High-Order Discretization of Backward Anisotropic Diffusion and Application to Image Processing

Lorella Fatone
Dipartimento di Matematica, Università di Camerino,
Via Madonna delle Carceri 9, 62032 Camerino (MC), Italy, lorella.fatone@unicam.it

Daniele Funaro
Dipartimento di Scienze Chimiche e Geologiche, Università di Modena e Reggio Emilia,
Via Campi 103, 60121 Modena (MO), Italy, daniele.funaro@unimore.it

Abstract

Anisotropic diffusion is a well recognized tool in digital image processing, including edge detection and denoising. We present here a particular nonlinear time-dependent operator together with an appropriate high-order discretization for the space variable. The iterative procedure emphasizes the contour lines encircling the objects, paving the way to accurate reconstructions at a very low cost. One of the main features of such an approach is the possibility of relying on a rather large set of invariant discontinuous images, whose edges can be determined without introducing any approximation.

1 Introduction

The literature on digital image processing is rather rich. For this reason, we restrict our reference list by only mentioning a few publications: [6], [7], [12], [13], [15]. For at least three decades, partial differential equations (PDEs) have been contributing to the progress in image processing. Since the seminal works of Mumford-Shah [10] and Perona-Malik [11], variational and PDE-based methods have found a wide range of applications. Some works in these directions, particularly related to anisotropic diffusion using PDEs, are for instance: [1], [2], [3],

[5], [9], [14]. Here, images are analyzed in a continuous context through appropriate nonlinear time-dependent equations, suitably discretized. The equations are dissipative in nature, and the nonlinear coefficients act with different strengths, depending on the intensity of the local gradient. In this framework, for example, edge detection algorithms tend to subdivide, during time evolution, the image into uniform regions, with the aim of providing successively a reasonable recovery of the contour lines of the objects involved. We use a similar approach in this paper, realizing some meaningful improvements with respect to the more classical versions. First of all, the discretization of the PDE will be of a higher-order. This property is achieved by using a larger stencil and, in addition to the standard grid-points, an intermediate grid (the so-called "staggered grid") built on half-fractional indices. Such a scheme can be adopted in several other situations, not necessarily related to the specific cases studied here.

The second change we propose concerns the nonlinear diffusion coefficient, which is built in order to ensure a large space of "invariant images". These are images presenting sharp discontinuities that are not modified by the action of the operator. The algorithm of edge detection in these situations does not introduce any approximation, so that the contours remain perfectly represented. Unfortunately, the differential equation resulting from this modification is rather dissipative in presence of sharp variations (with the exception of the above mentioned invariant images). This is in contrast with the usual approach and makes the algorithm non competitive when iterating many times with a time-step chosen according to the standard arrow of time. Things are instead getting better if we move backwards (negative time). Reverse nonlinear diffusion is in general an unstable process. This is however true for the continuous problem. In truth, we will implement the discrete scheme for just one or a few steps, so that the method is not going to be a source of instabilities. In this way, with a proper triggering of the parameters, the variations tend to become sharper, helping the process of edge determination. Some type of noise (such as "salt and pepper") is also greatly amplified, so that it can be easily detected and successively filtered in an appropriate fashion. For the sake of brevity, this last aspect will not be considered in the present paper.

The applications presented here are very preliminary. Improvements can be certainly examined, in conjunction with a series of other specialized filtering procedures. Actually, a single step of the method is very low-cost and can provide interesting information about the image under study. For example, this prediction can be used as a starting guess in view of implementing other, more sophisticated, techniques, as those already available in the market.

2 Introducing the algorithm in 1D

Let N denote the number of pixels along a given interval. Most of the times, one has $N = 256$. Their distance will be denoted by h. In practice, given x_0, we will deal with the points:

$$x_{i+1} = x_i + h, \qquad i = 0, 1, \ldots N, \tag{1}$$

where we added two extra pixels at the beginning and at the end. We will also consider the $N + 3$ shifted points, obtained from the expression:

$$x_{-1/2} = x_0 - \frac{h}{2}, \qquad x_{i+1/2} = x_i + \frac{h}{2}, \qquad i = 0, 1, \ldots N + 1. \tag{2}$$

We start from our image, which is represented by the values $u_i, i = 1, \ldots N$, belonging to a palette with tones of grey from 1 to K. In typical applications we have $K = 256$. The minimum grey level is 1 (black) and the maximum level $K = 256$ (white). We extend the image outside its limits by defining $u_0 = u_1$ and $u_{N+1} = u_N$. This will correspond later to what we call Neumann boundary conditions.

We first build the (first-order) variations:

$$d_{i+3/2} = \frac{1}{h}\left(u_{i+2} - u_{i+1}\right) \approx \frac{du}{dx}(x_{i+3/2}), \tag{3}$$

where u is some differentiable function assuming the values u_i at the points x_i. Successively, we introduce an approximation of the second derivative operator:

$$\begin{aligned} s_{i+3/2} &= \frac{1}{h}\left(\frac{u_{i+3} - u_{i+1}}{2h} - \frac{u_{i+2} - u_i}{2h}\right) = \frac{1}{2h^2}\left(u_i - u_{i+1} - u_{i+2} + u_{i+3}\right) \\ &= \frac{1}{2h}\left(d_{i+5/2} - d_{i+1/2}\right) \approx \frac{d^2u}{dx^2}(x_{i+3/2}), \qquad i = 0, \ldots, N-2. \end{aligned} \tag{4}$$

It is possible to show that the above formula is exact when u is a polynomial of degree less than or equal to three.

