Morphological Image Interpolation to Magnify Images with Sharp Edges

Valérie De Witte[1], Stefan Schulte[1], Etienne E. Kerre[1]
Alessandro Ledda[2], and Wilfried Philips[2]

[1] Ghent University
Department of Applied Mathematics and Computer Science
Fuzziness and Uncertainty Modelling Research Unit
Krijgslaan 281 (Building S9), B-9000 Gent, Belgium
[2] Telin Department - IPI Group
Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium
Valerie.DeWitte@UGent.be
http://www.fuzzy.ugent.be

Abstract. In this paper we present an image interpolation method, based on mathematical morphology, to magnify images with sharp edges. Whereas a simple blow up of the image will introduce jagged edges, called 'jaggies', our method avoids these jaggies, by first detecting jagged edges in the trivial nearest neighbour interpolated image, making use of the hit-or-miss transformation, so that the edges become smoother. Experiments have shown that our method performs very well for the interpolation of 'sharp' images, like logos, cartoons and maps, for binary images and colour images with a restricted number of colours.

1 Introduction

Image interpolation has many applications such as simple spatial magnification of images (e.g. printing low-resolution documents on high-resolution printer devices, digital zoom in digital cameras), geometrical transformation (e.g. rotation), etc. Different image interpolation methods have already been proposed in the literature, a.o. [[1] - [11]]. In this paper we will describe a new morphological image interpolation technique, for binary images as well as for colour images with a limited number of colours, with sharp edges. First we review some basic definitions of mathematical morphology, including the hit-or-miss transformation. In section 3 we introduce our new interpolation approach. We explain the pixel replication or nearest neighbour interpolation, used as the first 'trivial' interpolation step in our method. Thereafter we discuss our corner detection method, using different kinds of structuring elements, and describe our corner validation, first for magnification by a factor 2 and then for magnification by an integer factor $n > 2$. Finally, in section 4 we have compared our interpolation method experimentally to other well-known approaches. The results show that our method provides a visual improvement in quality on existing techniques: all jagged effects are removed so that the edges become smooth.

2 Basic Notions

2.1 Modelling of Images

A digital image I is often represented by a two-dimensional array, where (i, j) denotes the position of a pixel $I(i, j)$ of the image I.

Binary images assume two possible pixel values, e.g. 0 and 1, corresponding to black and white respectively. White represents the objects in an image, whereas black represents the background. Mathematically, a 2-dimensional binary image can be represented as a mapping f from a universe X of pixels (usually X is a finite subset of the real plane $\mathbb{R}^2$, in practice it will even be a subset of $\mathbb{Z}^2$) into $\{0, 1\}$, which is completely determined by $f^{-1}(\{1\})$, i.e. the set of white pixels, so that f can be identified with the set $f^{-1}(\{1\})$, a subset of X, the so-called domain of the image. A 2-dimensional grey-scale image can be represented as a mapping from X to the universe of grey-values $[0, 1]$, where 0 corresponds to black, 1 to white and in between we have all shades of grey. Colour images are then represented as mappings from X to a 'colour interval' that can be for example the product interval $[0, 255] \times [0, 255] \times [0, 255]$ (the colour space RGB). Colour images can be represented using different colour spaces; more information about colour spaces can be found in [13], [14].

2.2 Binary and Colour Morphology

Consider a binary image X and a binary structuring element A, which is also a binary image but very small in comparison with X. The translation of X by a vector $y \in \mathbb{R}^n$ is defined as $T_y(X) = \{x \in \mathbb{R}^n \mid x - y \in X\}$; the reflection of X is defined as $-X = \{-a \mid a \in X\}$.

Definition 1. *The binary dilation $D(X, A)$ of X by A is the binary image*

$$D(X, A) = \{y \in \mathbb{R}^n \mid T_y(A) \cap X \neq \emptyset\}.$$

The binary erosion $E(X, A)$ of X by A is defined as

$$E(X, A) = \{y \in \mathbb{R}^n \mid T_y(A) \subseteq X\}.$$

The dilation enlarges the objects in an image, while the erosion reduces them.

Property 1. *If A contains the origin (i.e. $\mathbf{0} \in A$), then*

$$E(X, A) \subseteq X \subseteq D(X, A).$$

With this property the binary image $D(X, A) \backslash E(X, A)$ can serve as an edge-image of the image X, which we call the morphological gradient $G^A(X)$ of X. Analogously we define the external morphological gradient $G^{A,e}$ and the internal morphological gradient $G^{A,i}$ as

$$G^{A,e}(X) = D(X, A) \backslash X \quad \text{and} \quad G^{A,i}(X) = X \backslash E(X, A),$$

which will give us the external and inner edge-image of X respectively. We will use the internal morphological gradient to detect the positions of jaggies in an image. More information about binary morphology can be found in [[15] - [17]]. In [18] we have extended the basic morphological operators dilation and erosion to vector morphological operators for colour images by defining a new vector ordering for colours in the RGB colour space. Therefore we will also use the notations $D(X, A)$ and $E(X, A)$ for colour images.

