A Locally Adaptive Edge-preserving Algorithm for Image Interpolation

LEÍZZA RODRIGUES[1], DÍBIO LEANDRO BORGES[2], LUIZ MARCOS GONÇALVES[3]

[1]UFG – Universidade Federal de Goiás, Escola de Eng. Elétrica e de Computação, Praça Universitária s/n, Setor Universitário, CEP-74605-220 Goiânia, GO, Brasil
{lefer@cultura.com.br}
[2]PUC-PR – Pontifícia Universidade Católica do Paraná, Programa de Pós-graduação em Informática Aplicada – PPGIA, Rua Imaculada Conceição, 1155 - Prado Velho, 80215-901 Curitiba, PR, Brasil
{dibio@ppgia.pucpr.br}
[3]UFRN – Universidade Federal do Rio Grande do Norte, Depto. de Eng. de Computação e Automação, 59072-970 Natal, RN, Brasil
{lmarcos@dca.ufrn.br}

Abstract. Digital image interpolation techniques are frequently used to enlarge pictures, i.e. zooming in, and they are upon increasingly demand for developing product applications using digital still cameras. One common difficulty with conventional interpolation techniques is that of preserving details, i.e. edges, and at the same time smoothing the data for not introducing spurious artifacts. A definitive solution to this is still an open issue, although there are working methods in the market, see e.g. Parker et. al. [6], Sakamoto et. al. [8] for recent surveys. In this paper we propose a locally adaptive edge-preserving algorithm for image interpolation, which deals with this problem, and different than other methods shows how to compute local thresholds preserving edges and not destroying smoothness at the same time.

1. Introduction

An image zooming system is of essential interest in many applications, such as for entertainment, or scientific visualization and image analysis tasks. In order to produce an enlarged picture from a given one (i.e. zooming) an interpolation method is commonly used, and many are known from the basic literature, see for example Pratt [7], and Gonzalez & Woods [2].

A typical problem with most interpolation techniques is that although smoothing the data and keeping the low frequencies in the new zoomed picture, they are not able to enhance the high frequencies or preserve the edges equally well. Visually those problems will result in either blurring or blocking artifacts. A possible solution would need a sort of non-linear interpolation, taking into account the directional variation for maintaining the sharpness of the new enlarged image. Other proposed techniques work in this direction, e. g. Battiato et. al. [1] and also Hong et. al. [3], although they rely on heuristic global thresholds for deciding upon the types of edges to interpolate. In this paper we propose a new algorithm to deal with this problem, in a sense it performs a gradient controlled weighted interpolation. The main differences from work in the literature is that our algorithm works in a locally adaptive way, using the sensing of the edges and smoothness at the same time, and computing local thresholds to decide the interpolation. We have compared our algorithm with most of the common ones from the literature and the outcome shows that the results are very competitive. The algorithm is first proposed for gray scale images, although extension for working with color images is almost straightforward and it will be done in future work

The rest of the paper is organized as follows. Section 2 presents in detail the algorithm for zooming digital images. Section 3 shows a set of experiments made with typical images, and also presents a comparison with most algorithms from the literature. Section 4 presents the main conclusions and points to future work.

2. A Locally-adaptive Edge-preserving Zooming algorithm for Image Interpolation

Generally, zooming in digital images is performed by doubling the original image, although factors other than 2 could be devised, we will consider here enlarging a picture by 2. As mentioned before the purpose is to design a technique which preserves the sharpness of the original data, although maintaining smoothness where appropriate. For practical applications the solution would also have to be of low complexity, and free of global threshold choices since it should work on a large variety of scenes and lighting conditions. Our approach works on detecting possible edges and common directions, and then controlling the interpolation based on the local variance of the original image.

Our algorithm works in four (4) stages, which we have named 1) expansion; 2) edge preservation; 3) gradient controlled smoothing; 4) gradient controlled filling.

2.1 Expansion

This is the simplest stage performing an expansion of the original image **I** (n x n) pixels onto a grid **Z** (2n-1 x 2n-1). Figure.1 shows the two images, notice that the pairs of even coordinates of Z are all left undefined at this stage, and the pairs of odd coordinates have the same value brought onto Z by the original image.

