Image structure preserving denoising using generalized fractional time integrals.

Edurado Cuesta, Mokhtar Kirane, Salman Amin Malik

Noname manuscript No.
(will be inserted by the editor)

Anisotropic like approach to image denoising by means of generalized fractional time integrals

Eduardo Cuesta · Mokhtar Kirane · Salman A. Malik

Received: date / Accepted: date

Abstract A generalization of the linear fractional integral equation $u(t) = u_0 + \partial^{-\alpha} Au(t)$, $1 < \alpha < 2$, which is written as a Volterra matrix–valued equation when applied as a pixel–by–pixel technique, has been proposed for image denoising (restoration, smoothing,...). Since the fractional integral equation interpolates a linear parabolic equation and a hyperbolic equation, the solution enjoys intermediate properties. The Volterra equation we propose is well–posed, and allows us to handle the diffusion by means of some *viscosity parameters* instead of introducing non linearities in the equation as in the Perona–Malik and alike approaches. Several experiments showing the improvements achieved by our approach are provided.

Keywords Image processing · Fractional integrals and derivatives · Volterra equations · Convolution quadrature methods.

Mathematics Subject Classification (2000) 44A35 · 44K05 · 45D05 · 65R20 · 68U10 · 94A08

Eduardo Cuesta
Department of Applied Mathematics, Industrial Engineering School (Francisco Mendizabal), University of Valladolid, Spain.
Tel: +34-983-423000 Ext. 6805
Fax: +34-983-423490
E-mail: eduardo@mat.uva.es

Mokhtar Kirane · Salman A. Malik
Laboratoire de Mathématiques Image et Applications, Université de La Rochelle, Avenue M. Crépeau, 17042 La Rochelle Cedex, France.

Mokhtar Kirane
E-mail: mokhtar.kirane@univ-lr.fr

Salman A. Malik
E-mail: salman.malik@univ-lr.fr

1 Introduction

Partial differential equations based methods for image processing (filtering, denoising, restorations, segmentation, edge enhancement/detection,...) have been largely studied in the literature (see [39] and references therein).

In that framework the first, and most investigated equation is might be the (parabolic) linear heat equation

$$\begin{cases} \partial_t u(t,\mathbf{x}) = \Delta u(t,\mathbf{x}), & (t,\mathbf{x}) \in [0,T] \times \Omega, \\ u(0,\mathbf{x}) = u_0(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \dfrac{\partial u}{\partial \eta}(t,\mathbf{x}) = 0, & (t,\mathbf{x}) \in [0,T] \times \partial\Omega, \end{cases} \tag{1}$$

where ∂_t, and Δ stand for the time derivative, and two–dimensional Laplacian, respectively, $\Omega \subset \mathbb{R}^2$ is typically a square domain, $\partial\Omega$ represents the boundary of Ω, $\partial/\partial\eta$ stands for the outward normal derivative, and u_0 the original image. Let us notice that $u(t,\mathbf{x})$ stands for the restored image at time level t, i.e. the original image $u_0(\mathbf{x})$ evolved in time.

The interest for this model comes out due to the fact that the solution of (1) can be written as a convolution

$$u(t,\mathbf{x}) = \int_{\mathbb{R}^2} G_{\sqrt{2t}}(\mathbf{x}-\mathbf{y}) u_0(\mathbf{y})\, \mathrm{d}\mathbf{y},$$

where G is the two–dimensional Gaussian kernel

$$G_\sigma(\mathbf{x}) := \frac{1}{2\pi\sigma^2} e^{-|\mathbf{x}|^2/2\sigma^2}.$$

Since convolution with a positive kernel is the basic tool in linear filtering, computing the solution of (1) is equivalent to Gaussian filtering in a classical way.

However, in this equation the diffusion is isotropic which, in the context of image processing, means that smoothing applies uniformly in the whole image, therefore independently of the image itself. This yields that

in most of cases edges and corners are severely blurred disabling this filter for practical applications.

In view of this, an anisotropic model seems to be a suitable approach to guarantee a preserving–edges regularization. This approach was initially proposed by Perona and Malik in [34]; it reads

$$\begin{cases} \partial_t u(t,\mathbf{x}) = \operatorname{div}\left(c(|\nabla u(t,\mathbf{x})|^2)\nabla u(t,\mathbf{x})\right), (t,\mathbf{x}) \in Q, \\ u(0,\mathbf{x}) = u_0(\mathbf{x}), \qquad \mathbf{x} \in \Omega, \\ \dfrac{\partial u}{\partial \eta}(t,\mathbf{x}) = 0, \qquad \partial\Omega, \end{cases} \tag{2}$$

for $(t,\mathbf{x}) \in Q = [0,T] \times \Omega$. The diffusion coefficient $c : [0,+\infty) \to [0,+\infty)$, is chosen to be close to zero near edges and corners, that is, pixels where gradient is large. On the contrary, c should be large in pixels with low gradient variation. Functions satisfying this assumptions are commonly called *edge stopping functions*. Unfortunately, *edge stopping functions* lead to backward–forward problems that are ill–posed. Typical examples of *edge stopping functions* are

$$c(s) = 1/(1+s) \qquad \text{or} \qquad c(s) = e^{-s}, \tag{3}$$

used firstly by Perona and Malik, and later by many others authors. Despite of the ill–posedness, numerical experiments with these models carried out by some authors show that no significant instabilities are observed; moreover, for large final times, images yielded seem to preserve and enhance edges, even by explicitly assuming the ill–posedness of the problem (see e.g. [21]). The reason for that, as reported by H. Amman [1], is that the numerical scheme used by Perona and Malik does not correspond to their equation but rather to a time–regularized one which is well–posed this time. In the same way, some other approaches have been proposed as for example the ones based on the total variation of suitable functional (see [38]).

These results have promoted the idea of replacing (2) by nearby equations keeping on the one hand the same practical and numerical properties, and on the other hand, lying in a *reasonable* functional space setting where the well–posedness can be guaranteed as well as the bounded variation, and further analytical and numerical properties. The first perturbed model was proposed in [7] where, for a suitable extension of u over $\mathbb{R}^2$ (e.g. by 0) denoted $\tilde{u}$, $c(|\nabla u|)$ is replaced by $c(|\nabla(G_\sigma * \tilde{u})|)$ (G_σ defined above). In that case, for $u_0 \in L_2(\Omega)$, the regularized problem admits a unique solution in $C([0,T],L_2) \cap L_2((0,T),H^1)$. Despite of some features of this model, images become uniformly grey (for grey–scale images) in the long run, thus the information gets lost (see [1]). Variants of this approach have been studied by many authors (see e.g. [22,23] and references therein).