2.3 The Hit-or-Miss Transformation

Consider again a binary image X and two binary structuring elements A and B. The hit-or-miss operator is defined by

$$X \otimes (A, B) = E(X, A) \cap E(X^c, B),$$

where X^c is the complement of X w.r.t. $\mathbb{R}^n$. The result is empty if $A \cap B \neq \emptyset$. The name hit-or-miss operator can be explained as follows: a pixel h belongs to the hit-or-miss transformation $X \otimes (A, B)$ iff $T_h(A)$ does not hit (intersect with) X^c and $T_h(B)$ does not hit X. The hit-or-miss operator is very useful for the detection of points inside an image with certain (local) geometric properties, e.g. isolated points, edge points, corner points.

As an example we show the detection of the upper-left corner points of objects in an image:

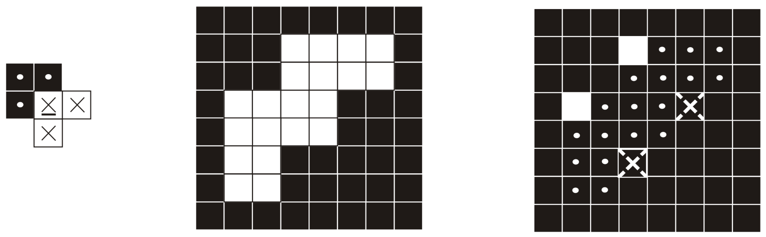

Fig. 1. From left to right: The structuring elements (A, B), where A contains the white pixels ($\times$) and B the black pixels ($\cdot$), the original binary image X and the hit-or-miss transformation $X \otimes (A, B)$ (only the white pixels)

3 New Morphological Image Interpolation Method

3.1 Pixel Replication or Nearest Neighbour Interpolation

The resolution of an image is the number of pixels per unit area, and is usually measured in pixels per inch. When we magnify an image V times, the number of pixels will increase (V^2 times). The easiest way to enlarge an image is to copy the existing pixel values to the new neighbouring pixels. If we magnify an image V times, one pixel in the original image will be replaced by a square of $V \times V$ pixels in the new image. This is called pixel replication or nearest neighbour interpolation. The result is quite poor, but we can use it as a first 'trivial' interpolation step.

3.2 Corner Detection

To detect the unwanted jaggies in the nearest neighbour interpolated image, we first determine the inner edge-image of the blown up image and then apply the hit-or-miss operator to obtain the positions of the object corner edge pixels. The advantage of the internal morphological gradient is that this gradient will give the correct pixel positions of corners in an image. When our original image is a binary image, the blown up image will also be a binary image, and so will the inner edge-image. On the other hand, if our original image is a colour image, the inner edge-image of the blown up image will be a colour image, but we transform it into a binary image by giving all non-black pixels a white value.

Let's call O the nearest neighbour interpolated image of the original image X, O^c the complement of O, O_{edge} the inner edge-image of O, and O^c_{edge} the inner edge-image of the complement image O^c. With a given pair of structuring elements (A, B) we first determine the hit-or-miss transformation $O_{edge} \otimes (A, B)$ and secondly the hit-or-miss transformation $O^c_{edge} \otimes (A, B)$. This way we will not only detect corners of objects in O, but also corners of objects in O^c.

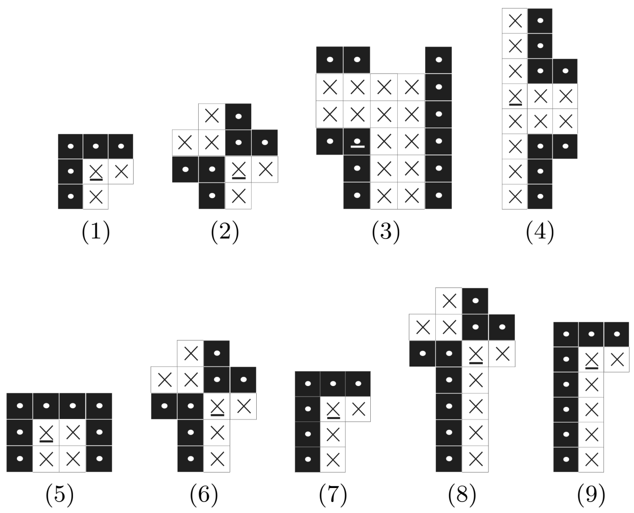

Fig. 2. The used structuring elements (1) (A_1, B_1) ... (9) (A_9, B_9) (the underlined element corresponds to the origin of coordinates) for magnification by a factor 2

