Proceedings of the 5th WSEAS/IASME Int. Conf. on SYSTEMS THEORY and SCIENTIFIC COMPUTATION, Malta, September 15-17, 2005 (pp300-305)

Image Interpolation using Consistent Neighborhood Correlations

RONEN SHER and MOSHE PORAT
Department of Electrical Engineering
Technion – Israel Institute of Technology
Haifa 32000
ISRAEL
http://visl.technion.ac.il/~sherr
http://visl.technion.ac.il/mp

Abstract: - We present a new approach to image interpolation using consistent relationships between adjacent pixels in an image. In the first stage, the localized relationships are learned from the input image. In the second stage, the relationships and the concluded governing rules are used to build an interpolated image. Our method is compared with other interpolation methods such as bilinear, bi-cubic and spline for enlargement of images by multiplications of 2. The results indicate significant reduction in the blockiness and smoothing effects compared to existing methods. We present additional results for related applications of audio signals and for color imaging.

Key-Words: Image compression, Image reconstruction, Markov process, Least square method.

1 Introduction

Image interpolation is of interest in many applications of image transmission and display. Several known methods exist for the implementation of these imaging systems. One basic approach is viewing the sampling and the interpolation scheme as projection onto a representative space, usually built from splines [1],[2]. Another approach is to solve partial differential equations to assess the missing information [3].

In this paper we propose and develop a new approach to this basic task, exploiting the high correlation between adjacent pixels.

1.1 Motivation

For a given square image, we can see a relation between its high-resolution version I_b of size $2M$x$2M$ and its lower resolution version I_s with size MxM. If we consider the size reduction in a straightforward manner, the pixels of I_s are contained in I_b, or more specifically, if k and l are the row and column indexes of the image respectively, then $I_b(2k-1,2l-1) = I_s(k,l)$. Obviously, I_b contains additional $3M^2$ more pixels compared to I_s (Figs. 1 and 2), however, although I_s has only a quarter of the pixels in I_b, we can still see the resemblance of their histograms (Fig.3).

Other motivating results are those of the JPEG Lossless algorithm [4] that show that the average compression ratio is less than half of the original image without loss of information. Turning to the standard version of JPEG, a compression ratio of 4 can be obtained with minimal loss of information, namely, maximal change of 3 gray levels for each pixel. Observing those results, it is clear that major information of the image is contained in I_s.

p_{11}	p_{12}	p_{13}
p_{21}	p_{22}	p_{23}
p_{31}	p_{32}	p_{33}

Fig.1: The original image (input).

p_{11}		p_{12}		p_{13}
p_{21}		p_{22}		p_{23}
p_{31}		p_{32}		p_{33}

Fig.2: The larger image. The gray pixels represent the original data from Fig. 1.

2 Image Sub Regions

It is a widely accepted observation that images' structure can be modeled by a Markov process [5]. This model predicts a gray level value of a pixel by a weighted sum of its neighbors and additive noise. Moreover, in image reconstruction the simplest, yet acceptable, way to enlarge an image is by bilinear interpolation, i.e., each missing pixel is predicted by the average of its four surrounding neighbors, and Bi-Cubic interpolation uses sixteen neighbors, both are also a weighted sum.

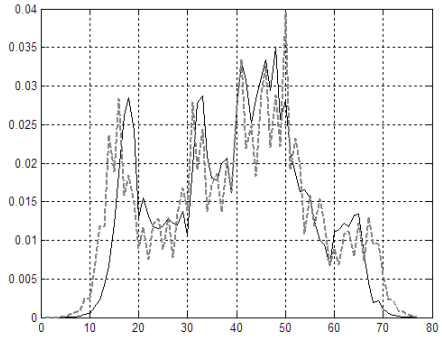

Fig.3: Lena normalized histograms of size (512x512) – dashed, and of size (256x256) – solid.

n_1^x	n_1^+	n_2^x
n_4^+	p	n_2^+
n_4^x	n_3^+	n_3^x

Fig. 4: An unknown pixel p, and its known neighbors of type "x" and "+".