Lately, further approaches have been proposed, e.g. by adding regularizing terms, like $\epsilon\Delta\partial_t u$, to the diffusion equation in (2) (see [4]). Let us also mention that practical experiments have been carried out by means of stable numerical discretizations applied to anisotropic diffusion equations of type (2) into spheres $\mathbf{S}^2$ (see [3]).

Moreover, higher order partial differential equations based regularizations are also used (see [17]), and in particular four order partial differential equations (see [25,40]). However, despite of the practical results seem to be quite good in most of cases, some of them have not been closely studied yet, both from the analytical and the numerical point of view.

In our work, we present a new approach based on fractional calculus which allows us to handle the diffusion, i.e. the smoothing in the image terminology, by means of a parameter which plays the role of "viscosity" parameter in a linear partial differential equation. The well–posedness is now guaranteed, and since the final objective is the practical implementation, very efficient numerical discretizations have been closely studied by many authors, and therefore they are at our disposal for the experiments we show in Section 6.

This paper is organized as follows. In section 2, we recall some facts concerning fractional calculus, and the first approaches to image processing by using fractional calculus. Section 3 focusses on generalized fractional integrals based approach to image processing which is the main novelty of this work. Numerical discretizations are presented in Section 4. The discussion on the implementation, and practical experiments, are shown in Sections 5 and 6 respectively.

2 Fractional calculus

Image filtering by means of fractional calculus is first considered in [9]. In that work, a generalization of the heat equation (1) is proposed; the equation in [9] can be written as

$$\begin{cases} \partial_t^\alpha u(t,\mathbf{x}) = \Delta u(t,\mathbf{x}), & (t,\mathbf{x}) \in [0,T] \times \Omega \\ u(0,\mathbf{x}) = u_0(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \dfrac{\partial u}{\partial \eta}(t,\mathbf{x}) = 0, & (t,\mathbf{x}) \in [0,T] \times \partial\Omega, \end{cases} \tag{4}$$

where ∂_t^α stands for the fractional time derivative of order $1 < \alpha < 2$ in the sense of Riemann–Liouville. Integrating in both sides, the problem (4) can be expressed as

$$u(t,\mathbf{x}) = u_0(\mathbf{x}) + \partial^{-\alpha}\Delta u(t,\mathbf{x}), \tag{5}$$

also with homogeneous Neumann boundary condition, and where $\partial^{-\beta}$, for $\beta > 0$, stands for the fractional integral of order $\beta \in \mathbb{R}^+$, in the sense of Riemann–Liouville.

Let us recall that, for $g : [0, +\infty) \to \mathbb{R}$, $g \in AC[0, +\infty)$ the integral of order $\beta \in \mathbb{R}^+$ in the sense of Riemann–Liouville is defined as the convolution integral

$$\partial^{-\beta} g(t) := \int_0^t k_\beta(t-s) g(s) \mathrm{d}s, \qquad t \geq 0, \tag{6}$$

where $k_\beta(t) := t^{\beta-1}/\Gamma(\beta)$, for $t > 0$ (see [24]). Now, the definition of the fractional derivative of order $\beta \geq 0$ is

$$\partial^{\beta} g(t) := \frac{d^m}{dt^m} \partial^{\beta-m} g(t), \qquad t \geq 0,$$

where $m \in \mathbb{N}$, $m - 1 < \beta \leq m$.

The interest of our model in the framework of image processing is due to the fact that, for $1 < \alpha < 2$, the problem (4) interpolates the linear (parabolic) heat equation (1) corresponding to $\alpha = 1$, and the linear (hyperbolic) wave equation

$$\begin{cases} \partial_t^2 u(t,\mathbf{x}) = \Delta u(t,\mathbf{x}), & (t,\mathbf{x}) \in [0,T] \times \Omega \\ u(0,\mathbf{x}) = u_0(\mathbf{x}), & \mathbf{x} \in \Omega, \\ \partial_t u(0,\mathbf{x}) = 0, & \mathbf{x} \in \Omega, \\ \dfrac{\partial u}{\partial \eta}(t,\mathbf{x}) = 0, & (t,\mathbf{x}) \in [0,T] \times \partial\Omega, \end{cases} \tag{7}$$

corresponding to $\alpha = 2$ (with zero initial velocity). Therefore, some properties of the solution of (4) are intermediate between the ones of (1) and (7) (see e.g. [19,20]). In particular, it can be easily proven that for the scalar equation (i.e. by replacing Δ in (4) with a complex λ with non–positive real part) the solution decays as $1/(1 + t^\alpha |\mathrm{Re}(\lambda)|)$, i.e. the solution is $o(t^{-\alpha})$, as $t \to +\infty$. In that case, the solution decays slower than for the scalar heat equation (with the same λ) for which the solution decays exponentially, and faster than for the scalar wave equation whose solution does not decays (but oscillates). The diffusion is now handled by the parameter α. We show below that the solution of the model we propose, is well behaved.

At the same time, fractional calculus was proposed for edge detection in [32], and later for image denoising in [2]. In these papers, the authors proposed anisotropic equations, where the anisotropy is handled by means of spatial fractional derivatives; however, the papers do neither include the study of the well–posedness of the problem nor the study of the behavior of the fully numerical discretization.

Finally, let us mention [12–16] where the fractional calculus, also applied to image processing, is understood as the fractional powers of the two–dimensional Laplacian, i.e. $(-\Delta)^\beta$, for $\beta > 0$.

3 Volterra equations

Despite the fact that the approach (5) in Section 2 seems to be very promising, the diffusion (smoothing) is still uniform all over the whole image as in (1).

The main contribution of this paper comes out of a refinement of such approach which consists in splitting the whole image into sub–images, and apply (4) with different values of α for each sub–image (see [8]). Roughly speaking, the choice of each α is carried out by setting values close to 1 for the sub–images with lower mean gradient variation, and close to 2 for the sub–images with higher mean gradient variation. This refinement provided very good practical results e.g. when applied in some satellite image classifications (see [37]).