Not all the corner pixels in the image should be changed, because some corners are 'real' corners, which have to be preserved in the magnified image, whereas others are part of jaggies and have to be removed. Therefore we will use different kinds of structuring elements in the hit-or-miss transformation. The structuring elements used for magnification by a factor 2 are shown in fig. 2, where A_i contains the white pixels ($\times$) and B_i the black pixels ($\cdot$), $i = 1 \ldots 9$. The other structuring elements are rotated or reflected versions of these. For example, the structuring elements (A_1, B_1) will allow us to detect all upper-left corner pixels, while using structuring elements that are properly rotated versions of (A_1, B_1) we will detect all upper-right, lower-left and lower-right corners. We not only look for corners of the 'foreground' objects, but also for corners of the 'background' objects. Consequently we will have 8 different kinds of corner pixels (4 'object' corner pixels and 4 'background' corner pixels). In the example in section 2.3 (fig. 1), not only the white pixels ($X \otimes (A, B)$) will be detected, but also the black pixels with white $\times$-symbol ($X^c \otimes (A, B)$).

3.3 Corner Validation

In this section we explain our method for magnification by a factor 2.

Step 1. We look for corners determined by the structuring elements (A_1, B_1) and their rotations. For example, if we detect an upper-left object corner pixel a or an upper-left background corner pixel a^* at position (i, j) in the edge-image O_{edge} (see figure 3). An upper-left object corner pixel will be determined by the hit-or-miss transformation $O_{edge} \otimes (A_1, B_1)$, while an upper-left background corner pixel will be determined by the hit-or-miss transformation of the complement of O_{edge}, i.e. $O^c_{edge} \otimes (A_1, B_1)$. Then we change the colour of the pixel at position (i, j) in the blown up image O by a mixture of the 'foreground' and 'background' colour of the surrounding pixels. We obtain this by adding the pixel values of the erosions $E(O, A_1)$ and $E(O, B_1)$ at position (i, j), that is, $O(i, j) = E(O, A_1)(i, j) + E(O, B_1)(i, j)$. Let $[r_c, g_c, b_c]$ and $[r_{c'}, g_{c'}, b_{c'}]$ be the RGB colour vector of the pixel $E(O, A_1)(i, j)$ and $E(O, B_1)(i, j)$ respectively, we define the new colour value of $O(i, j)$ with RGB components (r, g, b) as

$$r \overset{def}{=} (r_c + r_{c'})/2, \ g \overset{def}{=} (g_c + g_{c'})/2, \ b \overset{def}{=} (b_c + b_{c'})/2.$$

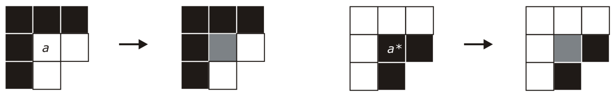

Fig. 3. Step 1 worked out for the structuring elements (A_1, B_1)

Step 2. We detect corners with the pair (A_2, B_2). If we detect such an object corner a or a background corner a^* at position (i, j) in O_{edge} (see figure 4), we fill the corner pixels at position $(i-1, j)$ and $(i, j-1)$ or at positions $(i-1, j-1)$ and (i, j), in the blown up image O with the foreground colour. Note that the foreground colour is defined by the minority colour in the image. Here, for the foreground colour of the pixel $O(i-1, j)$ we determine which of the two colour values $O(i-1, j)$ and $O(i, j)$ is less present in the image. The foreground colour of the pixel $O(i, j-1)$ is determined by the minority colour of the pixels $O(i, j-1)$ and $O(i, j)$. Analogously for $O(i-1, j-1)$ and $O(i, j)$.

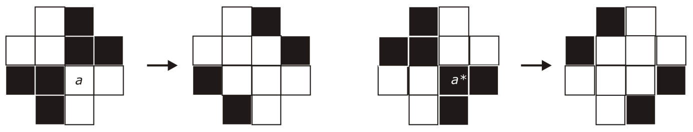

Fig. 4. Step 2 illustrated with structuring elements (A_2, B_2)

Step 3. We have added the structuring elements (A_3, B_3), (A_4, B_4), (A_5, B_5) and rotated or reflected structuring elements because we experienced that they are representative for the corner structures that should not be changed in an image. When we find a corner determined by one of these structuring elements, we leave the observed pixels unchanged in the interpolated image to avoid that real corners will be removed in the magnified image.

