Image Interpolation using an Adaptive Invertible Approach

Olivier Crave(1), Gemma Piella(2) and Béatrice Pesquet-Popescu(1)
1 GET-ENST, Signal and Image Proc. Dept., Paris, France
2 University of Pompeu Fabra, Barcelona, Spain

Abstract

We present a new image interpolation algorithm based on the adaptive update lifting scheme described in [1]. This scheme allows us to build adaptive wavelets that take into account the characteristics of the underlying signal. Inspired by this technique, we propose an image resizing scheme which has the ability to adapt itself to discontinuities like edges and assures a perfect reconstruction going from low to high resolution and then back to low resolution. Such a feature is highly desirable, for example, for forth and back conversion between the two existing High Definition Television formats, in order to preserve the integrity of the original image in a chain of successive transformations.

Objectives

Our goal was to implement an algorithm capable of performing changes in resolution by a fractional factor $3/2$. It should have the following properties:

- ✔ The interpolated image has to be subjectively pleasant to the viewer: preserve as much as possible the image quality without introducing blocking artifacts and excessive smoothing
- ✔ No loss of information: when going from the 1280×720 format to the 1920×1080 one, and then going back to 1280×720, the original image is retrieved.

Original image

Algorithm

Original image: x^0
Interpolated image: x^1

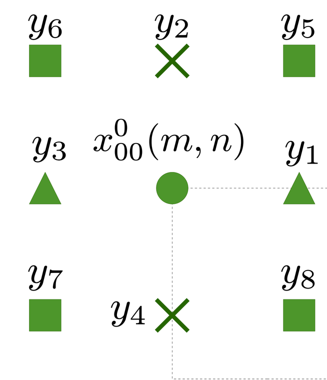
The images are decomposed into their four and nine polyphase components respectively.

The algorithm works as a sequence of 5 steps.

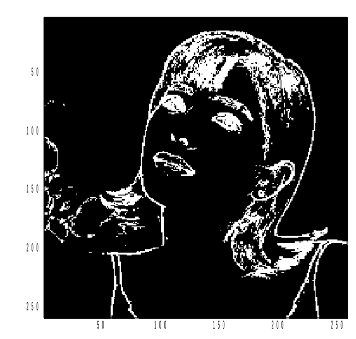
Step 1

First, compute x^1_{00} by updating x^0_{00} in an adaptive way (see [1] for more details):

- ➢ Choose the update filter depending on a local gradient.
- ➢ Consider a 3×3 neighborhood around sample $x = x^0_{00}(m,n)$, x^1_{00} is obtained as $x' = x^1_{00}(m,n) = \alpha_d x + \sum_{j=1}^{8} \beta_{j,d} y_j$ where α_d and $\beta_{j,d}$ are some predefined weights such that $\alpha_d + \sum_{j=1}^{8} \beta_{j,d} = 1$ and $d \in \{0,1\}$, the decision parameter is obtained by thresholding the chosen seminorm of a local gradient vector $\boldsymbol{v} \in \mathbb{R}^8$ with components $v_j = x - y_j,\ j \in \{1,\ldots,8\}$. In this way, only homogenous regions are smoothed while edges are preserved.



Neigbourhood of x



Decision map

We know from [1] the necessary and sufficient conditions which guarantee to recover d from x^1_{00} and $x^0_{01}, x^0_{10}, x^0_{11}$, and hence invert step 1 of the interpolation algorithm:
$x = x^0_{00}(m,n) = \frac{1}{\alpha_d}\left(x' - \sum_{j=1}^{8} \beta_{j,d} y_j\right)$.

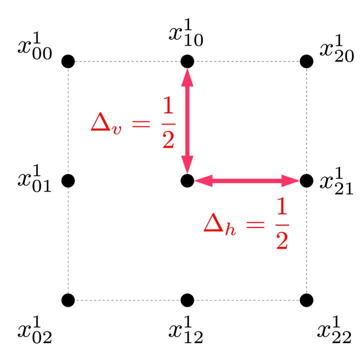
Step 2

From x^1_{00}, compute a gradient function g^0_a:
$g^0_a = (1 - \Delta G_x) \cdot (1 - \Delta G_y)$
where ΔG_x, ΔG_y are the normalized gradients of x^1_{00} in the horizontal and vertical directions, respectively.
Interpolate by 2 this gradient function using a bilinear method into a new gradient function g^1_a.

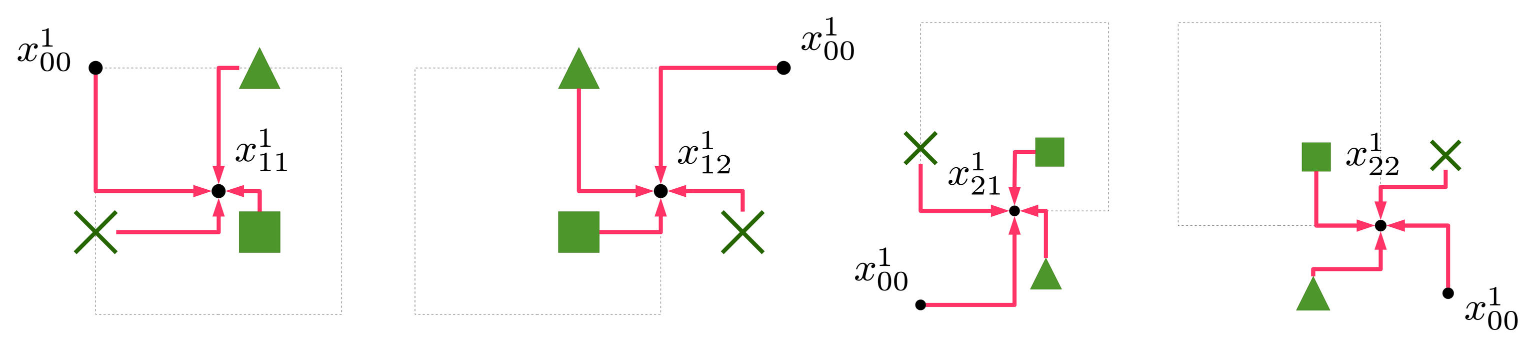
Compute a gradient function g^1_a identical to the one computed in step 2 of the interpolation algorithm.