This idea suggested us a finer approach which is intended to be the limit of the above situation, i.e. the application of the fractional equation (4) with a different value of the derivative order α for each single pixel. The values of α are chosen according to the gradient variation at each single pixel as we discuss in Section 5.

Hereafter, we will consider gray–scale images since for colored images processing becomes a more difficult task. Might be one can perform a similar methodology based on the one we present here but separately for each of the three layers (one per color) e.g. in the case of RGB format.

Let us start by taking the spatial discretization of the Laplacian in (4) based on a central difference scheme with mesh length $h > 0$. In such a way, Δ transforms into a $M^2 \times M^2$ five–diagonals matrix Δ_h (see Fig. 1), and in the same fashion, $u(t,\mathbf{x})$ is transformed into a $M^2 \times 1$ vector–valued function $\mathbf{u}(t)$ which stands stand for the vector–arranged image pixels at time level t.

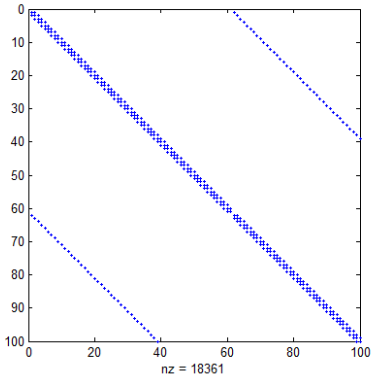

Fig. 1 Sparsity pattern of the discretized Laplacian

Actually, since at present the most of signals (image, sound,...) are handled in digital format, this approach becomes natural.

As we commented above, the novelty of our approach consists in replacing the (only one) order derivative α of equation (4) with a different value of α for each single pixel of the image. This approach reads now as the linear Volterra equation

$$\mathbf{u}(t) = \mathbf{u}_0 + \int_0^t \mathbf{K}(t-s)\mathbf{u}(s)\mathrm{d}s, \qquad 0 \le t \le T, \qquad (8)$$

where $\mathbf{u}_0$ is the vector–arranged initial data (original image), and the convolution kernel $\mathbf{K}$ is defined as

$$\mathbf{K}(t) = I(t) \cdot \Delta_h$$

with

$$I(t) = \begin{bmatrix} \dfrac{t^{\alpha_1}}{\Gamma(\alpha_1+1)} & 0 & \dots & 0 \\ 0 & \dfrac{t^{\alpha_2}}{\Gamma(\alpha_2+1)} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dfrac{t^{\alpha_{M^2}}}{\Gamma(\alpha_{M^2}+1)} \end{bmatrix}$$

and, $0 < \alpha_j < 1$, for $j = 1, 2, \dots, M^2$.

Let us notice that $\mathbf{K}$ is a five–diagonals matrix valued function, and since the Laplace transform of $\mathbf{K}$ exists, the well–posedness of (8) is then guaranteed (see e.g. [36]).

4 Time discretizations

4.1 Background

Time discretizations of Volterra equations as (8) have been largely studied in literature; let us mention, e.g., the convolution quadrature based methods (see [28–30]). In particular Runge–Kutta convolution quadrature methods (the convolution quadrature is based on classical Runge–Kutta methods) of that equations provide high order numerical methods jointly with good stability properties. In [31], these discretizations have been studied in the abstract setting of sectorial operators, i.e. for convolution kernels whose Laplace transform is of *sectorial type*. Let us recall that a complex function G is of *sectorial type* if there exist $0 < \theta < \pi/2$, $c \in \mathbb{R}$, and $\mu, M > 0$ such that G is analytic in the sector

$$S_\theta := \{\lambda \in \mathbb{C} : |\arg(\lambda - c)| < \pi - \theta\},$$

and

$$|G(\lambda)| \le \frac{M}{|\lambda|^\mu}, \qquad \lambda \in S_\theta.$$

Under these assumptions, the inverse Laplace transform can be written by means of the Bromwich formula as

$$g(t) = \frac{1}{2\pi i}\int_\gamma e^{\lambda t} G(\lambda)\mathrm{d}\lambda,$$

where γ is a complex path connecting $-i\infty$, and $+i\infty$ parallel to the boundary of S_θ with increasing imaginary part.

The convergence of these methods has been recently extended to analytic semigroups (see [6]) where the only one requirement on the kernel is the existence of the Laplace transform (weaker assumption that for sectorial case). Since the Laplace transform of each function $t^{\alpha_j}/\Gamma(\alpha_j+1)$ in (8) exists, and therefore the Laplace transform of $\mathbf{K}$ exists, these methods turn out to be appropriate for our purposes. In this case, if $\tilde{\mathbf{K}}$ denotes the Laplace transform of $\mathbf{K}$, then the inverse Laplace transform can be written as

$$\mathbf{K}(t) = \frac{1}{2\pi i}\int_\gamma e^{\lambda t}\tilde{\mathbf{K}}(\lambda)\mathrm{d}\lambda,$$

where $\gamma(r) = a + ri$, $-\infty < r < +\infty$, for $a \in \mathbb{R}^+$. Let us notice that now the whole path γ lies on the right–hand complex plane $\mathrm{Re}(\lambda) \ge a$.

In order to set a suitable Runge–Kutta convolution quadrature method, we must take into account that the time regularity of the solutions of (8) is constrained by the nature of the convolution kernel, in particular the continuity of their derivatives is guaranteed only up to the first order. Therefore, in this work, we will focus on the backward Euler convolution quadrature method, i.e. avoiding higher order schemes whose convergence will require more regularity on the solutions. This method will be sufficient to show the new features of our approach.

4.2 Convolution quadratures

For the sake of the readers convenience, we first recall the definition of the backward Euler convolution quadratures, and for the sake of the simplicity of the explanation we refer the readers to [6,30] for the definition of Runge–Kutta convolution quadratures in the general case and further references.