Step 4. We look at structuring elements of the form (A_6, B_6), (A_7, B_7) and rotated or reflected versions. See figure 5. In the first case (1), when an object corner a or background corner a^* is determined by (A_6, B_6) at position (i, j) in O_{edge}, we replace the pixel value at position $(i+1, j-1)$ or position $(i+1, j)$ in the image O by the colour with RGB components $r \overset{def}{=} (1/4 \times r_c + 3/4 \times r_{c'})$, g and b defined analogously, where $[r_c, g_c, b_c]$ and $[r_{c'}, g_{c'}, b_{c'}]$ are the RGB colour vectors of the pixels $E(O, A_1)(i, j)$ and $E(O, B_1)(i, j)$. In the second case (2), if a corner a or a^* is determined by the structuring elements (A_7, B_7) or by the structuring elements (A_6, B_6) in the special composition as illustrated in fig. 5(a) for a (for a^* we get an analogous figure), at position (i, j) in O_{edge}, we replace the original colour at position $(i+1, j)$ or position $(i+1, j-1)$ in O by the colour with RGB components $r \overset{def}{=} (3/4 \times r_c + 1/4 \times r_{c'})$, g and b analogous, where $[r_c, g_c, b_c]$ and $[r_{c'}, g_{c'}, b_{c'}]$ are the RGB colour vectors of the pixels $E(O, A_1)(i, j)$ and $E(O, B_1)(i, j)$. The colour value of $O(i, j)$ or $O(i, j-1)$ is changed to the RGB colour (r', g', b') with $r' \overset{def}{=} (1/4 \times r_c + 3/4 \times r_{c'})$, g' and b' analogous.

Step 5. At last we consider the pairs of structuring elements (A_8, B_8), (A_9, B_9) and their rotated or reflected structuring elements. When we find such an object

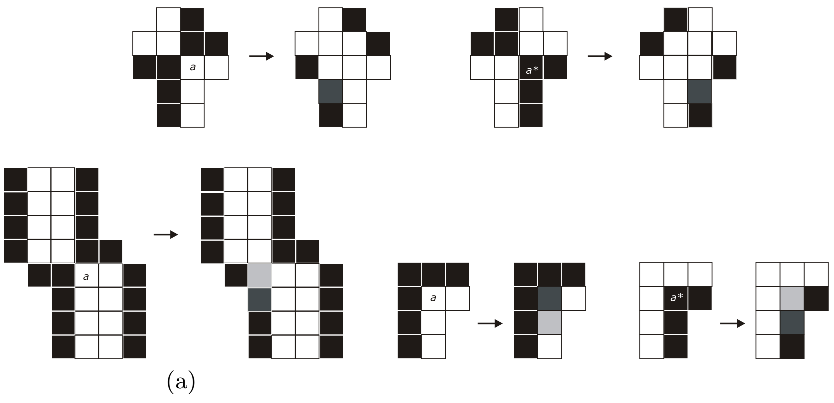

Fig. 5. Step 4, at the top: case (1) and at the bottom: case (2)

corner a (or background corner a^*) at position (i, j), we move the colour of pixel $O(i+1, j-1)$ $(O(i+1, j))$ to pixel $O(i+2, j-1)$ $(O(i+2, j))$ and give pixel $O(i+1, j-1)$ $(O(i+1, j))$ an intermediate colour value between the foreground and background colour of pixel $O(i, j)$.

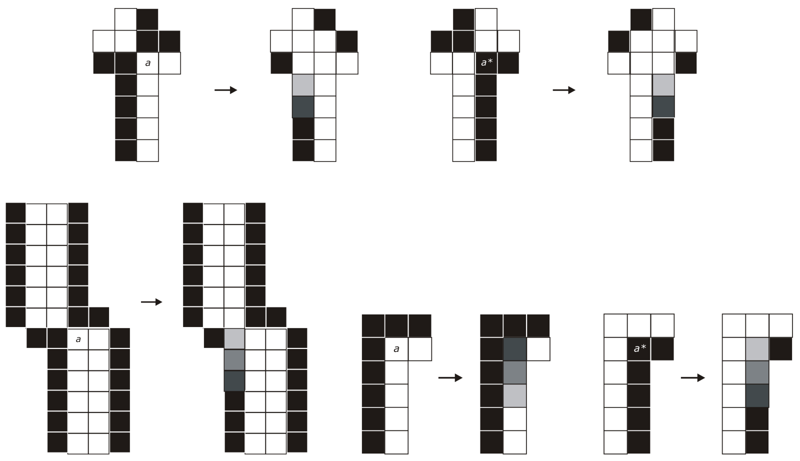

Fig. 6. Step 5, at the top: case (1) and at the bottom: case (2)