We then introduce the quantities:

$$R_{i+3/2} = \sqrt{\frac{(d_{i+3/2})^2}{1 + (d_{i+3/2})^2}}\; s_{i+3/2},, \qquad i = 0, \ldots, N-2, \tag{5}$$

where we also define:

$$R_{-1/2} = R_{5/2}, \quad R_{1/2} = R_{3/2}, \quad R_{N+1/2} = R_{N-1/2}, \quad R_{N+3/2} = R_{N-3/2}. \tag{6}$$

The next step is to interpolate the above values from the fractional to the integer grid. To this purpose we adopt the formula:

$$R_i \approx -\frac{1}{16}R_{i-3/2} + \frac{9}{16}R_{i-1/2} + \frac{9}{16}R_{i+1/2} - \frac{1}{16}R_{i+3/2}, \tag{7}$$

based on a Lagrange type interpolation. This is also exact up to polynomials of degree three.

The final step is to update the vector u_i, $i = 1, \dots N$. This is done as follows:

$$u_i \;\leftarrow\; u_i + \gamma R_i, \qquad i = 1, \dots N, \tag{8}$$

where $\gamma \in \mathbf{R}$ is a suitable parameter to be determined later. Again, we can prolong the set of values by defining $u_0 = u_1$ and $u_{N+1} = u_N$. In this way, if necessary, we can restart the cycle from formula (3) and get a successive update of the values u_i, $i = 1, \dots N$. In the next section we explore the reasons of the above construction.

3 Preliminary considerations in 1D

There are strong connections with partial differential equations. Indeed, let $u = u(t, x)$ be the solution of the following time-dependent nonlinear problem defined in a finite interval:

$$\frac{\partial u}{\partial t} = \sqrt{\frac{d^2}{1 + d^2}}\, \frac{\partial^2 u}{\partial x^2}, \tag{9}$$

where $d = \partial u / \partial x$. The problem is supplied with initial conditions and boundary constraints of Neumann type at the extremes of the interval.

The construction proposed in the previous section corresponds to a finite-differences approximation of the above time-dependent nonlinear problem, with respect to the space variable. This is followed by a first-order time discretization using the explicit Euler scheme. One step of the algorithm is actually represented by the passage in (8). Note that the parameter γ in (8) may assume positive or negative values. This means that time can flow in both directions, so that our differential problem in not necessarily dissipative in nature. We are not interested to study the behavior of (9) in time, and, in the applications that follow, we will implement (8) just for one or a few steps.

Contrarily to classical methods (see, for example, section 5), the anisotropic coefficient $\sqrt{d^2/(1 + d^2)}$ in (9) is less pronounced when the derivatives are small, therefore the solution tends to smooth out the edges during the time evolution.

For this reason, we will prefer to work with negative values of γ. Running backwards brings notoriously to ill-posed problem, although the question can be handled with the due hypotheses (see, e.g., [8]). We will use, however, very few steps of the algorithm and the parameters will be calibrated in order to stay away from impracticable situations.

We would like to mention some features of the algorithm in order to justify the choices we made. Suppose that between the given points x_i and x_{i+1} of a given 1D image we have a jump. With this we mean that $u_j = V_1$ for $j \leq i$, and $u_j = V_2$ for $j \geq i+1$, where $V_1, V_2 \in \mathbb{R}$, with $V_1 \neq V_2$. Then, after a cycle of (8), these values remain unchanged. This is true because the variations in (3) are all zero with the exception of $d_{i+1/2}$. On the other hand, it easy to check that $s_{i+1/2} = 0$ (see (4)). Since, for any k, the values $R_{k+1/2}$ in (5), (6) contain the products $d_{k+1/2}s_{k+1/2}$, they turn out to be identically zero. Thus, the algorithm exactly preserves isolated discontinuities, and therefore does not introduce any dissipation, whatever is γ. It is possible to check that this property holds true also for distributions where $u_j = V_1$ for odd indices and $u_j = V_2$ for even indices. There are, in fact, plenty of basic configurations that are preserved by the algorithm.

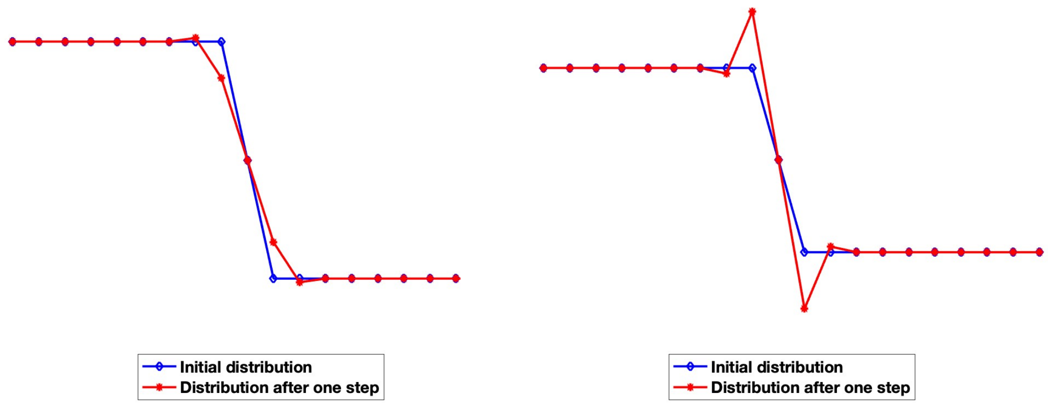

Figure 1: Transformation of the distribution u corresponding to (10) after one step of the algorithm (8) (left: $\gamma > 0$, right: $\gamma < 0$).