Let $\tau > 0$ be the time step of the discretization. The convolution integral in (8) reads

$$\begin{aligned} &\int_0^t \mathbf{K}(t-s)\mathbf{u}(s)\mathrm{d}s \\ &= \int_0^t \frac{1}{2\pi i}\int_\gamma e^{\lambda(t-s)}\tilde{\mathbf{K}}(\lambda)\mathrm{d}\lambda\,\mathbf{u}(s)\mathrm{d}s \\ &= \frac{1}{2\pi i}\int_\gamma \tilde{\mathbf{K}}(\lambda)Y(\lambda,t)\mathrm{d}\lambda, \end{aligned}$$

where $Y(\lambda,t)$ stands for the solution of the ordinary differential equation

$$\mathbf{y}'(t)=\lambda\mathbf{y}(t)+\mathbf{u}(t),\ 0\le t\le T,\quad \text{with}\quad \mathbf{y}(0)=0. \tag{9}$$

The backward Euler convolution quadrature is obtained as

$$\int_0^{t_n}\mathbf{K}(t_n-s)\mathbf{u}(s)\mathrm{d}s\approx\frac{1}{2\pi i}\int_\gamma\tilde{\mathbf{K}}(\lambda)Y_n(\lambda)\mathrm{d}\lambda,$$

where $t_n=n\tau$, and $Y_n(\lambda)$ stands for the approximation of $Y(\lambda,t_n)$ reached by the backward Euler method applied to (9). Therefore, if $\delta(z)/\rho(z)$ stands for the quotient of the backward Euler characteristic polynomials, i.e. $\delta(z)/\rho(z)=z-1$, then

$$\int_0^{t_n}\mathbf{K}(t_n-s)\mathbf{u}(s)\mathrm{d}s\approx\sum_{j=0}^{n}\mathbf{Q}_{n-j}^{(\alpha)}\mathbf{u}(t_j),$$

where the weights $\mathbf{Q}_j^{(\alpha)}$'s turn out to be the coefficients of

$$\tilde{\mathbf{K}}\left(\frac{1-\xi}{\tau}\right)=\sum_{j=0}^{+\infty}\mathbf{Q}_j^{(\alpha)}\xi^j.$$

Here, the quotient of the backward Euler characteristic polynomials is evaluated in $\xi=1/z$, leading to $\delta(\xi)/\rho(\xi)=\delta(\xi)=1-\xi$, and to the definition of weights

$$\mathbf{Q}_j^{(\alpha)}=\tau^\alpha\begin{bmatrix}\binom{\alpha_1}{j} & 0 & \dots & 0\\ 0 & \binom{\alpha_2}{j} & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & \dots & 0 & \binom{\alpha_{M^2}}{j}\end{bmatrix}\cdot\Delta_h, \tag{10}$$

for $j=0,1,2,\dots$, as $M^2\times M^2$ matrices. (see [6,10,30] for more details).

4.3 Numerical method. Convergence

Let $\mathbf{u}_n$ be the approximation of $\mathbf{u}(t_n)$, for $n\ge 0$. Then the time discretization of (8) by means of the backward Euler convolution quadrature method reads

$$\mathbf{u}_n=\mathbf{u}_0+\sum_{j=1}^{n}\mathbf{Q}_{n-j}^{(\alpha)}\mathbf{u}_j,\quad n\ge 1,$$

and keeping in mind the practical implementation, since the matrix Δ_h is not singular, the unique n-th approximation is reached by solving the linear system

$$\left(I-\mathbf{Q}_0^{(\alpha)}\right)\mathbf{u}_n=\mathbf{u}_0+\sum_{j=1}^{n-1}\mathbf{Q}_{n-j}^{(\alpha)}\mathbf{u}_j,\quad n\ge 1. \tag{11}$$

In the abstract setting in [6], optimal error bounds are reached. In particular, for the backward Euler based method, the first order is reached by assuming the existence and boundedness of the second derivative of the solution. However, since the second derivative of the solution of (8) is merely integrable but not continuous, these results cannot be directly applied. In our case, if one takes into account the nature of the convolution kernel, then the stability proven in [10] jointly with Theorem 3.1 in [28] allow us to guarantee that the method is convergent of first order.

Besides, in [6] an interesting result is also proven which becomes even more interesting when applications fit into the framework of image processing. For the readers convenience we recall this result in the case of the backward Euler convolution quadrature method we apply in this paper.

Theorem 1 *If* $\mathbf{u}$ *is the solution of (8), and* $\mathbf{u}_n$*, for* $n\ge 0$*, is the numerical solution yielded by (11), then there exists a probability density* $\rho_{n,\tau}:[0,+\infty)\to\mathbb{R}$ *such that*

$$\mathbf{u}_n=\int_0^{+\infty}\mathbf{u}(s)\rho_{n,\tau}(s)\mathrm{d}s,\qquad n\ge 1. \tag{12}$$

The interest of this theorem is that, since $\rho_{n,\tau}$ is a probability density, i.e. positive and such that

$$\int_0^{+\infty}\rho_{n,\tau}(s)\mathrm{d}s=1,$$

if $\mathbf{u}$ is for example positive, then the representation (12) guarantees the positivity of $\mathbf{u}_n$. In other words, the numerical solution (11) preserves, among other properties, the positivity of the continuous solution.

Moreover, let us point out that in [6] it is proven that the density of probability in (12) does not depend on the equation itself but only on the numerical method. In particular, for this method (see [10,33]), we have an explicit expression for such a density

$$\rho_{n,\tau}(t)=\frac{1}{\tau(n-1)!}\left(\frac{t}{\tau}\right)^{n-1}\mathrm{e}^{-t/\tau},\qquad n\ge 1.$$

Finally, we can mention some other very efficient methods to discretize (8) as for example the ones based on the discretization of the inverse Laplace transform (see [27]), the ones based on the adaptive fast and oblivious convolutions (see [26]), or the collocation methods (see [5] and references therein).

5 Implementation

In this section we will discuss some facts concerning the implementation of (11) itself.

First of all, the choice of α's should be done according to the idea of preserving edges and corners and removing noise. Therefore, in view of the discussion in Section 2, pixels where the gradient is large should be associated with values of α close to 2. On the contrary, pixels with lower gradients should be associated with values of α close to 1. Let us notice that the practical computation of the gradient variation turns out to be very simple in a discrete setting as the one we consider in the spatial variables.

