10] proposed an edge preserving interpolation technique yielding good subjective as well as objective results.

II. Proposed Algorithm

In this paper, we propose a novel edge preserving image interpolation algorithm. Our algorithm comprises three phases. The first phase consists of the filling of known pixels using the source image, while in the remaining two phases the unknown pixels are predicted. In the following we detail each of these phases.

A. Phase 1

The pixels of the interpolated image are categorised into four types, namely odd-odd, even-odd, odd-even and even-even pixels based on their spatial location. In the first phase, the source image $S(LR)$ of size $M \times N$ is used to fill the odd-odd spatial locations of the interpolated image $I(HR)$ based

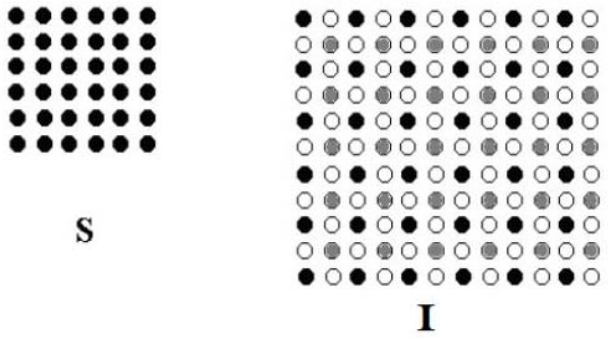

Fig. 2. First stage expansion of an image; black pixels in image I are the original pixels from the low resolution image S.

on the mapping $M : S \rightarrow I$ following

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

whose effect can be seen in Fig. 2.

B. Phase 2

In the second phase, the prediction of pixels at even-even positions in I is performed using the original pixels of the low resolution image leading to precise prediction. The pixels are predicted using a weighted mean given by

$$I_{2i,2j} = \frac{W_1(I_{i-1,j-1} + I_{i+1,j+1}) + W_2(I_{i-1,j+1} + I_{i+1,j-1})}{2(W_1 + W_2)}, \tag{2}$$

where the weights W_1 and W_2 are computed using a modified Manhattan distance as

$$W_1 = \begin{cases} 0.5 & \text{if } d_1 = d_2 = 0 \\ [1 + \frac{d_1}{d_2}]^{-1} & \text{if } d_1 > d_2 \\ [1 + \frac{d_2}{d_1}]^{-1} & \text{if } d_1 \leq d_2 \end{cases} \tag{3}$$

where d_1 and d_2 are the intensity differences between diagonally opposite neighbouring pixels of the current spatial location as shown in Fig. 3, and

$$W_2 = 1 - W_1. \tag{4}$$

W_1 and W_2 set to 0.5 hence denotes equal contribution of the 4-neighbours which can be concluded as a region without edges. The condition $d_1 > d_2$ shows the presence of an edge in perpendicular direction which should be preserved. Calculations of W_1 and W_2 involves decisions on strong or weak edges, based upon which the weights are provided leading to the preservation of weak edges.

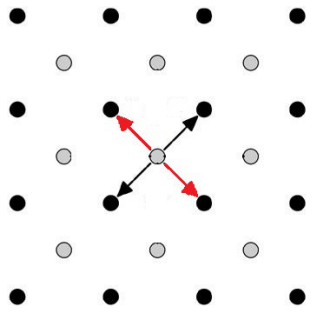

Fig. 3. Neighbouring, diagonally opposite pixels of the current spatial location used for calculating d_1 and d_2.

C. Phase 3

The third phase consists of the calculation of weights W_N and W_F based on the location of the current pixel, and then predicting the remaining pixels by a weighted sum of neighbouring pixels. The weights are calculated by

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

and

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

where P is the position factor, which can be optimised for different (kinds of) images.

The calculation of the position factor proceeds as follows:

- Image I is downsampled by a factor of 2 (i.e., if the initial size of image is $M \times N$, it is downsampled to $\frac{M}{2} \times \frac{N}{2}$.
- Using the recursive Algorithm 1, the P-factor is obtained.