The next interesting case to be studied is the one where we consider the following 1D image:

$$u_j = \begin{cases} V_1, \quad j \leq i-1, \\ \frac{1}{2}(V_1 + V_2), \quad j = i, \\ V_2, \quad j \geq i+1, \end{cases} \tag{10}$$

where $V_1, V_2 \in \mathbb{R}$, with $V_1 \neq V_2$. In Fig. 1 we show a couple of cases corresponding to $V_1 > V_2$ in (10). When γ in (8) is positive (plot on the left), after one step some dissipation is introduced, so that the corresponding picture is smoothed. When instead γ is negative (plot on the right) the jump is emphasized. Of course, the parameter γ has to be chosen in an appropriate range (also depending on the values V_1 and V_2). If γ is too small, nothing interesting happens. Otherwise, if it is too large, the final picture may come out to be heavily deteriorated. One may also consider to apply a certain number of steps of this process. The user may then decide the reliability of the outcome depending on the requested result. Different values of γ may be also implemented at each step.

4 The algorithm in 2D

In 2D the natural extension of (9) reads as follows:

$$\frac{\partial u}{\partial t} = \sqrt{\frac{d^2}{1+d^2}}\ \Delta u, \qquad (11)$$

where $d = |\nabla u|$, with ∇u, Δu being the gradient and the Laplace operator, respectively. Such an equation is defined on a square and Neumann boundary conditions are imposed. This formulation allows us to build the algorithm for the two dimensional case, as described here below.

We start by introducing the points:

$$x_{i+1} = x_i + h, \qquad y_{j+1} = y_j + h, \qquad\qquad i, j = 0, 1, \dots N. \qquad (12)$$

The values of grey attained at these points are denoted by $u_{i,j}$. Usually the minimum grey level is 1 and the maximum grey level is 256. In this construction we have to assume that the initial picture (only defined for $i, j = 1, \dots N$) has been bordered in order to satisfy a discrete version of Neumann boundary conditions. Note that the values $u_{0,0}$, $u_{N+1,0}$, $u_{0,N+1}$, $u_{N+1,N+1}$, are irrelevant, since they will not be used in the algorithm.

Intermediate grids, where the indices take semi-integer values, are naturally associated to the primary grid as in (2), that is:

$$x_{-1/2} = x_0 - \frac{h}{2}, \quad y_{-1/2} = y_0 - \frac{h}{2}, \quad x_{i+1/2} = x_i + \frac{h}{2}, \quad y_{j+1/2} = y_j + \frac{h}{2},$$
$$i, j = 0, 1, \dots N+1. \qquad (13)$$

The situation is summarized in Fig. 2.

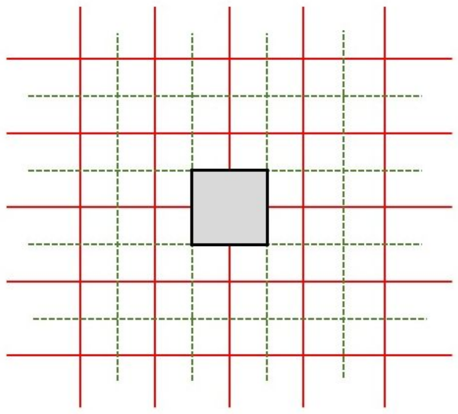

Figure 2: A generic pixel, denoted by the square, is centered at a grid point corresponding to integer indices (at the crossing of the solid lines). Another grid, with fractional indices, is used to carry out the computations.

The next step is to define the values at the shifted nodes by averaging the values at the integer nodes as follows:

$$u_{i+3/2,j+1} = \frac{1}{2}\left(u_{i+1,j+1} + u_{i+2,j+1}\right), \tag{14}$$

$$u_{i+1,j+3/2} = \frac{1}{2}\left(u_{i+1,j+1} + u_{i+1,j+2}\right), \tag{15}$$

where, for the sake of simplicity, we omitted to specify the range of variability of the indices. Successively we build the variations in both directions:

$$d^x_{i+3/2,j+3/2} = \frac{1}{h}\left(u_{i+2,j+3/2} - u_{i+1,j+3/2}\right), \tag{16}$$

$$d^y_{i+3/2,j+3/2} = \frac{1}{h}\left(u_{i+3/2,j+2} - u_{i+3/2,j+1}\right), \tag{17}$$

so that the approximation of the Laplace operator ends up to be:

$$\begin{aligned} l_{i+3/2,j+3/2} &= \frac{1}{2h}\left(d^x_{i+5/2,j+3/2} - d^x_{i+1/2,j+3/2} + d^y_{i+3/2,j+5/2} - d^y_{i+3/2,j+1/2}\right) \\ &\approx (\Delta u)(x_{i+3/2,j+3/2}). \end{aligned} \tag{18}$$

We then introduce the quantities:

$$D_{i+3/2,j+3/2} = \sqrt{\left(d^x_{i+3/2,j+3/2}\right)^2 + \left(d^y_{i+3/2,j+3/2}\right)^2}, \tag{19}$$

and

$$R_{i+3/2,j+3/2} = \sqrt{\frac{\left(D_{i+3/2,j+3/2}\right)^2}{1+\left(D_{i+3/2,j+3/2}\right)^2}}\, l_{i+3/2,j+3/2}. \tag{20}$$

We also extend the above values by introducing a double layer in proximity of the boundary, in order to enforce Neumann type boundary conditions, as done in (6).