However some remarks have to be taken into account:

- On the one hand, avoiding the singular situations which can be yielded in both–sides values of α, i.e. $\alpha = 1$ and $\alpha = 2$, at least from the numerical point of view, we will not reach these values when setting α's. In particular, in Section 6, for $j = 1, 2, \ldots, M^2$, $\alpha_j \in [1 + \epsilon, 2 - \epsilon]$ with $\epsilon = 10^{-3}$.
- On the other hand, since extreme situations appear such as *isolated* noisy pixels (see Figure 3 where a gray–scale image is shown as a three–dimensional surface), a particular choice of α's is expected; in particular for those pixels, values close to 1 will be associated. On the contrary, near edges and corners (very high gradient variation) no smoothing should be required, therefore α's close to 2 will be set for that pixels.
  According to this criteria, in this work, the setting of α values follows a profile distribution as in Figure 2.

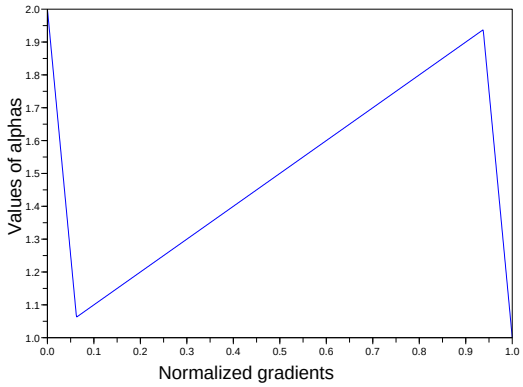

Fig. 2 Profile distribution of α's.

  Notice that this procedure allows one to establish different settings of distributions depending e.g. on the characteristics of each image.
- From a computational point of view, the number of different values of α's should be limited, otherwise if one admits a number of α's as large as the number of pixels, the implementation becomes unavailable in practical cases. In fact, in this work, we set $\alpha \in \{1 + j/N, 1 \leq j \leq N\}$, for a fixed integer N which in Section 6 turns out to be $N = 100$.

Let us also mention that in (10) a fixed number of weights $\mathbf{Q}_j^{(\alpha)}$ are computed once for all for each α_j. Moreover, the practical computation of that weights has been carried out by means of the Fast Fourier Transform as in [6], therefore saving a noticeable *run–time*.

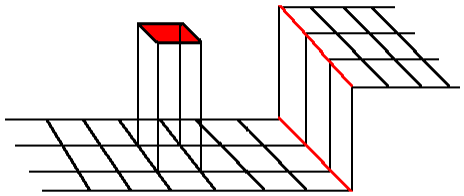

Fig. 3 Three dimensional representation of (expected) isolated noisy pixels for a gray–scale image.

Another fact of interest concerns the measure of goodness of an implementation or procedures in image filtering, restoration, and in general, to measure the quality of a processed image. To this end, in this work, we consider two criteria, SNR and $PSNR$, which have been largely used in literature (see e.g [11,18,35]), and which are commonly applied to determine the quality of a processed image in the sense commented above (filtering, restoration,...). In fact, SNR and $PSNR$ stand for the Signal to Noise Ratio and Peak Signal to Noise Ratio, respectively, and the unit for both of these ratios are decibels (dB). To be more precise, SNR of a restored image R compared to an ideal image I is defined as

$$SNR = 10 \cdot \log_{10}\left(\frac{\mathrm{var}(I)}{\mathrm{var}(I - R)}\right),$$

where $\mathrm{var}(x)$ stands for the variance of the vector x. Here I and R are considered as vector arranged gray–scale images, as in Section 3, with 256 gray levels. In the same way, $PSNR$ is defined as

$$PSNR = 10 \cdot \log_{10}\left(\frac{\sum_{i,j} 255^2}{\sum_{i,j}(I_{i,j} - R_{i,j})^2}\right),$$

where $I_{i,j}$, $R_{i,j}$ are the pixel values of I and R respectively. Notice that in restoration problems, we have a corrupted image, and try to restore the ideal image which is not in general available (in practice), but for the calculation of SNR and $PSNR$ (from above formulas) ideal image is required. For the experiments in Section 6, we take an image (ideal image) and we perturb that image by adding up some noise. This image is then used for restoration; therefore SNR and $PSNR$ can be computed for the restored images.

Finally, let us point out that all the computations have been carried out on an Intel(R)Core(TM)2Duo 2.53GHz, 3Gb RAM desktop PC.

6 Practical results

In this section, we show the improvements in the restorations provided by our approach. To this end, we perform some experiments where a noisy image is evolved by using the heat equation (1) (HE), Perona–Malik model (2) with $c(s) = \mathrm{e}^{-s}$ as commented in (3) (PM), and the model (8) we propose (VE).

Example 1

In this example, a 150×150 size Lena's image is considered (Figure 4.(a)), i.e. on the spatial domain $\Omega = [0, 150] \times [0, 150]$. The original image has been perturbed by additive gaussian noise, and the resulting image (Figure 4.(b)) has values $SNR = 5.7$ and $PSNR = 16$. In Table 1, we show the results yielded by the restoration carried out with the mentioned procedures.

In fact, in view of Table 1, it can be observed that similar results are reached with (PM) and (VE), better anyway than with (HE). However we should recall that (VE) stands for a linear model whose well–posedness is guaranteed on the contrary to happens with (PM).

For further close observation of the procedures applied to Figure 4.(b), a part of such image is considered in all instances; in particular, a small square has been zoomed (see Figures 4.(f)–(j)). On the one hand, the restored image obtained by (HE) (Figure 4.(h)) has severely lost the structure of the original image, and the gray–scale level of the pixels became almost uniform. However, (PM) somehow preserves edges/corners (Figure 4.(i)) but smooths very strongly the *flat* areas which causes a lost of information regarding the texture of the image. The (VE) denoises the image (Figure 4.(j)), also by preserving edges/corners of the image; but here smoothing in flat areas is not so strong as with (PM).

Example 2

In Figure 5.(a) a 256×256 size gray–scale image of a house has been perturbed also by additive gaussian noise (Figure 5.(b)) having $SNR = 5$ and $PSNR = 20$.