$max_i(Array)$: returns i maximum valued elements from $Array$ in decreasing value sequence
$index(i, Array)$: returns index value of an element valued i from $Array$
$[p_{max}; p_{min}]$ = the interval in which p can lie

$computePfactor(p_{min}, p_{max})$(
for $i = p_{min}$;$i \leq p_{max}$;i=i+$\frac{p_{max}-p_{min}}{10}$ **do**
  compute weights using i
  predict pixel using i
  compute ${}_iPSNR_p$ using i
end
$[PSNR_1, PSNR_2, PSNR_3]$=$max_3({}_iPSNR_p)$
k=$index(\ PSNR_1,{}_iPSNR_p)$
if $PSNR_1 = PSNR_2 = PSNR_3$ **then**
  $p = p_{min} + k \times \frac{p_{max}-p_{min}}{10}$
else
  $p_{max} = p_{min} + (k+1) \times \frac{p_{max}-p_{min}}{10}$
  $p_{min} = p_{min} + (k-1) \times \frac{p_{max}-p_{min}}{10}$
  $p = computePfactor(p_{min}, p_{max})$
end
return p
)

Algorithm 1: Algorithm to calculate the P-factor.

A sample peak signal-to-noise ratio (PSNR) vs. position factor plot for an image is shown in Fig. 4. From there, we can observe that there is a clear maximum in terms of PSNR.

Through extensive analysis, we observed that the P-factor is fairly independent of image resolution. This is illustrated in Fig. 5 which shows how PSNR changes with respect to the P-factor for different image resolutions. Consequently, we can

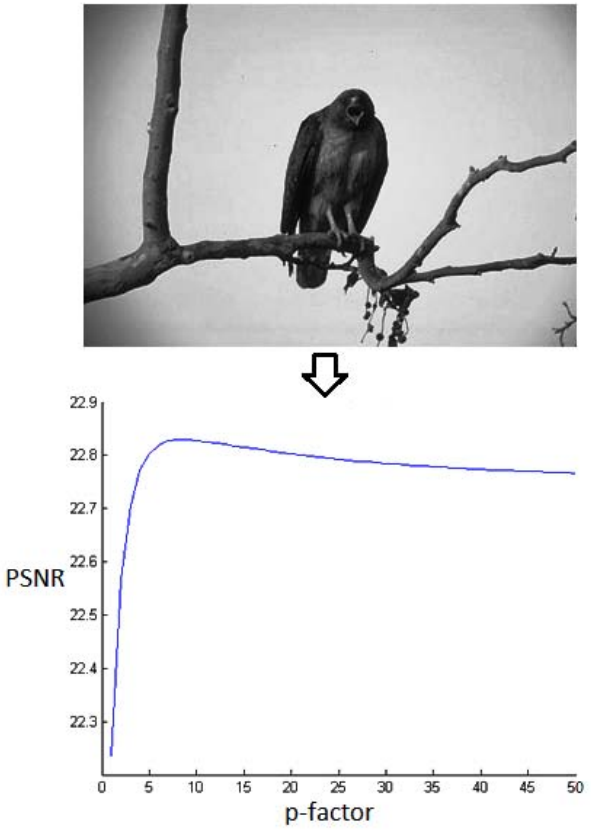

Fig. 4. Position factor vs. PSNR plot (bottom) for a sample image (top).

use the P-factor of the downsampled version for interpolation as used in Algorithm 1.

The selection of neigbouring pixels is based on their spatial locations with a division into near and far pixels. Near pixels are pixels adjacent to the current spatial location, while pixels adjacent to near pixels are categorised as far pixels. For prediction, their weighted sum is employed as

$$I_{(i,j)} = W_N(I_{i,j-1} + I_{i,j+1}) + W_F(I_{i-2,j-1} + I_{i-2,j+1} + I_{i+2,j-1} + I_{i+2,j+1}). \quad (7)$$

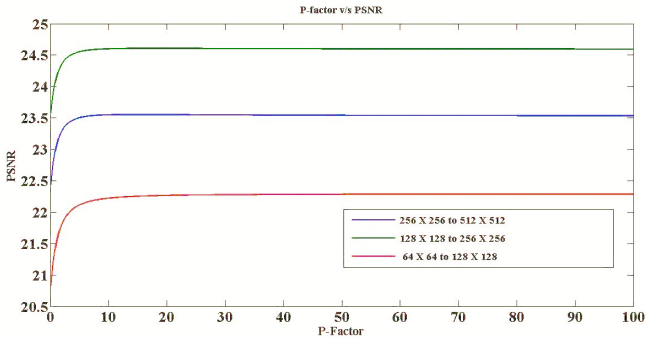

Fig. 5. Variation of PSNR with respect to P-factor for different resolutions of an image.