As in the one-dimensional case, the successive step is to interpolate the above values from the fractional to the integer grid. We do this by implementing the Cartesian product of formula (7), so obtaining:

$$\begin{aligned} R_{i,j} \;\approx\; & \frac{1}{256}\big(R_{i-3/2,j-3/2} - 9R_{i-3/2,j-1/2} - 9R_{i-3/2,j+1/2} + R_{i-3/2,j+3/2} \\ & -9R_{i-1/2,j-3/2} + 81R_{i-1/2,j-1/2} + 81R_{i-1/2,j+1/2} - 9R_{i-1/2,j+3/2} \\ & -9R_{i+1/2,j-3/2} + 81R_{i+1/2,j-1/2} + 81R_{i+1/2,j+1/2} - 9R_{i+1/2,j+3/2} \\ & R_{i+3/2,j-3/2} - 9R_{i+3/2,j-1/2} - 9R_{i+3/2,j+1/2} + R_{i+3/2,j+3/2}\big)\,. \end{aligned} \tag{21}$$

In order to avoid misunderstandings, let us observe that the number 256 at the denominator in (21) represents the algebraic sum of the integer weights at the numerator, and has nothing to do with the scale of greys of the color palette.

Finally, an update of the values $u_{i,j}$ is realized by setting:

$$u_{i,j} \;\leftarrow\; u_{i,j} + \gamma\, R_{i,j}, \qquad i,j = 1,\ldots N, \tag{22}$$

where again $\gamma \in \mathbf{R}$ is a suitable parameter that can be freely chosen depending on the kind of performances requested. If another step is required, the procedure restarts from the definitions in (14), (15), (16), (17). If necessary, different values of γ may be proposed at each step.

In Fig. 3 we display for $N = 10$ some black&white images that are left unchanged by the algorithm, independently from the value of γ in (22). This means that all the values $R_{i,j}$ are zero. Even if there is the presence of the Laplace operator, the scheme does not introduce any diffusion in these cases.

Before facing more complex situations (see section 6), a simple preliminary test has been conducted on the image of Fig. 4 (left), where we set $N = 20$. The pixels corresponding to $i < j$ are white, whereas the remaining ones are black. This displacement is not an invariant of the algorithm. Hence, some values $R_{i,j}$ turn out to be different from zero. After one step of (22) we are in a situation similar to that of Fig. 1. For choices of γ not excessively large, we get a smoothing

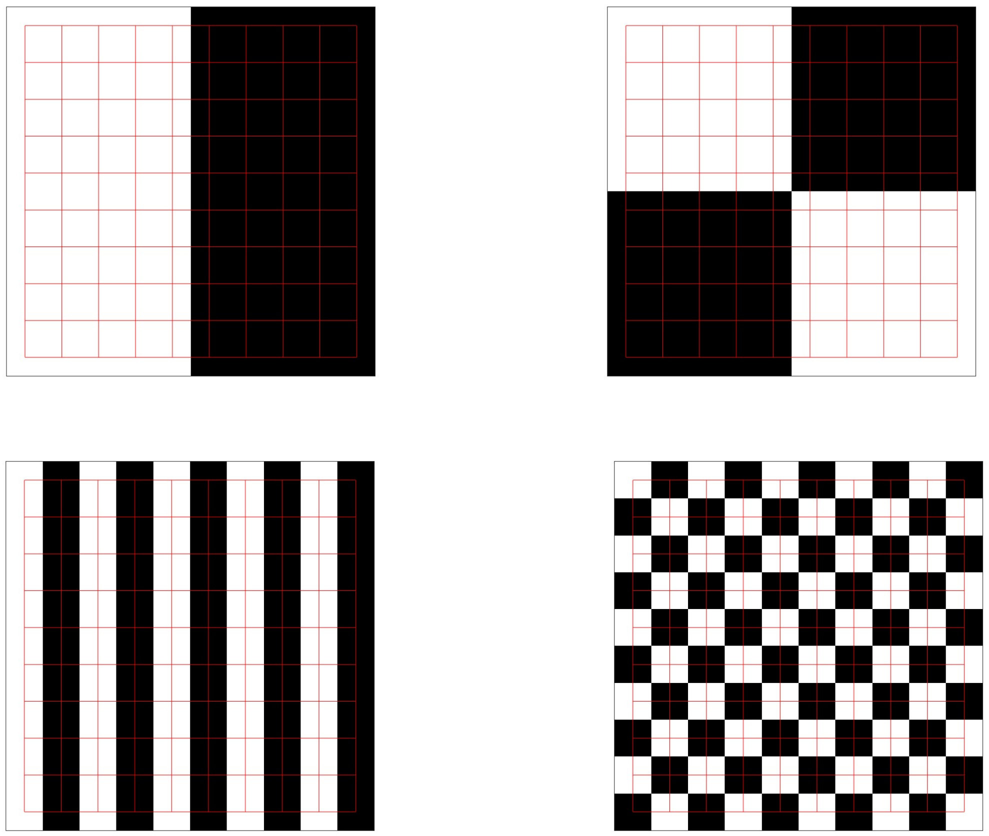

Figure 3: Although these images present sharp discontinuities, they remain unchanged after an application of the nonlinear differential operator resulting from the discretization of the spatial part in (11).

of the picture, if the parameter γ is positive (center image in Fig. 4). Indeed, with minor oscillations, the shades change from black to white through a series of intermediate greys. If, instead, γ is negative (image on the right in Fig. 4), we get a kind of overshooting in proximity of the discontinuity, that amplifies the width of the edge. In these plots, the grey palette has been re-scaled in such a way the global minimum is still black and the maximum white. We also note some discrepancies at the four corners. These are due to the use of formula (21), which requires a certain number of extra values outside the image domain in order to compute the values $R_{1,1}$, $R_{1,N}$, $R_{N,1}$, $R_{N,N}$. This says that Neumann type conditions are not always the optimal choice. We do not have, however, other alternatives to suggest at the moment.