According to the structure in Fig. 2 we separate the missing pixels in the larger image into two types: "x" pixels are the ones that have 4 known neighbors in each diagonal corner, and the "+" type pixels which have only two known neighbors in the horizontal edge or in the vertical edges of the image (Fig.4). Our goal is to predict the missing pixels by the most fitting weighted sum of their neighbors, i.e.,

$$\tilde{p} = \sum_i \alpha_i n_i \qquad (1)$$

where $\tilde{p}$ is the reconstructed pixel, n_i are its neighbors and α_i are the weighted coefficients.

To construct a robust model of an image we must consider the regions characteristics such as edges and smooth areas, this will involve changes in the pixels' gray levels. E.g., in case of a horizontal edge, n_1^x will be greater than n_4^x and $n_2^x > n_3^x$, similarly, if there is a vertical edge one gets $n_1^x > n_2^x$ and $n_3^x > n_4^x$, similar relationships will hold for the "+"neighbors. Quantitatively, the sharper the edge, the higher the variance of the neighbors' values. Diagonal edges are also considered to complete the model. For each region, the predicted value is a weighted sum with a different weight for each neighbor, depending on their relationships. If we limit ourselves to the 4 surrounding neighbors, there will be 4!=24 permutations, i.e., 1) $n_1^x > n_2^x > n_3^x > n_4^x$ 2) $n_1^x > n_2^x > n_4^x > n_3^x$ … 24) $n_4^x > n_3^x > n_2^x > n_1^x$.

As a sequel in each permutation there will be different coefficients in the weighted sum:

$$p^x = \sum_{i=1}^{4} \alpha_i^x n_i^x \qquad (2)$$

Similar observation is valid for the "+" neighbors, where the prediction is:

$$p^+ = \sum_{i=1}^{4} \alpha_i^+ n_i^+ \qquad (3)$$

Our assumption is that the coefficients α_i which model the small image will hold for the larger image too, thus can be used to estimate the missing pixels.

3 The Proposed Algorithm

Our algorithm is based on three major parts. First the governing rules of the data are studied and analyzed from the input image for the two types of pixels.

The output image is built in two steps. In the first step M^2 pixels are reconstructed, whereas the third part, which is based on the second, reconstructs the additional $2M^2$ pixels. The detailed algorithm is as follows.

1. The algorithm first scans the input image. The type of each pixel is determined to be "x" or "+". For each permutation we save the four x-neighbors and the matching pixel in a two-dimensional array, which we call Value Metric of the x-type (VMx). Since we want to model only the regions with significant changes in gray levels, we also add the constraint that the variance of the neighbors will be greater than a predefined reference value, i.e., $V = Var\left\{n_i^x\right\}_{i=1}^4 \geq \delta$.

2. During the scanning process, we also save the four "+" neighbors and the matching pixel in a different array VM+ with similar separation to 24

permutations, depending on their relations, and the same constraint of variance is used.

3. For **each** permutation, we find the coefficients in the weighted sum (2). We build those equations from VMx. As a consequence, there are 24·4=96 missing variables, however, the number of equations is the number of pixels in the input image without the pixels in the 4 sides of the image, i.e., M^2-4M, and without the smooth areas pixels $(V < \delta)$, so theoretically for an image with M>>12 which is mainly the case of interest, there is a solution for those equations, however, a further constraint is necessary to solve the problem.

Let us denote j to be the permutation index (j=1..24), and Q to be the total number of occurrence in the permutation j. Thus, for the permutation j, we can write Equation (2) Q times:

$$\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_Q \end{bmatrix} = \begin{bmatrix} n_{11} & n_{12} & n_{13} & n_{14} \\ n_{21} & n_{22} & n_{23} & n_{24} \\ \vdots & \ddots & & \vdots \\ n_{M1} & n_{M2} & n_{M3} & n_{M4} \end{bmatrix} \cdot \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} \qquad (4)$$

where the indexes x and j were dropped to simplify the expression. Or equivalently in matrices form:

$$\underline{\mathbf{P}^{\mathrm{x}}} = \underline{\underline{\mathbf{n}}} \cdot \underline{\boldsymbol{\alpha}} \qquad (5)$$

In our case of interest the input images have M>>12 thus, for each permutation we will have 4 missing variables and additional more equations. Since such a problem does not necessarily have a unique solution, the least square solution could be helpful:

$$\underline{\boldsymbol{\alpha}} = \left[\underline{\underline{\mathbf{n}}}^T \underline{\underline{\mathbf{n}}}\right]^{-1} \underline{\underline{\mathbf{n}}}^T \underline{\mathbf{P}^{\mathrm{x}}} \qquad (6)$$

This will produce the 24 vectors of the x-pixels coefficients - $\left\{\underline{\boldsymbol{\alpha}}_{\mathbf{j}}^{\mathrm{x}}\right\}_{j=1}^{24}$.

