Journal of Computational Mathematics, Vol.26, No.6, 2008, 856–875.

MULTIGRID METHOD FOR A MODIFIED CURVATURE DRIVEN DIFFUSION MODEL FOR IMAGE INPAINTING*

Carlos Brito-Loeza and Ke Chen
Department of Mathematical Sciences, University of Liverpool, Peach Street, Liverpool L69 7ZL, United Kingdom
Email: cbrito@liverpool.ac.uk, k.chen@liverpool.ac.uk

Abstract

Digital inpainting is a fundamental problem in image processing and many variational models for this problem have appeared recently in the literature. Among them are the very successfully Total Variation (TV) model [11] designed for local inpainting and its improved version for large scale inpainting: the Curvature-Driven Diffusion (CDD) model [10]. For the above two models, their associated Euler Lagrange equations are highly nonlinear partial differential equations. For the TV model there exists a relatively fast and easy to implement fixed point method, so adapting the multigrid method of [24] to here is immediate. For the CDD model however, so far only the well known but usually very slow explicit time marching method has been reported and we explain why the implementation of a fixed point method for the CDD model is not straightforward. Consequently the multigrid method as in [Savage and Chen, *Int. J. Comput. Math.,* 82 (2005), pp. 1001-1015] will not work here. This fact represents a strong limitation to the range of applications of this model since usually fast solutions are expected. In this paper, we introduce a modification designed to enable a fixed point method to work and to preserve the features of the original CDD model. As a result, a fast and efficient multigrid method is developed for the modified model. Numerical experiments are presented to show the very good performance of the fast algorithm.

Mathematics subject classification: 68U10, 65F10, 65K10.
Key words: Image inpainting, Variational models, Regularization, Multilevel methods.

1. Introduction

Image inpainting has been defined as the process of reconstituting the missing or damaged portions of an image, in order to make it more legible and to restore its unity. The aim of inpainting is then to modify an image in a way that is non-detectable for an observer who does not know the original image [2].

There are a variety of reasons why images can have damaged parts, for instance because of some physical degradation like aging, weather or intentional scratching. Not only that, we also would like to recover parts of objects of an image occluded by other objects or to reconstruct parts that have been missing due to digital communication processes. We can imagine a number of applications of this technique: among the most known are the restoration of old pictures with scratches or missing patches [2], text removal, digital zooming and superresolution [11], error concealment [14], disocclusion in computer vision, X-Ray CT artifacts reduction [18], and the long list continues. Inpainting techniques deal with these kinds of problems trying

* Received December 20, 2007 / Revised version received January 24, 2008 / Accepted April 18, 2008 /

to reconstruct in the best possible way the missing or damaged parts of an image from the available information.

The mathematical interest in this field became increasingly active in the last decade since the very first works on image interpolation by Mumford, Nitzberg and Shiota [22], Masnou and Morel [21] and Caselles, Morel and Sbert [5]. However it was the pioneering work of Bertalmio *et al* [2] who proposed an algorithm to imitate the work of inpainting artists who manually restore old damaged pictures which mainly motivated all the subsequent research in this field [11, 12]. This algorithm cleverly transports a smoothness image measure (namely the Laplacian of the image) along the level lines (contours of the same image intensity) directed into the inpainting domain; in their paper, they also showed that some intermediate steps of anisotropic diffusion are necessary to avoid blurring of edges. This algorithm was created mostly intuitively but later on turned out to be closely related to the Navier-Stokes equation, as showed by Bertozzi *et al*; see [4]. Since then, many other authors have proposed different models for digital inpainting.

Chan and Shen [11] introduced the Total Variation (TV) model for local inpainting based on the celebrated total variation based image denoising model of Rudin, Osher and Fatemi [23]. Later on the same authors modified this model to improve its performance for large scale inpainting, and created the so-called Curvature-Driven Diffusion (CDD) model [10]. Furthermore they, together with Kang, introduced a higher order variational model [9] based on the Euler's elastica which connects the level lines by using Euler elastica curves [19] instead of using straight lines as the TV model does. Unfortunately for the latter two models there appear to exist no fast methods to find the numerical solution. The aim of this paper is to develop a fast multigrid algorithm for the CDD model.