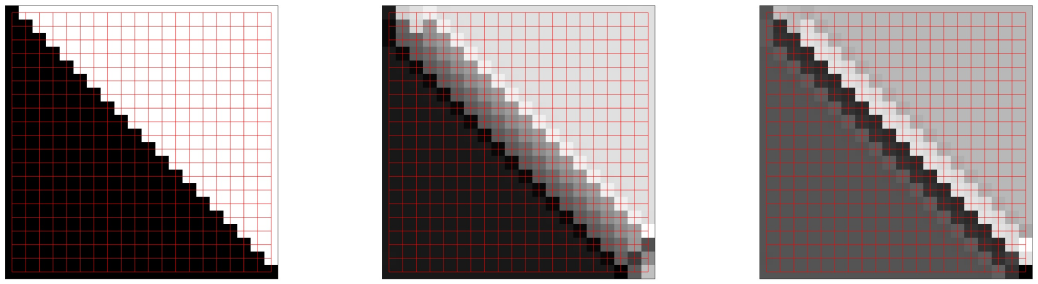

Figure 4: Transformation of the image on the left by applying one step of the algorithm (center: $\gamma > 0$, right: $\gamma < 0$).

5 A classical scheme for anisotropic diffusion

The algorithm discussed so far belongs to the category of anisotropic diffusion schemes. These techniques are based on the solution of a nonlinear time-dependent PDE, presenting a dissipative behavior. The dissipation coefficient generally depends on the magnitude of the local gradient. As an example, we just introduce here the classical scheme proposed by Perona and Malik in their pioneering work [11], which is based on the following nonlinear evolution equation:

$$\frac{\partial u}{\partial t} = \mathrm{div}\Big(g_a(\|\nabla u\|)\ \nabla u \Big), \tag{23}$$

where, for a given parameter a, the function g_a is chosen as follows:

$$g_a(\eta) = \left(1 + \frac{\eta^2}{a^2}\right)^{-1}, \qquad \eta \geq 0. \tag{24}$$

In this way, the diffusion coefficient is small when $\|\nabla u\|$ is large (in particular, in proximity of the edges), and vice versa. Let us observe that (11) is not expressed in divergence form, though this variant could be in principle examined.

We assume that $u(t, x, y)$ in (23) is defined on a square where homogeneous Neumann type conditions are enforced at the boundary. The time t belongs to the interval $[0, T]$, for some $T > 0$. The initial configuration is related to the image under study. After having introduced a time-step $\Delta t > 0$, similarly to what has been done in the previous sections, we denote by $u^k_{i,j}$ the pixel representation of our image at the grid points (x_i, y_j) at some given time $t^k = k\Delta t$, $k = 0, \ldots, K$, with $K\Delta t = T$. The values are updated according to a classical five-points stencil associated with the explicit finite difference discretization of equation (23), i.e.:

$$\frac{u_{i,j}^{k+1}-u_{i,j}^{k}}{\Delta t} = \frac{1}{h}\left[G_{i+1/2,j}^{k}\left(\frac{u_{i+1,j}^{k}-u_{i,j}^{k}}{h}\right) - G_{i-1/2,j}^{k}\left(\frac{u_{i,j}^{k}-u_{i-1,j}^{k}}{h}\right) + \right.$$
$$\left. G_{i,j+1/2}^{k}\left(\frac{u_{i,j+1}^{k}-u_{i,j}^{k}}{h}\right) - G_{i,j-1/2}^{k}\left(\frac{u_{i,j}^{k}-u_{i,j-1}^{k}}{h}\right)\right], \qquad (25)$$

where:

$$\begin{aligned} G_{i,j}^{k} &= g_a\left(\sqrt{\left(\frac{u_{i+1,j}^{k}-u_{i-1,j}^{k}}{2h}\right)^2+\left(\frac{u_{i,j+1}^{k}-u_{i,j-1}^{k}}{2h}\right)^2}\right) \\ &\approx g_a\left(\left\|\nabla u(t^k,x_i,y_j)\right\|\right), \end{aligned} \qquad (26)$$

and, for instance:

$$G_{i+1/2,j}^{k} = \frac{1}{2}\left(G_{i,j}^{k}+G_{i+1,j}^{k}\right), \qquad (27)$$

with similar definitions adopted for the other terms in (25).

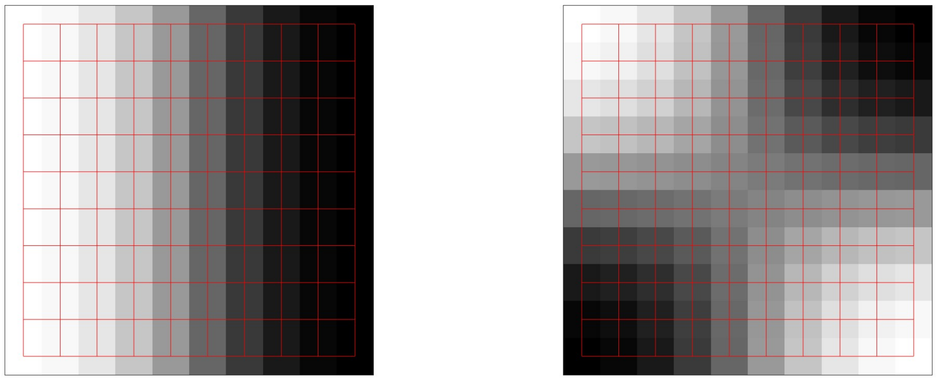

Figure 5: Deformation of the images on top of Fig. 3, after the application of the scheme (25) when $T = 2$, $\Delta t = 0.2$ and $a = 5$.

For the treatment of the boundary points, we take into account a discrete version of Neumann boundary conditions. Since the approximation method is explicit in time, suitable restrictions on Δt should be imposed for stability. For the details we address to [11]. This last issue is not relevant to the discussion of our scheme (actually, the parameter γ in (22) can be either positive or negative), since the aim is to implement the method just for a few iterations.