3.4 Magnification by an Integer Factor $n > 2$

Now, for magnification by an integer factor $n > 2$, we have to extend the structuring elements to a larger size but a similar shape, and the way of filling up the edge pixels will change a bit, but is very analogous. In figure 7 we have illustrated this process for magnification by a factor 3:

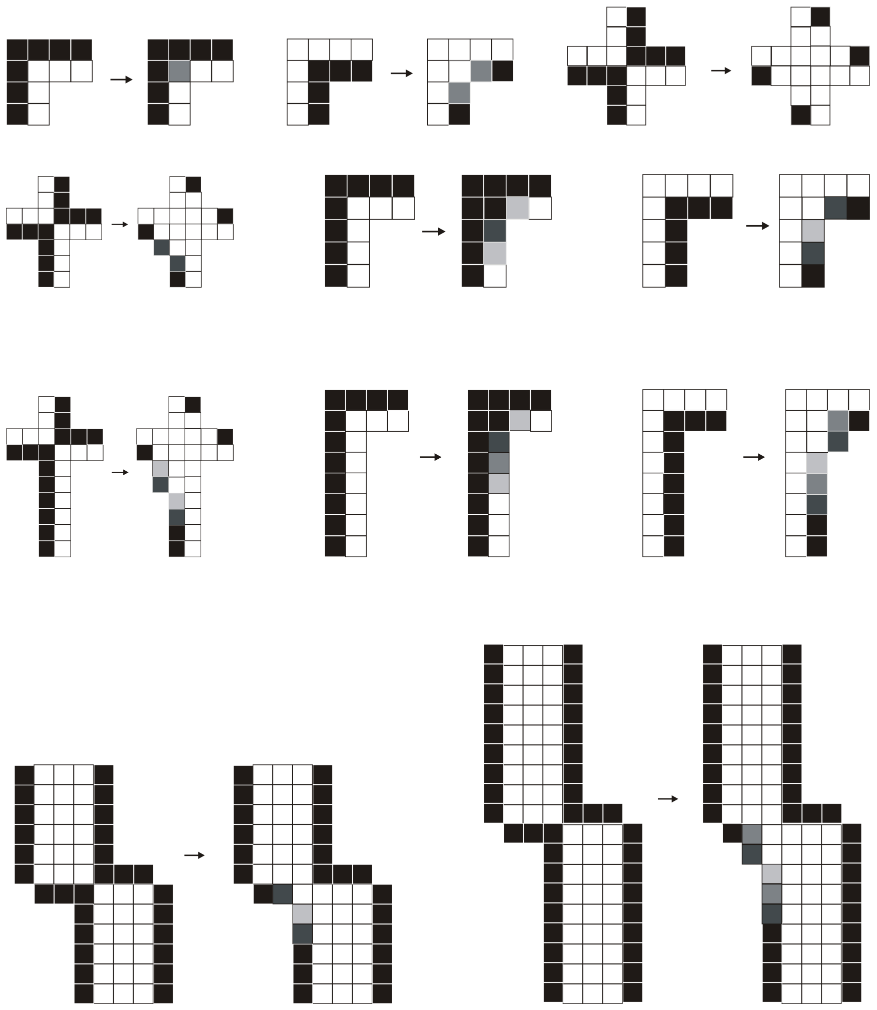
Fig. 7. The corner validation for magnification by a factor 3

4 Experimental Results

Fig. 8 and 9 show some results of our interpolation method.

Fig. 9 and 10 illustrate the result of several interpolation methods. We have compared our technique with the following state-of-the-art methods: the high-quality magnification filter Hq [10], which analyses the 3×3 area of the source pixel and makes use of lookup tables to get interpolated pixels of the filtered image, the anti-alias filter [12], which detects aliasing and changes only aliased edges