Step 3

Next, compute $x^1_{11}(m,n) = w_{a0}x^1_{00}(m,n) + w_{a1}x^0_{01}(m,n) + w_{a2}x^0_{11}(m,n) + w_{a3}x^0_{10}(m,n)$
The weights w_{aj} have two components: one related to g^1_a and the other to the spatial ``distance'' $r_j = (1-\Delta_{h_j}) \cdot (1-\Delta_{v_j})$ where Δ_{h_j} denotes the normalized horizontal distance and Δ_{v_j} the vertical one between the given sample and $x^1_{11}(m,n)$, respectively. Then, $w_{aj} = r_j \cdot g^1_a$.
Hence, $w_{a0} = \frac{1}{4} g_{a,00}\ ,\ w_{a1} = \frac{3}{8} g_{a,01}\ ,\ w_{a2} = \frac{9}{16} g_{a,11}\ ,\ w_{a3} = \frac{3}{8} g_{a,10}$.
By symmetry, we obtain x^1_{12}, x^1_{21}, x^1_{22}.



Distance computation



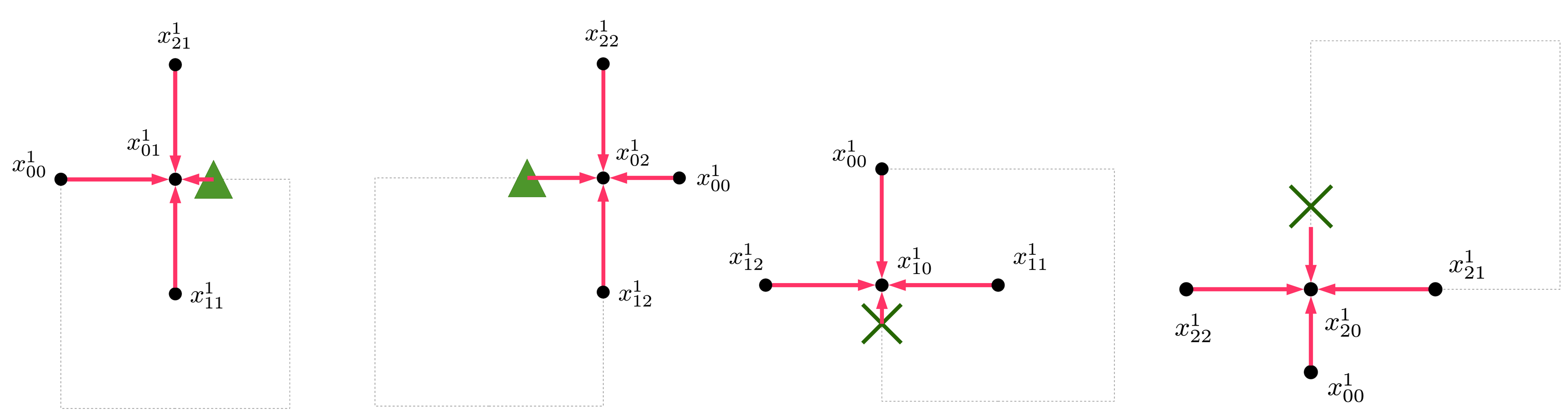
Step 4

From x^1_{00}, x^1_{11}, x^1_{12}, x^1_{21} and x^1_{22} compute a new gradient function g^0_b.
Upsampled it by $3/2$ using a bilinear interpolation to obtain g^1_b.

Compute the gradient functions g^0_b, g^1_b as it is described above.

Step 5

Finally, calculate $x^1_{01}(m,n) = w_{b0}x^1_{00}(m,n) + w_{b1}x^0_{01}(m,n) + w_{b2}x^1_{2,1}(m-1,n) + w_{b3}x^1_{11}(m,n)$
This time, the $r_j, j = 0,\ldots,3$ components are $\frac{1}{2}, \frac{3}{4}, \frac{1}{2}, \frac{1}{2}$. Thus,
$w_{b0} = \frac{1}{2} g^1_{b,00}\ ,\ w_{b1} = \frac{3}{4} g^0_{b,01}\ ,\ w_{b2} = \frac{1}{2} g^1_{b,21}\ ,\ w_{b3} = \frac{1}{2} g^1_{b,11}$
By symmetry, we obtain the rest of the polyphase components, x^1_{02}, x^1_{10} and x^1_{20}.



Interpolated image

Conclusion

The proposed algorithm adaptively updates one polyphase component of the original image and then computes the rest of the components of the output image by means of a gradient-driven interpolation.

The major characteristics of this method are its invertibility and its adaptivity to the image content. The experiments show that the proposed method is comparable, in quality, to bilinear interpolation.

Future work

We want to improve the algorithm such that the interpolated images are of the same quality as the ones one would obtain using bicubic interpolation and still assure a perfect reconstruction of the original image.
Other improvements are planned like being able to use any rational factor, or adapting the algorithm to deal with sequences of images.

Reference

[1] H.J.A.M. Heijmans, B. Pesquet-Popescu and G. Piella, ``Building nonredundant adaptive wavelets by update lifting,'' *Applied and Computational Harmonic Analysis*, no. 18, pp. 252--281, May 2005.