There are clear differences with respect to the scheme proposed in this paper. In particular, the second order operator in (23) is straightly approximated on the grid points with integer indices. This realizes an approximation error only of the second order in the space variable. As we said in section 3, the anisotropy coefficient of the Laplace operator acts differently in our case, suggesting a reverse of the arrow of time. Finally, the Perona-Malik scheme does not guarantee the preservation of the invariant situations examined in the previous section. For instance, in Fig. 5, we show what happens by applying (25) to the images on top of Fig. 3.

6 Tests on complex images

A typical application in image processing is edge detection, where the techniques already available are numerous. Remaining in the context of anisotropic diffusion, after applying our procedure, the resulting image may be treated by selecting the pixels that tend to diverge from the average. In practice, the new image is renormalized between 1 and 256 and the following filter is applied:

$$\mathcal{F}_\tau(\phi) = \begin{cases} 256, & \phi \geq \tau \quad \text{or} \quad \phi \leq 256 - \tau, \\ 1, & \text{otherwise}, \end{cases} \tag{28}$$

where τ is a suitable integer between 128 and 256.

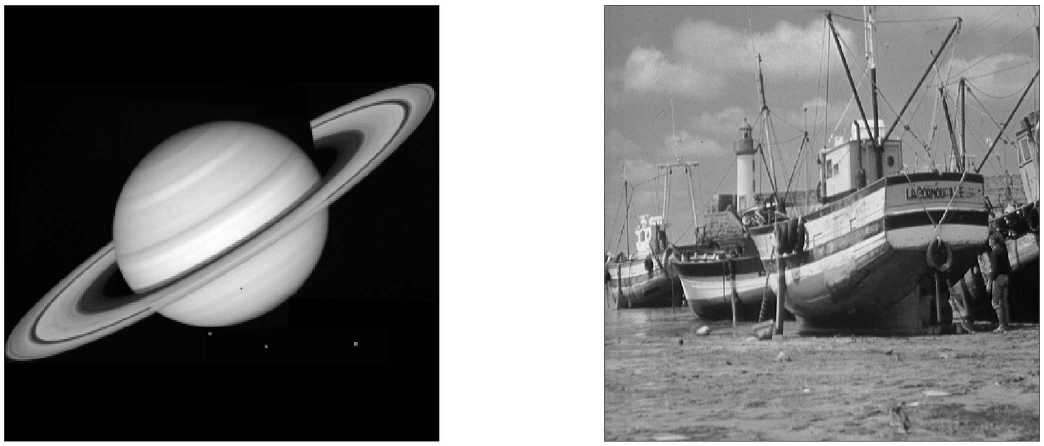

Figure 6: The two images used for the tests.

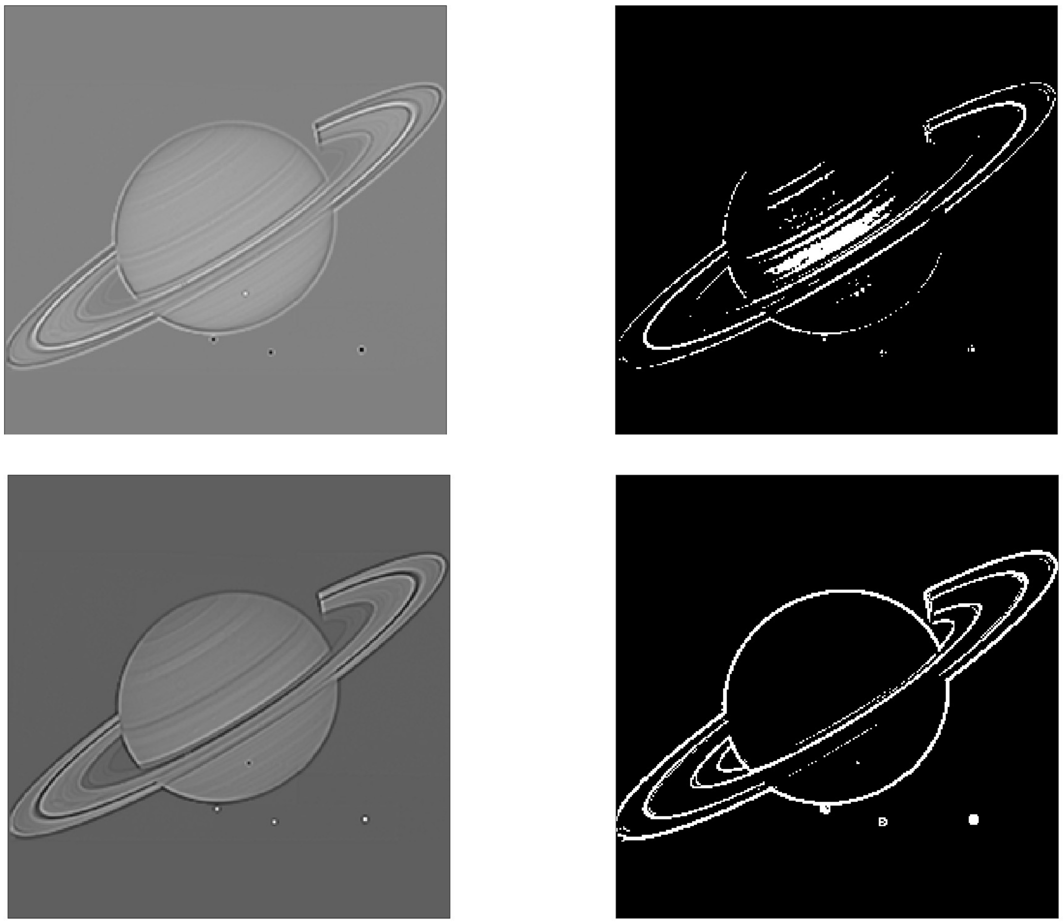

Figure 7: The Saturn image has been treated with one step of the algorithm (22) ($\gamma = 8$ top, $\gamma = -8$ bottom) and a passage of the cut off procedure (28) with $\tau = 162$.