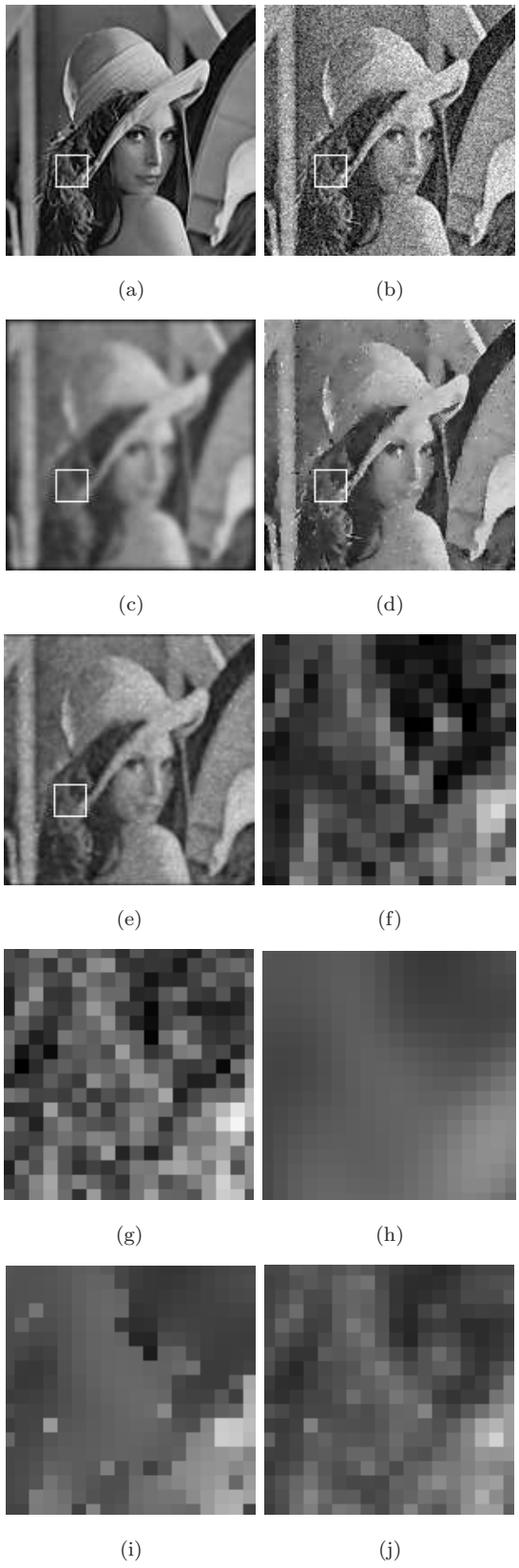

Fig. 4 Denoising of Lena image: (a) Original image, (b) noisy image perturbed by guassian noise, (c) with (HE), (d) with (PM), (e) with (VE), (f) zoomed part of original image, (g) zoomed part of noisy image, (h) zoomed part with (HE), (i) zoomed part with (PM), and (j) zoomed part (VE).

Table 1 First experiment.

	SNR	PSNR	Figure
(HE)	4.6	17	Fig. 4.(c)
(PM)	10.4	18	Fig. 4.(d)
(VE)	9.5	19.3	Fig. 4.(e)

Table 2 Second experiment.

	SNR	PSNR	Figure
(HE)	6.5	17	Fig. 5.(c)
(PM)	12	27	Fig. 5.(d)
(VE)	11	26	Fig. 5.(e)

As in Example 1, the restorations have been applied yielding the results in Table 2 and, as in Example 1, the ratios SNR and $PSNR$ for (PM) and (VE) are quite similar. However, in this example, a remarkable difference can be observed when zooming a small square of the image. In particular, it can be observed that in the zoomed area (Figure 5.(f)), the noisy image (Figure 5.(g)) is severely blurred with (HE) as expected (Figure 5.(h)), while (PM) and (VE) (Figures 5.(i)-(j) respectively) clearly preserve the image structure (i.e. edges/corners). Moreover, a further analysis of Figures 5.(i)-(j) reveals that the structure of the original image keeps clearer with (VE) than with (PM) meaning here that smoothing with (PM) in *flat* areas shows stronger than with (VE). Finally, isolated noisy pixels appear when restoring with (PM) on the contrary to what happens with (VE).

Example 3

Since in Experiments 1 and 2, SNR and $PSNR$, for (PM) and (VE), keep very close one to each other, the efficiency of our approach seems to be based on no more than an optical evidence. However, in this experiment, we will show that the efficiency of our approach is more than optical, and to this end, we will consider an image where the texture plays a crucial role, and where the restoration procedures can be stressed.

The images we consider in these experiments are a 250×250 size image of a wood (Figure 6. (a)), and a 512×512 size naive image (Figure 6. (e)), both strongly perturbed with gaussian noise (ratios $SNR = 0.014$ and $PSNR = 12$, see Figure 6. (b), $SNR = 0.054$ and $PSNR = 16.2$, see Figure 6. (f), respectively). In these kind of images, the texture tuns out to be more important than edges preservation, and among the numerical results in Table 3, a simple overview shows the goodness of our approach vs. (PM).

However, to be more precise in our analysis, Table 3 shows a numerical evidence of the efficiency of our method; in fact it must be highlighted that SNR and