A related inpainting model was proposed by Esedoglu and Shen [16] and is based on the very successfully Mumford-Shah image segmentation model. This model is also good for local inpainting but shares the same problem as the TV model in that it cannot reconnect separated parts of broken objects far apart. To fix this problem, the same authors of [16] proposed the Mumford-Shah-Euler inpainting model which in the same fashion of the Euler's elastica model uses the information encoded in the curvature to reconnect smoothly the level lines. More recently, in separate works, Bertozzi, Esedoglu and Gillete [3] proposed a model to inpaint binary images based on the Cahn-Hilliard equation and Grossauer and Scherzer [17] proposed a model based on the complex Ginzburg-Landau equation. It remains to develop fast multigrid methods for these models.

Each one of the above models has its own particular features which may not suit all applications. However as rightly remarked in [13] one of the most interesting open problems in digital inpainting (whatever the model) is the fast and efficient digital realization. The new multigrid method for the CDD model is our first step in this direction.

We remark that measuring the quality of restoration is non-trivial as physical perceptions can be different and the 'true' solutions may not be unique [12]. In our tests, we have chosen one perception as the true solution.

The rest of this paper is organized as follows. Section 2 introduces the image inpainting problem and two variational models, followed by the review of a commonly-used numerical method in Section 3. Section 4 describes first the modified CDD model and then the framework of a nonlinear multigrid method with emphasis on two smoothers, global and local. A local Fourier analysis is shown to give an indication of the effectiveness of both smoothers. Finally Section 5 presents some testing results illustrating the effectiveness of the modified model and the associated multigrid method.

2. Problem Formulation and Variational Models

Given an image $z = z(x, y)$ defined on a domain $\Omega \in \Re^2$ and one subset $D \subset \Omega$ where the pixel values of z are missing or damaged due to some reason as illustrated in Figure 2.1. The typical inpainting problem is to try to reconstruct the values of z in D from the available information on $\Omega \backslash D$ which may contain noise. The subset D which is known as the inpainting domain may have complicated topology and be not necessarily connected.

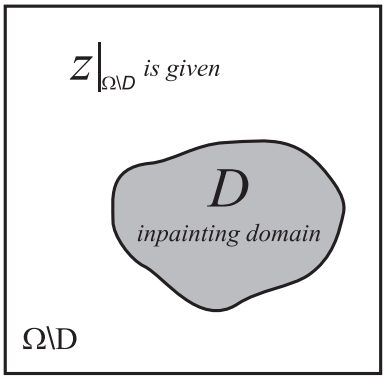

Fig. 2.1. Illustration of a typical inpainting problem

2.1. The total variation model

In this section we review very briefly the TV model [11], which was constructed based on three principles that a good inpainting model must satisfy. These are: Locality meaning that to carry out the inpainting process only the information surrounding the inpainting domain must be used, Restoration of narrow edges meaning that the model must be able to reconstruct the missing parts of the edges which give the most visually information in an image, and Robustness to noise meaning that it must get rid of any noise present and restore the missing part.

Assume that $u = u(x, y)$ and $\eta = \eta(x, y)$ are respectively the true image and the unknown additive Gaussian noise satisfying $z = u + \eta$ in $\Omega \backslash D$. Following Rudin, Osher and Fatemi [23], the TV inpainting model is as follows

$$\min_u \int_\Omega |\nabla u| \; dxdy + \frac{\lambda}{2} \int_{\Omega\backslash D} (u - z)^2 \; dxdy. \tag{2.1}$$

Although direct minimization ideas [6, 7] could be applied, so far the above minimization is mainly solved via its associated Euler-Lagrange equation:

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) + \lambda_E (z - u) = 0 \quad \text{with} \quad \frac{\partial u}{\partial \vec{n}} = 0 \;\text{ in }\; \partial\Omega, \tag{2.2}$$

where $\vec{n}$ is the unit outward normal on the boundary $\partial\Omega$ and λ_E

$$\lambda_E = \begin{cases} \lambda > 0 & (x, y) \in \Omega \backslash D \\ 0 & (x, y) \in D \, . \end{cases} \tag{2.3}$$

In D alone, where $\lambda_E = 0$, equation (2.2) reduces to an ill-conditioned boundary value problem with non unique solution as it was shown in [5].