We carried out some tests on the two images shown in Fig. 6. We mainly worked on edge detection, although this is only one of the possible applications. After running a single step of the algorithm, we get the outcomes displayed in Figs. 7, 8, 9. Different values of γ (positive and negative) in the scheme (22), and τ in the filtering procedure (28), are chosen for comparisons. The results are not bad at all, especially if we consider that they have been obtained after just one step of the iterative procedure (22), thus with a cost proportional to the number of pixels. We point out that similar algorithms (such as the Perona-Malik, described in the previous section) require instead a large number of iterations. As expected, the use of negative values of γ (i.e., by going backward in time) provides sharper details. Of course, the goodness of the performances should be judged depending on the type of target one has in mind. We also tried some experiments (not shown here) by iterating (22) more times. The results are not so neat, however, since the images start becoming too blurred.

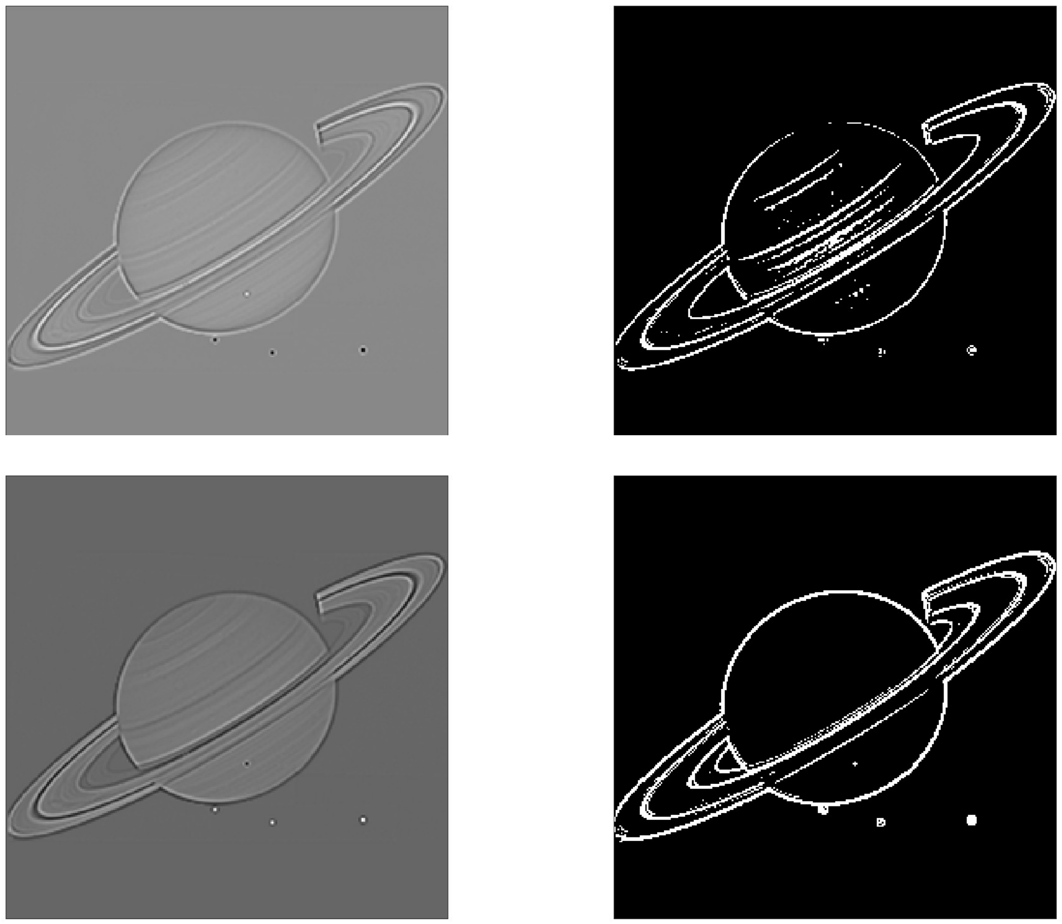

Figure 8: The Saturn image has been treated with one step of the algorithm (22) ($\gamma = 15$ top, $\gamma = -15$ bottom) and a passage of the cut off procedure (28) with $\tau = 155$.

Finally, we conclude this section by showing in Fig. 10 the results obtainable with a standard algorithm, such as the Canny edge detector [4]. In particular the images in Fig. 10 are obtained by using the Matlab command `edge`. By default, `edge` uses the Sobel detection method. We refer to the MathWorks documentation for further insights.