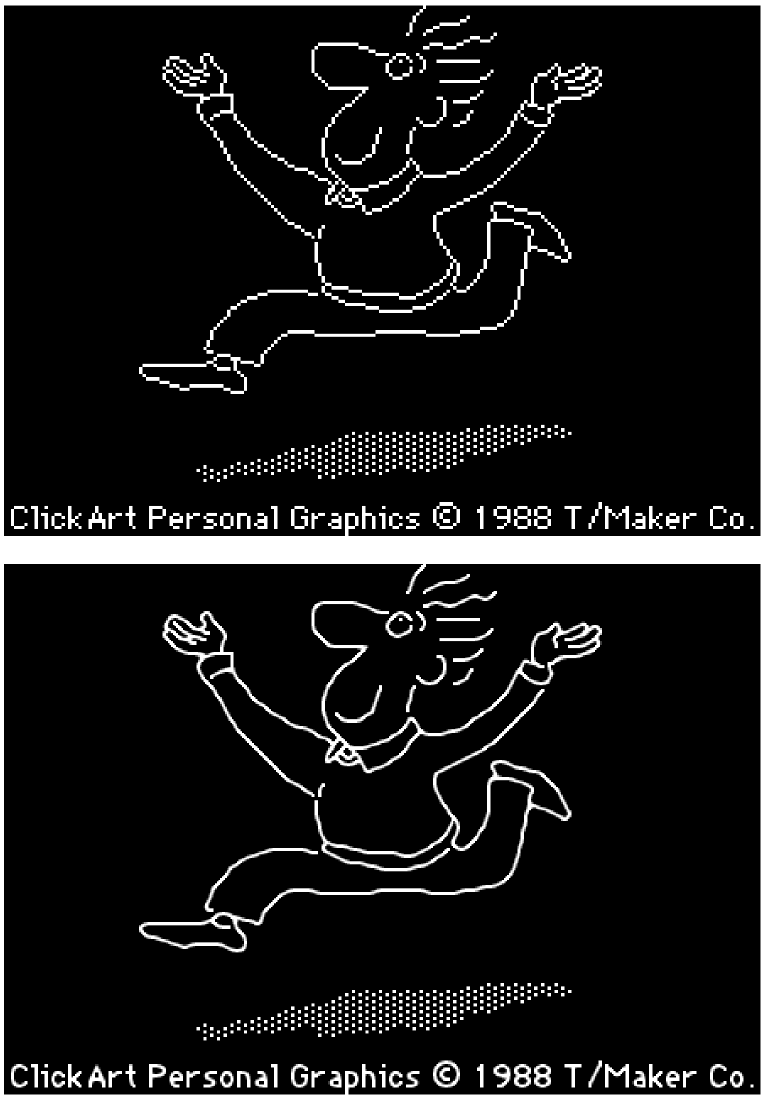
ClickArt Personal Graphics © 1988 T/Maker Co.
ClickArt Personal Graphics © 1988 T/Maker Co.

Fig. 8. At the top: the pixel replication 'cartoon' image for magnification by a factor 2, at the bottom: the result of our morphological interpolation method

and pixels, and some classical linear interpolation methods [1], in particular, bilinear and bicubic interpolation, which use the (weighted) mean of respectively 4 and 16 closest neighbours to calculate the new pixel value, and sinc interpolation, which makes use of windowed sinc functions. The main advantages and drawbacks of these linear interpolation filters are pointed out in [1].

We may conclude that our new method provides very beautiful results. Improvements in visual quality can be noticed: unwanted jaggies have been removed