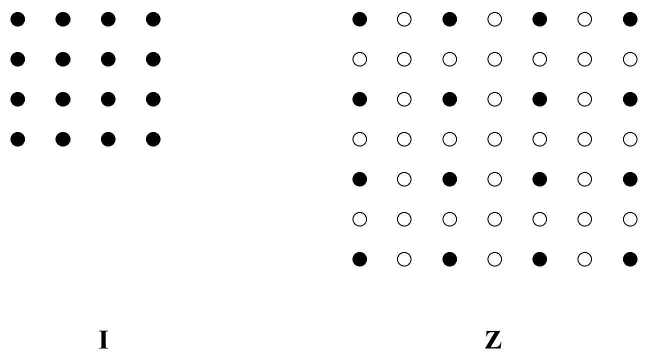

Figure.1 Expansion stage showing original image **I** (n x n) and grid **Z** (2n-1 x 2n-1).

For enlarging a picture for values bigger than 2, the algorithm can work sequentially making a doubling enlargement each step, 2, 4 8, and so on.

2.2 Edge Preservation

At this stage the idea is to sense the edges from the original image **I**, based on the lattice formed around the pairs of even coordinates of grid **Z**. This is a simple and fast way to sense the edge directions. More relative directions could be sensed using a different neighborhood, although the advantages of expanding the number of directions for the sake of precision have to be put against the cost of increasing the complexity of the algorithm. From our results, and from the work seen in the literature these do not seem to be an issue, and we favored the low complexity approach. Figure.2 shows the arrangement of this lattice for computing the interpolating values. Five (5) cases are sensed based on the local variance of the data. Figure.3 shows the five cases to be taken into consideration, where 3.a) represents no edge, 3.b) edge in the SW-NE direction, 3.c) edge in the NW-SE direction, 3.d) edge in the NS direction, 3.e) edge in the EW direction. In the process of producing an enlarged picture, our proposal is to consider the edges based on the local variance of the original data (Figure.2), for this two thresholds are computed locally, T_1 and T_2.

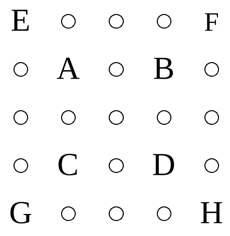

Figure.2 Local neighborhood of grid **Z** to be interpolated with points from **I**.

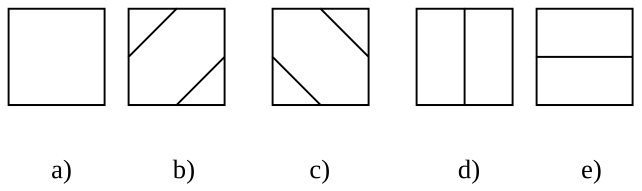

Figure.3 Five cases of presence of edges to be considered. a) no edge; b) edge in the SW-NE direction, 3.c) edge in the NW-SE direction, 3.d) edge in the NS direction, 3.e) edge in the EW direction.

Let us consider σ the standard deviation of points(A,B, C, D, E, F, G, H), our thresholds will be:

$$T_1 = \sigma \quad (1)$$

$$T_2 = \frac{\sigma}{\sqrt{2}} \quad (2)$$

Sensing a variation, either favoring one of the directions in the lattice chosen in the local neighborhood, or a smoothing factor when no direction is shown to be more present, is the function of the thresholds. The rationale of our choices for T_1 and T_2 is to perform interpolation with low squared error.

Based on the information sensed locally, this stage will perform a scan over the grid **Z** for interpolating the points whose coordinates are both even. Figure.4 shows a local layout of the pixels to be interpolated.

```
A   L1  B
L4  W   L2
C   L3  D
```

Figure.4 Layout of the local neighborhood to be interpolated at the stage “Edge Preservation”.

The steps will be the following:

```
1. For every pixel W (even coordinates of Z), do
    1.1 If |range(A,B,C,D)| < T1, then W = (A+B+C+D)/4
    1.2 If |A-D| > T2 & |A-D| >> |B-C|, then W = (B+C)/2
    1.3 If |B-C| > T2 & |B-C| >> |A-D|, then W = (A+B)/2
    1.4 If |A-D| > T1 & |B-C| > T1 & (A-D)*(B-C) > 0, then L1=(A+B)/2; L3=(C+D)/2
    1.5 If |A-D| > T1 & |B-C| > T1 & (A-D)*(B-C) < 0, then L4=(A+C)/2; L2=(B+D)/2
```

After this stage there will be many points left undefined. The algorithm will leave them and go to the third stage. After passing the four stages if there still be holes left other runs of the algorithm should be done only on the points left undefined, until all the points in **Z** are interpolated.