To illustrate the virtue and limitations of the TV model we shall present two inpainting examples. In the top of Figure 2.2, the inpainting domain D represented by the horizontal noisy bars is relatively small in size compared with the characteristic feature of the image, therefore the TV model performs very well and carries out a good inpainting. In the bottom of Figure 2.2, however, D (represented by the noisy triangle) is relatively large and therefore the TV model gives an unpleasant result.

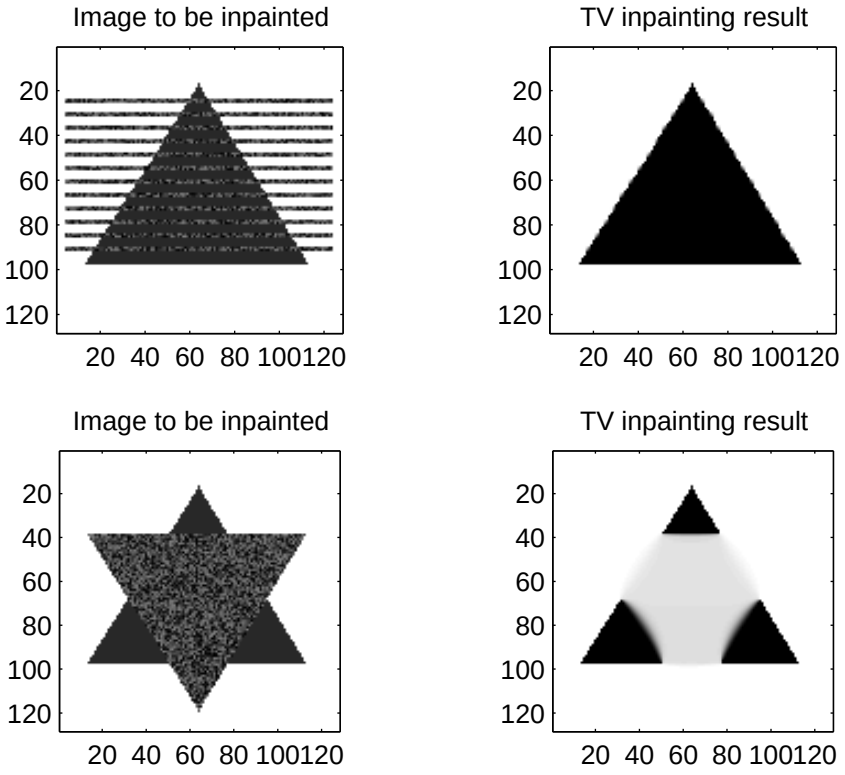

Fig. 2.2. On one hand, at the top, we show a TV inpainting satisfying the connectivity principle [10,11]. On the other hand, at the bottom, one that does not satisfy it

2.2. A curvature-driven diffusion model

The CDD model which we review in this section was designed to correct the inability of the TV model in reconnecting separated parts of broken objects far apart.

Looking for a solution to this problem, Chan and Shen [10] realized that in the TV model the diffusion coefficient given by $\hat{D} = \frac{1}{|\nabla u|}$ only depends on the contrast or strength of the level lines and it does not depend on the geometric information of the level lines themselves. They found that the curvature $\kappa = \kappa(x, y)$ defined by $\kappa = \nabla \cdot \frac{\nabla u}{|\nabla u|}$ could be used to modify the diffusion coefficient $\hat{D}$ by introducing a function $g = g(|\kappa|)$ within it. This way the geometric

information encoded in κ is used to strengthen the diffusion coefficient where necessary. The new diffusion coefficient $\hat{D}$ is then given by

$$\hat{D} = \frac{g(|\kappa|)}{|\nabla u|}, \quad \text{with} \quad g(s) = \begin{cases} 0 & s = 0 \\ \infty & s = \infty \\ > 0, & 0 < s < \infty. \end{cases} \tag{2.4}$$

On one hand, the choice of $g(\infty) = \infty$ was selected to take advantage of those points with very high or infinity curvature and use them to encourage reconnection by increasing $\hat{D}$ as much as possible.