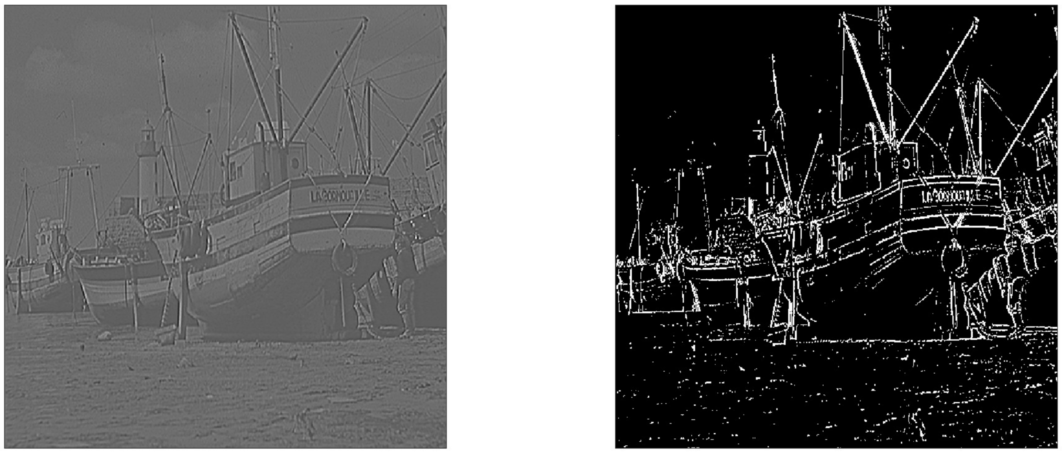

Figure 9: The image of the boat has been treated with one step of the algorithm (22) ($\gamma = -8$) and a passage of the cut off procedure (28) with $\tau = 165$.

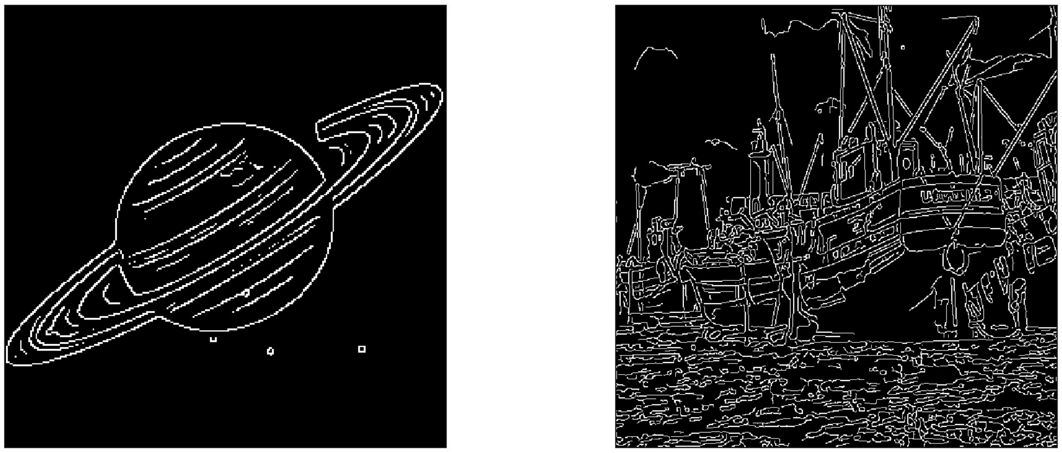

Figure 10: The results of the Canny algorithm applied to the two images used for the tests.

7 Conclusions

A suitable diffusive PDE with anisotropic coefficient has been proposed, together with a high-order space discretization making use of staggered grid points. Some exploratory tests have been carried out in the framework of digital image processing. A single step of the algorithm is already sufficient to emphasize the contour lines of the objects. Thus, the procedure can successfully find application in edge detection or denoising techniques, also in conjunction with other well-known methods.

Acknowledgments

Both authors are members of GNCS-INDAM (Gruppo Nazionale Calcolo Scientifico - Istituto Nazionale di Alta Matematica), Rome.

References

[1] Alvarez, L., Lions, P.L., Morel, J.M.: Image selective smoothing and edge detection by nonlinear diffusion (ii), SIAM Journal on Numerical Analysis, **29**, 845-866 (1992).

[2] Alvarez, L., Guichard, F., Lions, P.L., Morel, J.M.: Axioms and fundamental equations of image processing, Archives for Rational Mechanics and Analysis, **123**, 199-257 (1993).

[3] Buades, A., Coll B., Morel, J.M.: A review of image denoising algorithms, with a new one, Multiscale Modeling and Simulations, **4**(2), 490-530 (2005).

[4] Canny, J.: A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, **PAMI-8**(6), 679-698 (1986).

[5] Felsberg, M.: On the relation between anisotropic diffusion and iterated adaptive filtering. In: Rigoll, G. (eds) Pattern Recognition. DAGM 2008. Lecture Notes in Computer Science, vol 5096. Springer, Berlin, Heidelberg (2008).

[6] Gonzalez, R.C., Woods R.E.: Digital Image Processing, 4th Edition, Pearson, New York (2018).

[7] Granlund, G.H., Knutsson, H.: Signal Processing for Computer Vision. Kluwer Academic Publishers, Dordrecht (1995).

[8] Hummel, R.A., Kimia, B., Zucker, S.W.: Deblurring Gaussian blur, Computer Vision, Graphics, and Image Processing, **38**(1), 66-80 (1987).

[9] Lindenbaum, M., Fischer, M., Bruckstein, A.M.: On gabor contribution to image enhancement, Pattern Recognition, **27**, 1-8 (1994).

[10] Mumford, D., Shah, J.: Optimal approximation by piecewise smooth functions and associated variational problems, Communications on Pure and Applied Mathematics, **42**, 577-685 (1989).

[11] Perona, P., Malik, J.: Scale space and edge detection using anisotropic diffusion, IEEE Transactions on Pattern Analysis and Machine Intelligence **12**(7), 629-639 (1990).

[12] Sapiro, G.: Geometric Partial Differential Equations and Image Analysis, Cambridge University Press (2001).

[13] Solomon, C., Breckon, T.: Fundamentals of Digital Image Processing: A Practical Approach With Examples in Matlab, John Wiley & Sons Inc. (2010).

[14] Weickert, J.: Theoretical foundations of anisotropic diffusion in image processing, Computing, Suppl. **11**, 221-236 (1996).

[15] Weicker, J.: Anisotropic Diffusion in Image Processing, ECMI Series, Teubner-Verlag, Stuttgart, Germany (1998).