2.3 Gradient Controlled Smoothing

This stage will take care of points with at least one odd coordinate of **Z** left undefined until now. Figure.5 shows the layout of a local neighborhood with point Q being this point left undefined.

```
    L1
L4  Q   L2
    L3
```

Figure.5 Layout of local neighborhood at the third stage.

There will be two situations, first one where points L1 and L3 are known. In this case, do

```
1. If L2 or L4 are undefined, do
    1.1 If |L1-L3| < T1, then Q = (L1+L3)/2
2.If L2 and L4 are both known, do
    2.1 If |L1-L3| > T2 & |L1-L3| >> |L2-L4|, then Q = (L2+L4)/2
    2.2 If |L2-L4| > T2 & |L2-L4| >> |L1-L3|, then Q = (L1+L3)/2
Otherwise, leave Q undefined.
```

In the second situation, L2 and L4 are known, then do

```
1.If L1 or L3 are undefined, do
    1.1 If |L2-L4| < T1, then Q = (L2+L4)/2
2.If L1 and L3 are both known, do
    2.1 If |L2-L4| > T2 & |L2-L4| >> |L1-L3|, then Q = (L1+L3)/2
    2.2 If |L1-L3| > T2 & |L1-L3| >> |L2-L4|, then Q = (L2+L4)/2
Otherwise, leave Q undefined.
```

2.4 Gradient Controlled Filling

At the final stage the points left undefined in the interpolation will be filled, using an weighted value based on a bin histogram of the local neighborhood. The stage works as follows:

```
1.Compute m= (gray scale)/p bins of the image (e.g. gray scale = 256, p=16). Each i-th bin will include values from p(i-1) to pi-1
2.Find a pixel W, in Z, with both coordinates even, left undefined, do
    2.1 For each of the surrounding pixels A,B,C, and D, find to which bin they belong to, pick the median of each bin, and then W = mean(medians of bins of A, B, C, D)
3.Find a pixel Q, in Z, with at least one odd coordinate, left undefined, do
    3.1 For each of the surrounding pixels L1,L2,L3, and L4, find to which bin they belong to, pick the median of each bin, and then Q = mean(medians of bins of L1, L2, L3, L4)
```

If after the four stages there will be points in **Z** undefined, run algorithm starting from second stage again.

Next section will explain many experiments we have run with the proposed method, and six (6) others commonly used.

3. Experiments

In this section, we present some experiments done in order to demonstrate the effectiveness of the proposed method for the image interpolation problem presented in this work.

We start with a gray level version of the Lena image with 512x512 pixel size, as shown in Figure.6(a). A sample containing Lena's eye with size of 47x47 pixels is chosen to be zoomed, as seen in Figure.6(b). For the interpolation process inherent to the zooming, we have also used for comparison purposes, besides our method, other common algorithms from the literature, such as the nearest neighbor (NN), bilinear (BL), cubic (taking into consideration one (C-1) and two variables (C-2)), cubic b-spline (SPL), and linear (taking the correlation into consideration) (LC). Those are considered the most common techniques, and the best options available, see for details and other tests Lehmann et.al. [4], Maeland [5], Parker et.al.[6], and Sakamoto et.al. [8].

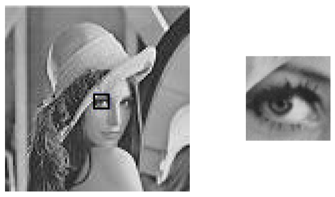
(a) (b)

Figure.6 (a) Original 512x512. (b) Sample 47x47.

The results of each interpolation method applied to the sample (Figure.6 (b)) can be seen in Figure.7. It can be noticed that the method proposed in the current work keeps the magnified image as smooth as possible while sharpening it. This is a consequence of using an interpolation based on the local adaptive procedure proposed, weighting smoothness and high-contrast. It makes a balancing between these two conditions.