III. Experimental results

We performed an extensive set of experiments to appropriately test our proposed algorithm. As is common, we evaluate the performance of an interpolation technique using PSNR, defined as

$$\text{PSNR}(O, I) = 10 \log_{10} \frac{255^2}{\text{MSE}(O, I)}, \quad (8)$$

between the original image O and the interpolated image I generated from the downsampled version S of O, where $\text{MSE}(O, I)$ is the mean squared error between O and I.

In order to put our obtained results into context, we have implemented five other interpolation techniques, namely bicubic interpolation (BIC), content adaptive interpolation (CAI) [9], context-based image dependent (CBID) interpolation [11], context-based image independent (CBII) interpolation [11], efficient edge preserving interpolation (EEPI) [10], and interlinear interpolation (ILI) [12].

TABLE I. PSNR comparison of different interpolation techniques on the images of Fig. 6.

image	BIC	CAI	CBID	CBII	EEPI	ILI	proposed
1	23.32	24.37	24.81	24.99	24.79	25.11	**25.39**
2	28.62	30.54	30.50	30.49	30.60	30.68	**30.93**
3	27.38	29.13	29.49	29.42	29.61	29.77	**29.99**
4	29.34	31.08	31.04	30.97	31.18	**31.38**	31.26
5	31.33	33.92	33.79	33.74	33.82	**34.07**	33.99
6	26.02	27.26	27.83	27.67	27.88	**28.13**	28.02
7	19.71	20.74	21.05	21.11	21.08	21.17	**21.26**
8	29.22	30.88	30.93	30.89	31.08	**31.18**	31.12
9	25.87	27.01	27.15	27.13	27.23	**27.31**	27.30
10	27.74	29.99	**32.41**	29.96	29.65	32.39	29.88
11	18.93	20.03	20.42	20.46	20.43	20.51	**20.53**
12	26.25	27.56	28.17	28.11	28.30	**28.33**	28.30
13	26.15	27.94	30.18	28.08	27.62	29.89	**27.99**
14	20.64	21.29	21.95	21.88	21.96	**22.11**	22.09
15	23.58	24.36	25.07	25.04	25.16	**25.35**	25.16
16	21.54	22.28	22.92	22.86	22.82	22.82	**22.99**
17	22.50	24.12	24.5	24.38	24.68	24.91	**24.99**
18	19.52	20.41	21.11	21.06	21.17	21.27	**21.64**
19	25.81	27.20	27.96	27.85	28.16	28.20	**28.99**
20	25.39	26.41	26.91	26.89	27.04	27.23	**27.78**
21	27.60	28.69	28.95	28.92	29.06	29.21	**29.85**
22	23.14	24.24	24.70	24.65	24.62	24.68	**24.81**
23	31.98	34.83	34.50	34.48	34.71	**34.86**	34.83
24	29.45	31.13	31.36	31.33	31.47	**31.57**	32.02
25	32.43	33.82	33.9	33.89	34.11	34.16	**34.33**
26	30.81	32.69	32.84	32.83	32.9	32.96	**33.38**
27	30.82	32.70	32.89	32.9	33.02	33.04	**34.10**
28	28.58	29.74	29.97	29.97	30.02	30.03	**30.31**
29	23.13	24.14	24.69	24.63	24.6	24.57	**24.99**
30	26.03	27.50	27.84	27.64	27.81	28.07	**28.87**
31	19.32	19.79	20.59	20.58	20.52	20.54	**20.62**
32	24.47	25.95	26.61	26.51	26.67	26.73	**27.22**
33	26.42	28.14	28.13	28.04	28.21	28.35	**28.87**
34	26.00	27.64	27.96	27.84	27.96	**28.02**	28.00
35	27.87	29.83	30.05	29.97	30.14	30.13	**30.39**
36	24.45	25.80	26.11	26.05	26.22	26.30	**26.78**
37	27.24	28.65	28.82	28.78	28.9	29.00	**29.25**
38	29.40	31.14	31.07	30.98	31.26	31.47	**31.60**
39	30.82	32.48	32.75	32.69	32.70	32.73	**32.97**
40	33.40	36.04	35.72	35.65	35.86	**36.13**	35.86
41	26.76	28.67	28.8	28.78	28.95	29.02	**29.05**
42	25.60	26.63	27.03	26.85	26.96	**27.16**	26.13
43	28.50	31.63	31.54	31.33	31.91	**32.10**	**32.10**
44	17.09	18.12	18.51	18.43	18.39	18.45	**18.90**
45	26.77	28.72	29.68	29.27	28.91	29.51	**29.52**
46	23.99	25.41	25.82	25.65	25.66	25.83	**25.87**
47	21.35	22.25	23.15	22.62	22.62	**23.14**	22.89
48	27.73	30.29	30.24	30.18	30.42	30.57	**31.01**
49	22.90	17.37	18.23	18.25	18.04	18.33	**18.78**
50	27.59	29.13	29.25	29.14	29.21	29.28	**29.47**
average	26.01	27.39	27.80	27.64	27.72	27.96	**28.05**