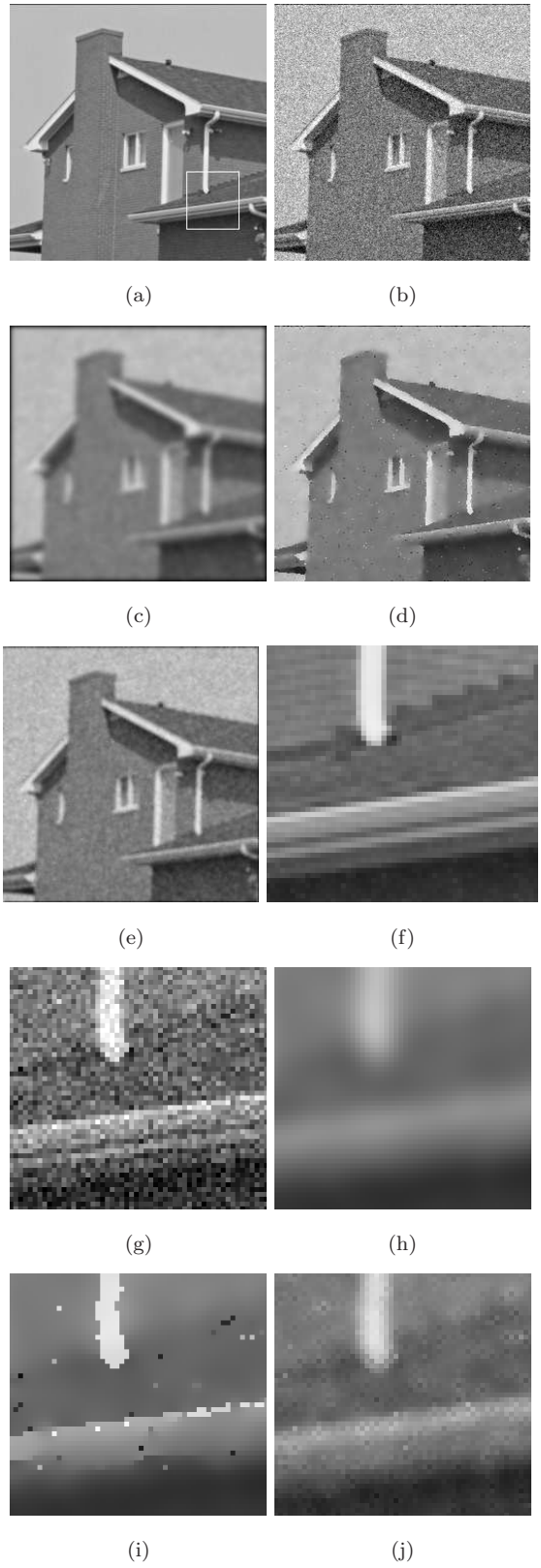

Fig. 5 Denoising of house image: (a) Original image, (b) noisy image perturbed by guassian noise, (c) with (HE), (d) with (PM), (e) with (VE), (f) zoomed part of original image, (g) zoomed part of noisy image, (h) zoomed part with (HE), (i) zoomed part with (PM), (j) zoomed part with (VE).

$PSNR$ are improved by using (VE) in comparison with (PM).

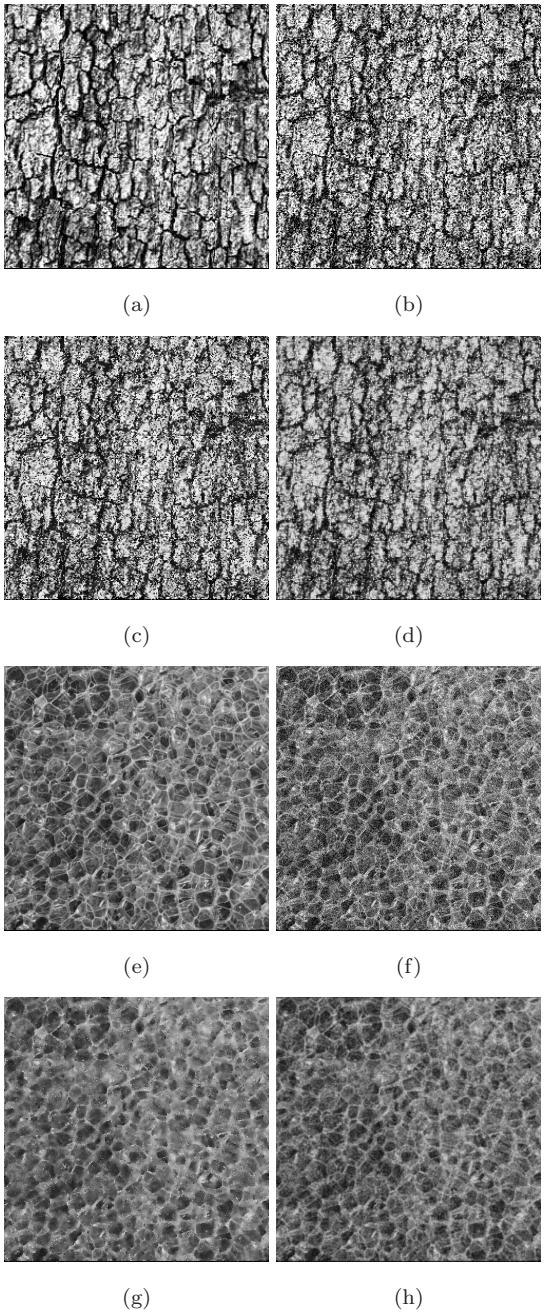

Fig. 6 Denoising of textured images: (a) original image of wood, (b) noisy image perturbed by gaussian noise, (c) (PM), (d) (VE) (e) original naive image, (f) noisy image perturbed by gaussian noise, (g) (PM), (h) (VE).

Table 3 Third experiment.

	SNR	PSNR	Figure
(PM)	1.8	12.1	Fig. 6.(c)
(VE)	4	14.4	Fig. 6.(d)
(PM)	5	21.2	Fig. 6.(g)
(VE)	6.8	22.6	Fig. 6.(h)

7 Conclusions and outlook

In the present work, we propose a partial differential equation based approach to image processing (filtering, denoising, enhancing,...) whose main novelty is that it fits into the framework of fractional calculus (derivatives and integrals) hence Volterra equations.

The interest of our work is twofold: On the one hand, the model we propose allows us to handle the smoothing by means of certain "viscosity" parameters which define the matrix–valued linear Volterra equation we propose. In other words, the smoothing is now handled by means of a linear partial equation, i.e. without introducing tricky nonlinear terms in the equation as many authors propose.

On the other hand, the model we propose fits into a closed mathematical setting, both from the analytical and the numerical point of view. To be more precise, the well–posedeness of the Volterra equation we propose is guaranteed and numerical methods for its discretization have been largely studied in the literature as well.

As an additional interesting property of our proposal is that one can change the *profile* of the filter merely by changing the "viscosity" parameters setting, i.e. the distribution function we use. This allows the user to adapt somehow the filter to each single image according to its characteristics.

Acknowledgements The research of the first author has been supported by DGI–MCYT under project MTM2004–07194 cofinanced by FEDER funds.