A few remarks on this solution:

- Due to the inverse operation we must ensure that $\left[\underline{\underline{\mathbf{n}}}^T \underline{\underline{\mathbf{n}}}\right]$ will have no zeros rows or/and columns. To satisfy this condition a simple preprocessing step can be performed: If the minimum value of the input is zero (non-negative images) we add the value 1 to the whole image, which can be subtracted from the output at the end. Since $\left[\underline{\underline{\mathbf{n}}}^T \underline{\underline{\mathbf{n}}}\right]$ is built from the input pixels, it will be invertible.
- The least square method will produce a solution that is more immune to noise and to side effects. Therefore it is desired that the number of equations in (4) will be significantly higher than four, or equivalently that the number of pixels in the input image will be significantly greater than 12^2.

4. For the "+" type pixels we use the same technique with VM+ to solve (3), which will yield the 24 vectors of coefficients - $\left\{\underline{\boldsymbol{\alpha}}_{\mathbf{j}}^{+}\right\}_{j=1}^{24}$

5. After the coefficients for the x-pixels $\left\{\underline{\boldsymbol{\alpha}}_{\mathbf{j}}^{\mathrm{x}}\right\}_{j=1}^{24}$ are found, we use them to reconstruct the x-pixels in the enlarged image. This is done by scanning the sparse image and for each x-pixel we find its matching permutation j from its known neighbors' relations, and calculate its prediction value using Equation (2). Here, we include another permutation for smooth areas $(V < \delta)$, in which the reconstruct value is the simple average of its neighbors: $p^{\mathrm{x}} = 0.25 \cdot \sum_{i=1}^{4} n_i$.

6. This step is based on the previous one, the input image is I_{x} and its structure is as shown in Fig.5. For each "+" pixel we find its matching permutation from its known neighbors, calculate its value from (3), and for cases where $V < \delta$ we use a simple average. Notice that for each "+" pixel only two of its neighbors were present in the input image and the other two were reconstructed in Step 5.

7. Presently we have the output image of size 2Mx2M (its structure is shown in Fig.6). If we have added 1 to the input we can subtract it now, although the visible effect can be ignored. Finally to complete the sides of the output image, a simple average of the surrounding pixels is used.

p_{11}		p_{12}		p_{13}
	$\tilde{p}^{\mathrm{x}}$		$\tilde{p}^{\mathrm{x}}$	
p_{21}		p_{22}		p_{23}
	$\tilde{p}^{\mathrm{x}}$		$\tilde{p}^{\mathrm{x}}$	
p_{31}		p_{32}		p_{33}

Fig.5: The image I_{x} after Step 5 of the algorithm.

The algorithm can be generalized to arbitrary sized images. In such cases the input image size is M_1xM_2 and the output will be 2M_1x2M_2.

p_{11}	$\tilde{p}^{+}$	p_{12}	$\tilde{p}^{+}$	p_{13}
$\tilde{p}^{+}$	$\tilde{p}^{x}$	$\tilde{p}^{+}$	$\tilde{p}^{x}$	$\tilde{p}^{+}$
p_{21}	$\tilde{p}^{+}$	p_{22}	$\tilde{p}^{+}$	p_{23}
$\tilde{p}^{+}$	$\tilde{p}^{x}$	$\tilde{p}^{+}$	$\tilde{p}^{x}$	$\tilde{p}^{+}$
p_{31}	$\tilde{p}^{+}$	p_{32}	$\tilde{p}^{+}$	p_{33}

Fig.6: The structure of the output image after Step 6 of the algorithm.