Fig. 6. The 50 test images used in the experiments.

Our test database comprises a total of 50 images [11], all with a resolution of 512×512 pixels. Fig. 6 shows the image dataset, while Table I gives the PSNR results for all evaluated algorithms.

As can be seen from Table I, our proposed algorithm provides the highest average PSNR and hence best image quality after interpolation, and also the best image quality for the majority of the 50 test images. This clearly demonstrates that our approach is able to deliver better performance compared to other methods and to yield a useful image interpolation technique.

A visual comparision of the quality of the different methods is presented in Fig. 7 which shows an original image together with interpolated versions obtained by all tested methods, and confirms that our presented approach is capable of correctly preserving edges and of providing superior image quality.

IV. Conclusions

In this paper, we have presented a simple edge preserving image interpolation algorithm. Our proposed algorithm interpolates an image by predicting different weights based on different spatial locations of the pixels in order to preserve edge information in the upsampling process. Through extensive experiments, a relationship between the predicted pixel and its surrounding is confirmed, and consequently our algorithm if found to yield improved image quality compared to other methods. In future work, we will aim at automatically establishing the employed position factor for different types of images.

References

[1] F. N. Fritsch and R. E. Carlson, "Monotonicity preserving bicubic interpolation: A progress report," *Computer Aided Geometric Design*, vol. 2, no. 1, pp. 117–121, 1985.

[2] J. A. Parker, R. V. Kenyon, and D. Troxel, "Comparison of interpolating methods for image resampling," *IEEE Transactions on Medical Imaging*, vol. 2, no. 1, pp. 31–39, 1983.

[3] G. Vendroux and W. G. Knauss, "Submicron deformation field measurements: Part 2. improved digital image correlation," *Experimental Mechanics*, vol. 38, no. 2, pp. 86–92, 1998.

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

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

[6] K. Tang, O. C. Au, L. Fang, Z. Yu, and Y. Guo, "Image interpolation using autoregressive model and gauss-seidel optimization," in *6th International Conference on Image and Graphics*, 2011, pp. 66–69.

[7] 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 *IEEE International Symposium on Circuits and Systems*, 2011, pp. 861–864.

[8] A. Kumar, N. Agarwal, J. Bhadviya, and A. K. Tiwari, "An efficient 2-d Jacobian iteration modeling for image interpolation," in *19th IEEE International Conference on Electronics, Circuits and Systems*, 2012, pp. 977–980.

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

[10] A. K. Jha, A. Kumar, G. Schaefer, and M. A. R. Ahad, "An efficient edge preserving image interpolation algorithm," in *International Conference on Informatics, Electronics & Vision*, 2014.