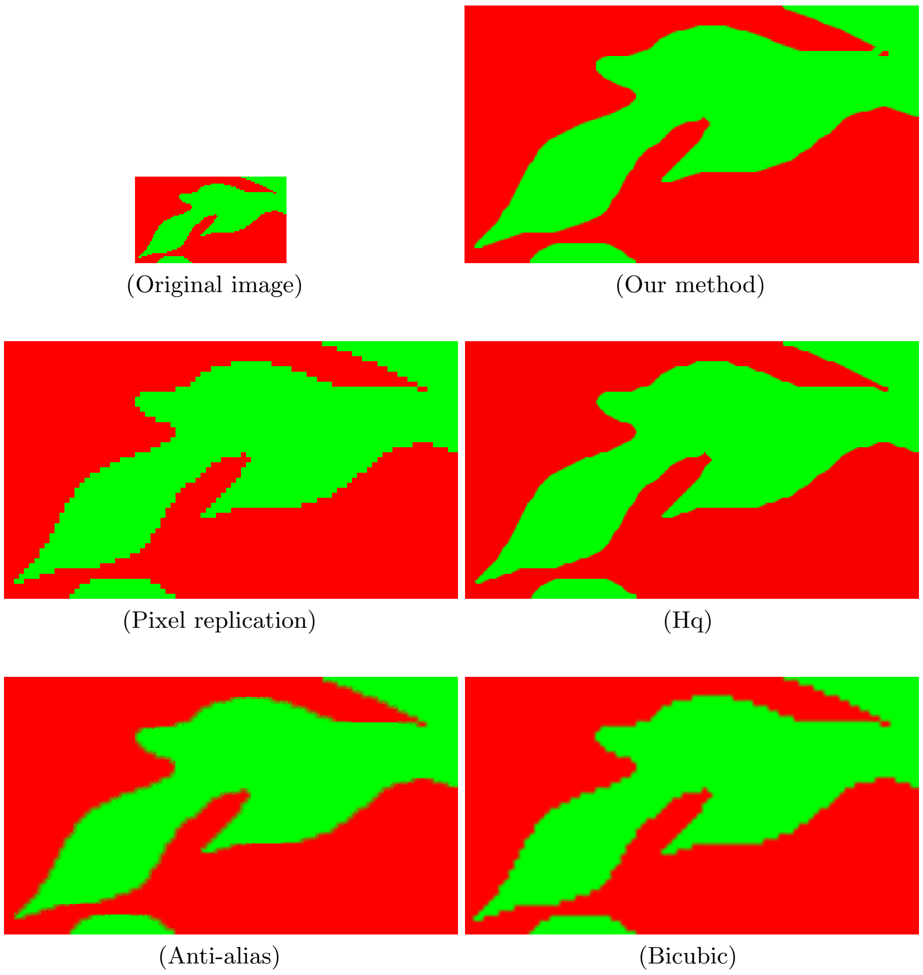

Fig. 9. Interpolation results for magnification by a factor 3

so that edges have become smoother. Good results are also obtained with the hq interpolation method, but our method outperforms all the others.

Our method was implemented in Matlab, which makes it hard to compare the computational complexity of this method with the others. As future work we will reimplement all methods in the same program language, Java or C++, to make them comparable with each other.

Remark: Sometimes it is desired that binary logos, cartoons and maps remain binary, or that no new colours are introduced in a colour image after magnification. Our method can also produce such a result: we only have to insert a threshold in our method, that is, in section 3.3 all new coloured pixels with RGB colour values greater than or equal to $(1/2 \times [r, g, b]_{\text{foreground colour}}$

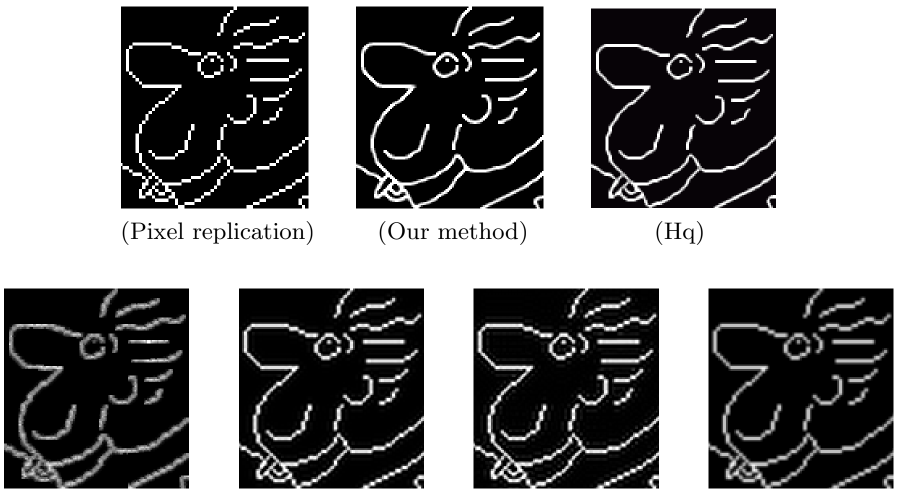

Fig. 10. Result of several interpolation methods for magnification by a factor 2

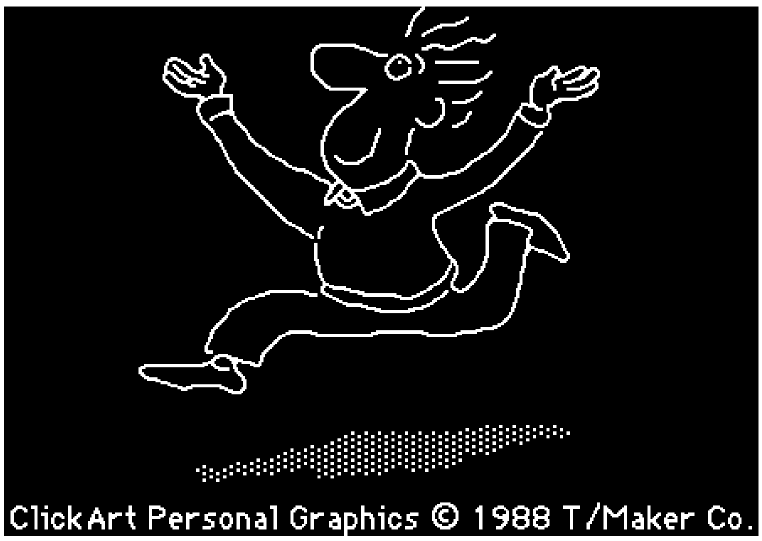

Fig. 11. Our new morphological interpolation method, giving as result a binary image of the original binary image ‘cartoon’ (magnification by a factor 2)

$+ 1/2 \times [r, g, b]_{\text{background colour}}$) are assigned to the foreground colour value, while all other colour pixels are transformed to the background colour value. The main visual improvements can be seen in fig. 11: the contours are smooth and text is also interpolated very well.