On the other hand, the choice of $g(0) = 0$ is to avoid the CDD model degenerating to the TV model. According to Chan and Shen the choice of $g(0) = a \neq 0$ could endanger the connectivity principle, see [10]. They suggested [10]

$$g(s) = s^p, \quad \text{with} \quad s > 0, \quad p \geq 1. \tag{2.5}$$

We shall find another way to satisfy the connectivity principle by allowing $g(0) \neq 0$ in §4.

To get rid of possible noise present on the initial image which could be propagated to the interior of the inpainting domain, a fidelity term is used as in the TV model. Thus, by defining the vector field $V = \langle V^1, V^2 \rangle$ by $V = G\frac{\nabla u}{|\nabla u|}$ with $G = G[(x, y), |\kappa|]$

$$G = \begin{cases} 1 & (x, y) \in \Omega \backslash D \\ |\kappa|^p & (x, y) \in D, \end{cases} \tag{2.6}$$

the CDD scheme is to solve the following third order nonlinear equation for u:

$$\nabla \cdot V + \lambda_E(z - u) = 0 \quad \text{with} \quad \frac{\partial u}{\partial \vec{n}} = 0 \ \text{ in } \ \partial\Omega, \tag{2.7}$$

where $\vec{n}$ and λ_E are defined as before. Since V contains the term $|\nabla u|^{-1}$, to avoid the singularity at flat regions $|\nabla u|_\beta := \sqrt{|\nabla u|^2 + \beta}$ is used instead of $|\nabla u|$, where β is a small parameter. Equation (2.7) will become

$$\alpha \nabla \cdot V + \chi(z - u) = 0 \tag{2.8}$$

if we let

$$\alpha = \begin{cases} \frac{1}{\lambda} & \text{in } \Omega \backslash D \\ 1 & \text{in } D \end{cases} \quad \text{and} \quad \chi = \begin{cases} 1 & \text{in } \Omega \backslash D \\ 0 & \text{in } D. \end{cases}$$

3. Review of Numerical Methods

In this section, we intend to review the state-of-the-art methods for numerically solving the CDD model. Surprisingly the list is very short and it only has been solved using an explicit time marching scheme.

3.1. Discretization

We start by discretizing the CDD model (2.7) for a general coefficient G as follows

$$\frac{V^1_{i+\frac{1}{2},j} - V^1_{i-\frac{1}{2},j}}{h} + \frac{V^2_{i,j+\frac{1}{2}} - V^2_{i,j-\frac{1}{2}}}{h} + \lambda_E(z_{i,j} - u_{i,j}) = 0. \tag{3.1}$$

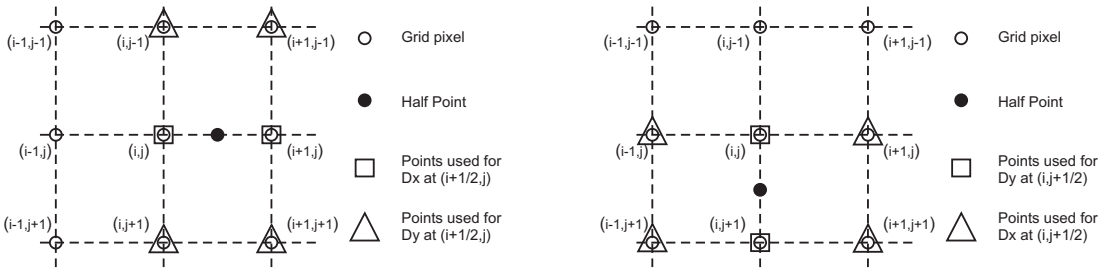

Fig. 3.1. On the left side an x-half-point and on the right side a y-half-point

The finite difference scheme is illustrated in Figure 3.1. Here we assume h is the stepsize and the discrete image $z \in \Re^{m\times n}$; we shall mainly consider the case of $n = m$.