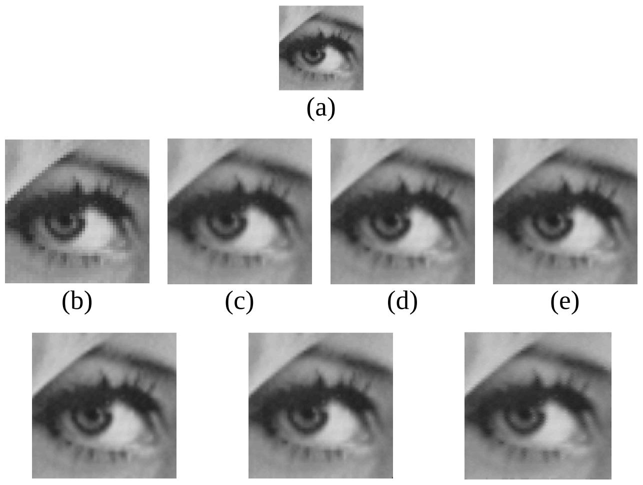
(a)
(b) (c) (d) (e)
(f) (g) (h)

Figure.7 – Zoomed images resulted from the interpolation methods applied to the sample of Lenna image. (a) Sample 47x47to be zoomed; Results using (b) Nearest neighbor (NN). (c) Bilinear. (BL) (d) Cubic 1-var (C-1). (e) Cubic 2-var (C-2). (f) Cubic B-spline. (SPL) (g) Linear taking correlation into consideration (LC) (g) Our Locally adaptive non-linear interpolation (LAI).

We conducted other experiments with different textured images to show the versatility of our approach. One such example is the aerial image shown in Figure.8. In the same way, a region of size 47x47 is chosen for the zooming process, seen in Figure.8(b). The results of each interpolation for comparison are shown in Figure.9.

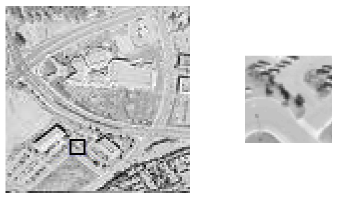
(a) (b)

Figure.8 (a) Original 512x512. (b) Sample 47x47.

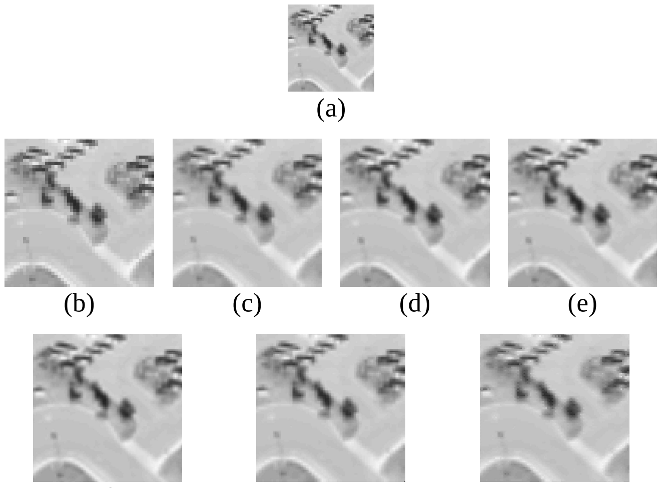

Figure.9 - Zoomed images resulted from the interpolation methods applied to the sample of Aerial image. (a) Sample 47x47to be zoomed; Results using (b) Nearest neighbor (NN). (c) Bilinear. (BL) (d) Cubic 1-var (C-1). (e) Cubic 2-var (C-2). (f) Cubic B-spline. (SPL) (g) Linear taking correlation into consideration (LC) (g) Our Locally adaptive non-linear interpolation (LAI)..

Although visually the results of our proposed technique can be considered good, it is difficulty only looking at the images from Figures 7 and 9, to make a rank of the most appropriate. One way commonly used to compare interpolation techniques is to compute a Signal to Noise Ratio (SNR), or a cross-correlation between an original image and the interpolated ones, as suggested in Battiato et. al. [1], Sakamoto et. al. [8].

For the process of comparing the interpolated images by cross-correlation, we will take a sample (as in Figure.6 (b)) with 47 x 47 pixels, apply a reduction by decimation by creating a new sample **I2** (23 x 23) pixels taking only the odd columns and rows of sample **I1** (47 x 47). Then, apply all the interpolation methods to zoom **I2** to a new image **Z1**. Cross-correlation will be applied to the pairs **Z1—I1**, and the methods with the highest values of cross-correlation will be more appropriate and effective, since they will approximate better the original image **I1**. Off course the reduction could introduce further variation in the images, which could favor or not certain interpolations. However, **I1** and **I2** are fixed and they are the same for all methods. As mentioned before this is a practical and commonly used method for comparison.