For a binary image interpolation method making use of mathematical morphology and giving a black-and-white result, we also refer to [19].

5 Conclusion

This paper presents a morphological method that improves the visual quality of magnified binary images with sharp boundaries. The problem of interpolating an image is the introduction of unwanted jagged edges in the blown up image. We developed a new approach to avoid these jaggies, making use of mathematical morphology. We also demonstrated that our method gives beautiful results for the magnification of colour images with sharp edges with a limited number of colours. As future work we will extend our approach towards all colour images with sharp edges, therefore we will define a new vector ordering for colours in the RGB colour space, and towards images with 'vague' edges, again using mathematical morphology.

Acknowledgement. This research was financially supported by the GOA-project 12.0515.03 of Ghent University.

References

1. Lehmann, T., Gönner, C., Spitzer, K.: Survey: Interpolations Methods In Medical Image Processing, IEEE Transactions on Medical Imaging, Vol. 18, No. 11, (1999) 1049–1075
2. Digital Image Interpolation: http://www.cambridgeincolour.com/tutorials/image-interpolation.htm
3. Allebach, J., Wong, P.W.: Edge-Directed Interpolation, Proceedings of the IEEE International Conference on Image Processing ICIP 96, Vol. 3, Switzerland, (1996) 707–710
4. Li, X., Orchard, M.T.: New Edge-Directed Interpolation, IEEE Transactions on Image Processing, Vol. 10, No. 10, (2001) 1521–1527
5. Muresan, D.D., Parks, T.W.: New Edge-Directed Interpolation, IEEE Transactions on Image Processing, Vol. 13, No. 5, (2004) 690–698
6. Tschumperlé, D.: PDE's Based Regularization of Multivalued Images and Applications, PhD thesis, Université de Nice, France, 2002
7. Morse, B.S., Schwartzwald, D.: Isophote-Based Interpolation, Proceedings of the IEEE International Conference on Image Processing ICIP 98, USA, (1998) 227-231
8. Luong, H.Q., De Smet, P., Philips, W.: Image Interpolation Using Constrained Adaptive Contrast Enhancement Techniques, Proceedings of the IEEE International Conference on Image Processing ICIP 05, Italy, (2005) 998–1001
9. Honda, H., Haseyama, M., Kitajima, H.: Fractal Interpolation for Natural Images, Proceedings of the IEEE International Conference on Image Processing ICIP 99, Japan, (1999) 657–661
10. Stepin M.: hq3x Magnification Filter, http://www.hiend3d.com/hq3x.html (2003)
11. Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-Based Super-Resolution, IEEE Computer Graphics and Applications, Vol. 22, No. 2, (2002) 56–65

12. Power Retouche: Anti-aliasaing Filter, http://www.powerretouche.com/Antialias_plugin_introduction.htm (2001-2006)
13. Sangwine, S. J., Horne, R.E.N.: The Colour Image Processing Handbook, Chapman & Hall, London (1998)
14. Sharma, G.: Digital Color Imaging Handbook, CRC Press, Boca Raton (2003)
15. Heijmans, H. J.A.M., Ronse, C.: The Algebraic Basis of Mathematical Morphology, Part1: Dilations and Erosions, Computer Vision, Graphics and Image Processing, Vol. 50, (1990) 245–295
16. Ronse, C., Heijmans, H. J.A.M.: The Algebraic Basis of Mathematical Morphology, Part2: Openings and Closings, Computer Vision, Graphics and Image Processing, Vol. 54, (1991) 74–97
17. Heijmans, H. J.A.M.: Morphological Image Operators, Advances in Electronics and Electron Physics, Academic Press, Inc., London (1994)
18. De Witte, V., Schulte, S., Nachtegael, M., Van der Weken, D., Kerre, E.E.: Vector Morphological Operators for Colour Images, Proceedings of the Second International Conference on Image Analysis and Recognition ICIAR 2005, Canada, (2005) 667–675
19. Ledda, A., Luong, H.Q., Philips W., De Witte, V., Kerre, E.E.: Image Interpolation Using Mathematical Morphology, Accepted for the Second International Conference on Document Image Analysis for Libraries DIAL 06, France, (2006)