References

1. Amann, H.: Time-delayed perona-malik type problems. Proceedings of the International Conference on Differential Equations (Bratislava, Check Republic 25-29 July 2005), K. Mikula, et al. (eds.), Comenius University Press pp. 15–38 (2005)
2. Bai, J., Feng, X.C.: Fractional anisotropic diffusion for image denoising. IEEE Trans. Image Process. **16**, 2492–2502 (2007)
3. Bartels, S., Prohl, A.: Stable discretizations of scalar and contrained vectorial perona-malik equation. Interfaces Free Bound. **9**(4), 431–453 (2007)
4. Bellettini, G., Fusco, G.: A regularized perona-malik functional: some aspects of the gradient dynamics. Proceedings of

the International Conference of Differential Equations (Hasselt, Belgium 22-26 July 2003), F. Dumortier, H. Broer, et al. (eds.), World Scientific pp. 639–644 (2003)
5. Brunner, H.: Collocation Methods for Volterra Integral and Related Functional Equations. Cambridge Monographs Applied and Computational Mathematics. Cambridge University Press (2004)
6. Calvo, M., Cuesta, E., Palencia, C.: Runge-kutta convolution quadrature methods for well-posed equations with memory. Numer. Math. **107**, 589–614 (2007)
7. Catte, F., Lions, P.L., Morel, J.M., Coll, T.: Image selective smoothing and edge detection by nonlinear difussion. SIAM J . Numer. Anal. **29**(1), 182–193 (1992)
8. Cuesta, E.: Some advances on image processing by means of fractional calculus. In: Nonlinear Science and complexity, vol. II, p. (in press). World Scientific (2008)
9. Cuesta, E., Finat, J.: Image processing by means of a linear integro-differential equation. IASTED pp. 438–442 (2003)
10. Cuesta, E., Palencia, C.: A numerical method for an integro-differential equation with memory in banach spaces. SIAM J . Numer. Anal. **41**, 1232–1241 (2003)
11. Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K.: Color image denoising via sparse 3d collaborative filtering with grouping constraint in luminance-chrominance space. In: IEEE INT. CONF. IMAGE PROCESS., ICIP 2007 (2007)
12. Didas, S.: Denoising and enhancement of digital images, variational methods, integrodifferential equations, and wavelets. Ph.D. thesis, Saarland University (2004)
13. Didas, S., Burgeth, B., Imiya, A., Weickert, J.: Regularity and scale–space properties of fractional high order linear filtering. Scale Space and PDE Methods in Computer Vision. Lecture Notes in Computer Vsion **3459**, 13–25 (2005)
14. Didas, S., Steidl, G., Weickert, J.: Discrete multiscale wavelet shrinkage and integrodifferential equations. In: P. Scheelkens, T. Ebrahimi, G. Cristóbal, F.T. (eds.) (eds.) Optical and Digital Image Processing–Photonic Europe. Proceedings of SPIE, vol. 7000, pp. 7000S–1–7000S–12. Bellingham (2008)
15. Didas, S., Steidl, G., Weickert, J.: Integrodifferential equations for continuous multiscale wavelet shrinkage: The discrete case. Technical rerpot 214, Department of Mathematics, Saarland University (2008)
16. Didas, S., Weickert, J.: Integrodifferential equations for continuous multiscale wavelet shrinkage. Inverse Problems and Imaging **1**(1), 47–62 (2007)
17. Didas, S., Weickert, J., Burgeth, B.: Properties of higher order nonlinear diffusion filtering. J. Math. Imaging Vis. **35**, 208–226 (2009)
18. Emmanuel, J.L.S., C, E.J., Donoho, D.L., Wavelet, A., Denoising, I.: The curvelet transform for image denoising. IEEE Transactions on Image Processing **11**, 670–684 (2000)
19. Fujita, Y.: Integro-differential equation which interpolates the heat equation and the wave equation. Osaka J. Math. **27**, 319–327 (1990)
20. Fujita, Y.: Integro-differential equation which interpolates the heat equation and the wave equation (ii). Osaka J. Math. **27**, 797–804 (1990)
21. Hamza, A.B., Krim, H., Unal, G.B.: Unifing probabilistic and variational estimation. IEEE Signal Process. Mag. **2**, 37–47 (2002)
22. Kačur, J., Mikula, K.: Solution of nonlinear diffusion appearing in image smoothing and edge detection. Appl. Numer. Math. **17**(1), 47–59 (1995)
23. Kačur, J., Mikula, K.: Slow and fast diffusion effects in image processing. Computation and Visualization in Science **3**, 185–185 (2001)
24. Kilbas, A.A., Srivastava, H.M., Trujillo, J.J.: Theory and Applications of Fractional Differential Equations. Elsevier (2006)
25. Lisaker, M., Lundervold, A., Tai, X.C.: Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. IEEE Trans. Image Process. **12**(12), 1579–1590 (2003)
26. López-Fernández, M., Lubich, C.: Adaptive, fast,and oblivious convolution in evolution equations with memory. SIAM J . Sci. Comp. **30**, 1015–1037 (2008)
27. López-Fernández, M., Palencia, C.: On the numerical inversion of the laplace transform of certain holomorphic mappings. Appl. Numer. Math. **51**, 289 – 303 (2004)
28. Lubich, C.: Convolution quadrature and discretized operational calculus i. Numer. Math. **52**, 129–145 (1988)
29. Lubich, C.: Convolution quadrature and discretized operational calculus ii. Numer. Math. **52**, 413–425 (1988)
30. Lubich, C.: Convolution quadrature revisited. BIT **44**, 503–514 (2004)
31. Lubich, C., Ostermann, A.: Runge–kutta methods for parabolic equations and convolution quadrature. Math. Comput. **60**, 105–131 (1993)
32. Mathieu, B., Melchior, P., Oustaloup, A., Ceyral, C.: Fractional differentiation for edge detection. Signal Process. **83**, 2421–2432 (2003)
33. Oustaloup, A. (ed.): Backward Euler method as a positivity preserving method for abstract integral equations of convolution type, vol. Proceedings of Fractional Differentiation and its Applications (2006)
34. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Trans. on Pattern Anal. and Mach. Intell. **12**(7), 629–639 (1990)
35. Portilla, J., Simoncelli, E.P.: Image denoising via adjustment of wavelet coefficient magnitude correlation. In: Proceedings of the 7th ICIP, pp. 10–13. IEEE Computer Society (2000)
36. Pruss, J.: Evolutionary Integral Equations and Applications. Birkhäuser Verlag, Basel (1993)
37. Quintano, C., Cuesta, E.: Improving satellite image classification by using fractional type convolution filtering. IEEE Trans. Geosc. Rem. Sens. (submitted)
38. Rudin, L., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithm. Phys. D **60**, 259–268 (1992)
39. Weickert, J.: Anisotropic Diffusion in Image Processing. B.G. Teunbner Stuttgart (1998)
40. You, Y.L.: Fourth-order partial differential equations for noise removal. IEEE Trans. Image Process. **9**(10), 1723–1730 (2000)