To enlarge an image by a factor greater than 2 we use the algorithm iteratively, until we reach the desired size. Another generalization is interpolation of color images, where each color component can be treated as an individual gray level image.

4 Experiments

First we show the resembling values of the x-kind coefficients $\left\{\underline{\boldsymbol{\alpha}}_{\mathbf{j}}^{\mathbf{x}}\right\}_{j=1}^{24}$ in the large and small image for Lena with size 512x512 and its reduced version of size 256x256. This will correlate with our basic assumption in Section 2. In each image there are 24 coefficients vectors, each vector has 4 elements. Fig. 7 shows the results: at top left the first elements of the 24 vectors are shown, top right depicts the second elements; bottom left is for the third elements and bottom right for the fourth elements.

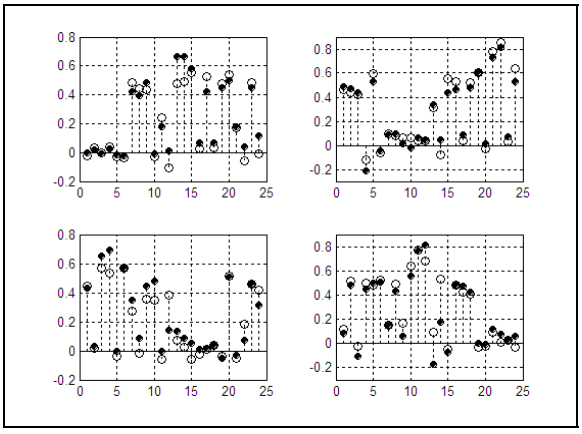

Fig.7: The coefficients values.
Larger image – black disks, Small image – empty circles.

We compared the proposed algorithm with bilinear , Bi-Cubic and Bi-Cubic Spline interpolations on a set of images with input size 128x128 and the resulting output size is 256x256 (Figs. 8-12).

The value $\delta = 5$ was chosen to yield the best results. As can be seen, significant improvement can be observed in the blockiness and smoothness of the proposed approach verses other interpolation methods.

5 Additional Applications

The algorithm can be used for interpolation of one dimension signals where the coefficients are learned from the four neighbors of the input. An example is shown in Fig. 13 for voice signal, where the original even samples of the signal were replaced with zeros and the coefficient were learned from the odd samples I_{odd} (the input), i.e., for the sample $I_{odd}(t)$, the four neighbors are $I_{odd}(t-1)$, $I_{odd}(t-2)$, $I_{odd}(t+1)$, $I_{odd}(t+2)$, where t is the samples index.

For CCD images with Bayer pattern [6], the input has three components, each is a sparse image, the red and blue component has the structure of Fig.2 and the green component has the structure of Fig.5. The relations can be learned from each color and the suggested method can be used to reconstruct the missing pixels (Fig.14).

6 Conclusions

We have presented a new approach to image interpolation based on the high correlation between adjacent pixels. A step-by-step algorithm has been formulated to achieve an enlarged image of 4 times the input data. The results indicate a significant improvement over existing methods in side effects like smoothness and blockiness. Our method can be also applied to one dimension (voice) and multi-dimensions (CCD images) signals with very good results. Our conclusion is that the new approach and algorithm could be helpful in most applications of signal interpolation.

Fig.8: The input image.

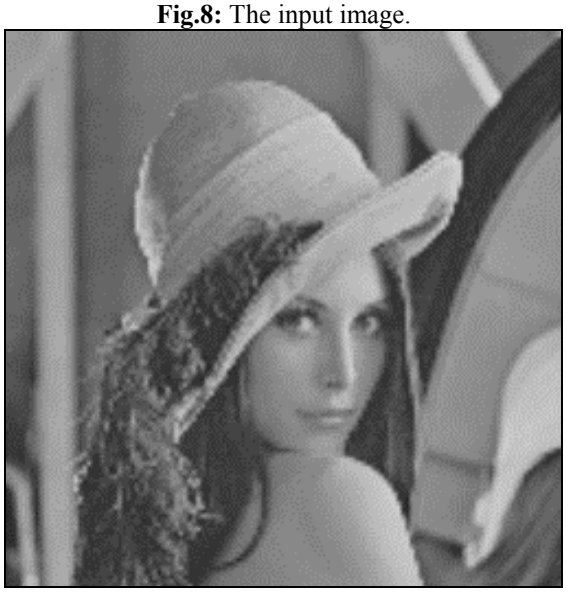

Fig.9: Bilinear interpolation.