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

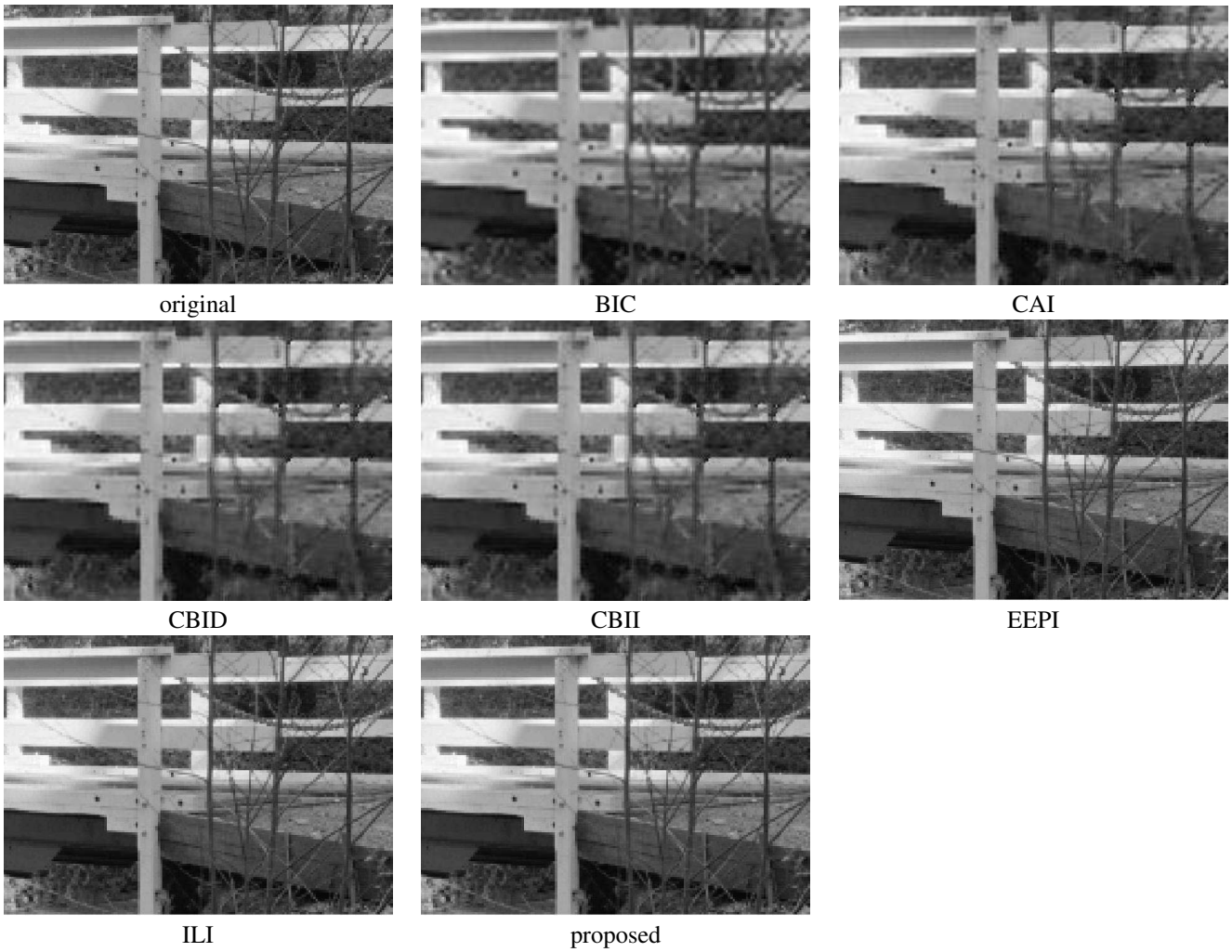

Fig. 7. Comparison of sample image interpolated by the different methods.

[12] A. K. Jha, V. Narwal, Y. Gupta, and A. Kumar, "Interlinear image interpolation scheme for real time application," in *Annual IEEE India Conference INDICON*, 2014.