A cross-correlation value defined between a pair of images can be computed as:

$$CC = \left| \frac{\left(\sum_{1,1}^{k,l} A(i,j)B(i,j) - k.l.a.b \right)}{\sqrt{\left(\left(\sum_{1,1}^{k,l} A^2(i,j) - k.l.a^2 \right) \left(\sum_{1,1}^{k,l} B^2(i,j) - k.l.b^2 \right) \right)}} \right| \quad (3)$$

Where,

$A(i,j); B(i,j)$ are images to be correlated;

k,l are the number of columns and rows of the images;

a,b are the mean values of images A, and B respectively;

Table.1 shows the cross-correlation values computed for the two image samples (Lenna and Aerial). From the values shown is can be seen that the three which better approximate the original image, i.e. better overall interpolation results, are the Bilinear (CC-BL), Linear with correlation (CC-LC), and the method we propose here Locally Adaptive Interpolation (CC - LAI).

	Lenna sample	Aerial sample
CC - NN	0.9515	0.8300
CC – BL	0.9933	0.9226
CC – C-1	0.9746	0.8792
CC – C-2	0.9746	0.8792
CC - SPL	0.9746	0.8792
CC - LC	0.9931	0.9208
CC - LAI	0.9902	0.9093

Table.1 – Cross-correlation values computed between the original samples of images Lenna, and Aerial, and the interpolated ones by each method. CC-NN: Nearest Neighbor. BL: Bilinear, C-1: Cubic 1 var., C-2: Cubic 2-var, SPL: Cubic Spline, LC: Linear with correlation, LAI: our Locally Adaptive Interpolation.

The method we have proposed here is quite competitive as can be seen from the results, because besides realizing a good qualitative zooming it is fast and easy to implement. Different than other methods the values (as thresholds) are set locally and dependent on the data. Next section points to conclusions and future work.

4. Conclusions

In this paper we have proposed a new method for realizing interpolation in digital gray scale images. Other works fom the literature, such as Battiato et. al. [1], and Hong et. al. [3] have also proposed interpolation techniques which aimed at keeping edges and details while smoothing.

Our work is similar to those in these aspects, however it incorporates three new important features: first, it models the edges based on a larger neighorhood, two concentric squares, in order to give more coherence to the interpolating decision nods; second, it proposes a way to compute the thresholds needed for those decisions based on the variance of the data, and third it computes the thresholds locally, making it widely applicable.

We have showed tests according to the commonly used in the area, such as computing a cross-correlation to measure the best interpolation methods. Our Locally Adaptive Interpolation (LAI) method ranked as one of the best choices, with the advantage of easiness to implement and low complexity.

As future works it would be interesting to check the performance of the technique as first approximations of super-resolution, or for active vision applications, since it is aimed for real time use.

Acknowledgements

Author L. Rodrigues is supported by a bursary from CAPES – MEC, program DS/PROAP. We thank Dr. Rodrigo Lemos (EEE-UFG) for all the support given to L. Rodrigues throughout this work.

References

[1] S. Battiato, G. Gallo, and F. Stanco, "A New Edge-Adaptive Zooming Algorithm for Digital Images", in *Proc. Signal Processing and Communications SPC 2000*, pp. 144—149, Spain, 2000.

[2] R. C. Gonzalez and R. E. Woods, *"Digital Image Processing"*, Addison-Wesley, USA, 1993.

[3] K. Hong, J. Paik, H. Kim and C. Lee, "An edge-preserving image interpolation system for a digital camcorder", *IEEE Transactions on Consumer Electronics*, vol. 42, n. 3, pp.279--284, Aug 1996.

[4] T. Lehmann, C. Gonner, and K. Spitzer, "Survey: interpolation methods in medical image processing", *IEEE Transacions on Medical Imaging*, vol. 18, n.11, pp. 1049—1075, Nov 1999.

[5] E. Maeland, "On the comparison of interpolation methods", *IEEE Transactions on Medical Imaging*, vol.7, n.3, pp. 213—217, Sep 1988.

[6] J. Parker, R. Kenyon, and D. Troxel, "Comparison of interpolating methods for image resampling*", IEEE Transactions on Medical Imaging*, vol.2, n.1, pp.31—39, 1997.

[7] W. Pratt, *"Digital Image Processing"*, 2nd ed., John Wiley, USA, 1991.

[8] T. Sakamoto, C. Nakanishi and T. Hase, "Software pixel interpolation for digital still cameras suitable for a 32-bit MCU", *IEEE Transactions on Consumer Electronics*, vol. 44, n. 4, pp.1342--1351, Nov 1998.