Now we have to approximate V^1 and V^2 at the half-points, for instance at $(i+\frac{1}{2}, j)$, u_x is approximated by central differences $(u_x)_{i+\frac{1}{2},j} = (u_{i+1,j} - u_{i,j})/h$, u_y by average approximation

$$(u_y)_{i+\frac{1}{2},j} = (u_{i+1,j+1} - u_{i+1,j-1} + u_{i,j+1} - u_{i,j-1})/4h$$

and $|\nabla u|_\beta$ in the natural way :

$$\frac{1}{h}\sqrt{(u_{i+1,j} - u_{i,j})^2 + \left(\frac{1}{4}(u_{i+1,j+1} - u_{i+1,j-1} + u_{i,j+1} - u_{i,j-1})\right)^2 + h^2\beta} \tag{3.2}$$

Hence Eq.(3.1) becomes

$$\begin{aligned} &-G_{i+\frac{1}{2},j}\left(\frac{\alpha(u_x)_{i+\frac{1}{2},j}}{h|\nabla u|_{i+\frac{1}{2},j}}\right) + G_{i-\frac{1}{2},j}\left(\frac{\alpha(u_x)_{i-\frac{1}{2},j}}{h|\nabla u|_{i-\frac{1}{2},j}}\right) \\ &-G_{i,j+\frac{1}{2}}\left(\frac{\alpha(u_y)_{i,j+\frac{1}{2}}}{h|\nabla u|_{i,j+\frac{1}{2}}}\right) + G_{i,j-\frac{1}{2}}\left(\frac{\alpha(u_y)_{i,j-\frac{1}{2}}}{h|\nabla u|_{i,j-\frac{1}{2}}}\right) + \chi u_{i,j} = \chi z_{i,j}. \end{aligned} \tag{3.3}$$

To approximate the curvature term κ in G or in (2.4), we use the same idea of the ghost half points to approximate the divergence operator

$$\kappa = \nabla \cdot \frac{\nabla u}{|\nabla u|} = \frac{\partial}{\partial x}\left[\frac{u_x}{|\nabla u|}\right] + \frac{\partial}{\partial y}\left[\frac{u_y}{|\nabla u|}\right]. \tag{3.4}$$

By using again central differences and averages we have for example at $(i+\frac{1}{2}, j)$ that

$$h \cdot \frac{\partial}{\partial x}\left[\frac{u_x}{|\nabla u|}\right]_{i+\frac{1}{2},j} = \left[\frac{u_x}{|\nabla u|}\right]_{i+1,j} + \left[\frac{u_x}{|\nabla u|}\right]_{i,j}. \tag{3.5}$$

$$\begin{aligned} 4h \cdot \frac{\partial}{\partial y}\left[\frac{u_y}{|\nabla u|}\right]_{i+\frac{1}{2},j} &= \left[\frac{u_x}{|\nabla u|}\right]_{i+1,j+1} - \left[\frac{u_x}{|\nabla u|}\right]_{i+1,j-1} \\ &\quad + \left[\frac{u_x}{|\nabla u|}\right]_{i,j+1} - \left[\frac{u_x}{|\nabla u|}\right]_{i,j-1}. \end{aligned} \tag{3.6}$$

Finally (2.7) becomes a system of nonlinear algebraic equations denoted by

$$\begin{aligned} u_{i,j}\mathbb{S}_{i,j} - u_{i+1,j}\left(C_{i+\frac{1}{2},j}\right) - u_{i-1,j}\left(C_{i-\frac{1}{2},j}\right) \\ - u_{i,j+1}\left(C_{i,j+\frac{1}{2}}\right) - u_{i,j-1}\left(C_{i,j-\frac{1}{2}}\right) - \chi z_{i,j} = 0 \end{aligned} \tag{3.7}$$

where the new C notation represents the nonlinear terms, for instance,

$$C_{(i+\frac{1}{2},j)} = \frac{\alpha\, G_{(i+\frac{1}{2},j)}}{h|\nabla u|_{(i+\frac{1}{2},j)}} \tag{3.8}$$