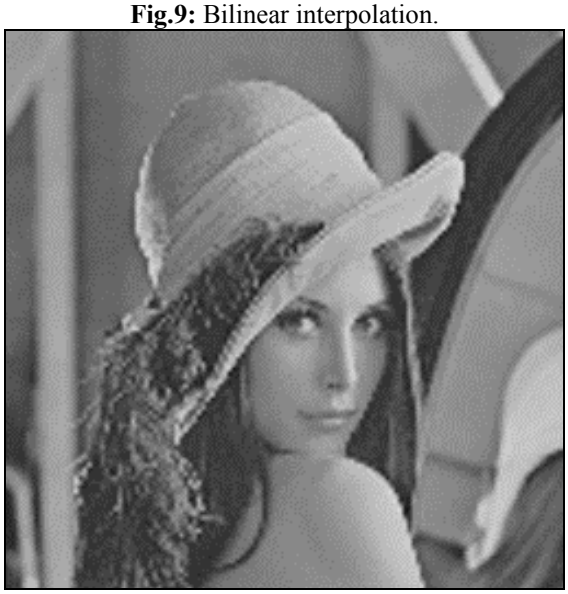

Fig.10: Bi-Cubic interpolation.

Fig.11: Bi-Cubic Spline interpolation.

Fig.12: Output of the proposed approach.

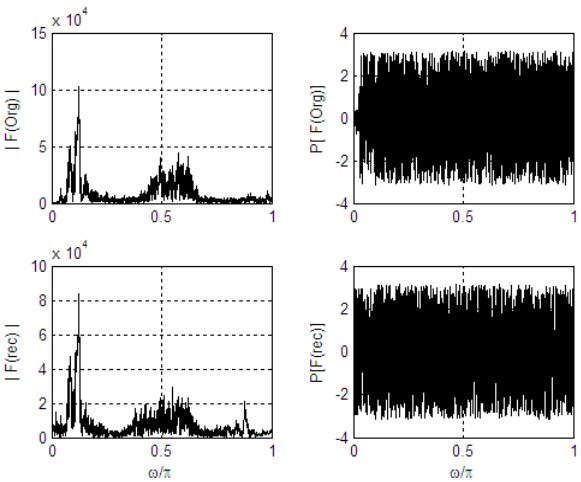

Fig.13: FFT of the word 'Diskette' (size 4225) Original and reconstructed versions.

Fig.14: CCD reconstruction for the image 'Lily' with size 186x230. Top: bilinear interpolation; Bottom: the new approach.

7 Acknowledgements

This research was supported in part by the HASSIP Research Program HPRN-CT-2002-00285 of the European Commission, and by the Ollendorff Minerva Centre. Minerva is funded through the BMBF.

References:

[1] M. Unser, "Splines: a perfect fit for signal and image processing," *IEEE Signal Processing Mag.*, vol. 16, pp. 22–38, 1999.

[2] T. Blu, P. Thévenaz, and M. Unser, "MOMS: Maximal-order interpolation of minimal support," *IEEE Trans. Image Processing*, vol. 10, pp. 1069–1080, 2001.

[3] Gilboa G., Sochen, N.A., and Zeevi Y.Y, Forward-and-backward diffusion processes for adaptive image enhancement and denoising. *IEEE Trans. Image Proc*. 11, No. 7, pp. 689-703, 2002.

[4] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Processing*, vol. 9, pp. 1309–1324, 2000.

[5] L. Ke, M.W. Marcellin, "Near-lossless image compression: minimum-entropy, constrained-error DPCM", *IEEE Trans. Image Processing*, vol. 7, pp. 225-228, 1998.

[6] Ron Kimmel, "Demosaicing: Image Reconstruction from Color CCD Samples", *IEEE Trans. on Image Processing,* Vol. 8, No. 9, pp. 1221-1228, 1999.