(and the notation for C at the other three half-points is similar) and $\mathbb{S}_{i,j}$ is defined as

$$\mathbb{S}_{i,j} = C_{i+\frac{1}{2},j} + C_{i-\frac{1}{2},j} + C_{i,j+\frac{1}{2}} + C_{i,j-\frac{1}{2}} + \chi. \tag{3.9}$$

The Neumann's boundary condition on $\partial\Omega$ is determined by the TV denoising model and therefore is treated as

$$u_{i,0} = u_{i,1}, u_{i,n+1} = u_{i,n}, u_{0,j} = u_{1,j}, u_{m+1,j} = u_{m,j}. \tag{3.10}$$

On the other hand, as in [10], the inpainting domain D is mathematically understood as an open set, i.e., not including its boundary and therefore away from $\partial\Omega$.

3.2. Explicit time marching method

In this method, solving (2.7) indirectly, one looks for the steady-state solution of a parabolic equation of the form:

$$\frac{\partial u}{\partial t} = r(u), \quad \text{with } r(u) = \nabla \cdot V + \lambda_E(z-u), \tag{3.11}$$

or, $r(u) = \alpha\nabla \cdot V + \chi(z-u)$, with the initial condition $u(x,y,0) = z(x,y)$ and appropriate boundary conditions and using an explicit Euler method for the left hand side, we get

$$u_{i,j}^{k+1} = u_{i,j}^{k} - \tau r(u_{i,j}^{k}), \quad k = 0, 1, \ldots \tag{3.12}$$

Here a size restriction on the time step $\tau = \Delta t$ has to be imposed to guarantee the stability of the numerical solution. This is the main drawback of the time marching method, the problem being that due to its high nonlinearity, τ must be chosen very small which implies a large number of iterations to reach a meaningful solution. One option is to accelerate this method using the ideas developed in [20]. However, even in that case the cpu-time consumed by the resulting algorithm is still not appropriate for large images.

3.3. A possible fixed point method

For numerically solving the nonlinear algebraic equation (3.3) at each (i,j) point we fix the nonlinear terms G and $|\nabla u|$ at some k-step and solve for the $k+1$ step as in [1, 27] for other problems. We have from (3.7) that

$$\begin{aligned} u_{i,j}^{k+1}\mathbb{S}_{i,j} - u_{i+1,j}^{k+1}\left(C_{i+\frac{1}{2},j}^{k}\right) - u_{i-1,j}^{k+1}\left(C_{i-\frac{1}{2},j}^{k}\right) \\ - u_{i,j+1}^{k+1}\left(C_{i,j+\frac{1}{2}}^{k}\right) - u_{i,j-1}^{k+1}\left(C_{i,j-\frac{1}{2}}^{k}\right) = \chi z_{i,j} \end{aligned} \tag{3.13}$$

where similar to before

$$C_{(i+\frac{1}{2},j)}^{k} = \frac{\alpha\, G_{(i+\frac{1}{2},j)}^{k}}{h|\nabla u^{k}|_{(i+\frac{1}{2},j)}} \tag{3.14}$$

and so on, and

$$\mathbb{S}_{i,j} = C_{i+\frac{1}{2},j}^{k} + C_{i-\frac{1}{2},j}^{k} + C_{i,j+\frac{1}{2}}^{k} + C_{i,j-\frac{1}{2}}^{k} + \chi. \tag{3.15}$$

Then such a fixed point method amounts to solving the linear system of (3.13)

$$A(\mathbf{u}^k)\mathbf{u}^{k+1} = \mathbf{z}, \tag{3.16}$$

where $\mathbf{u}^k$ and $\mathbf{z}$ are defined as $\mathbf{u}^k = [u^k_{1,1}, u^k_{2,1} \ldots, u^k_{n,1}, u^k_{1,2}, \ldots, u^k_{n,m}]$ and $\mathbf{z} = [\chi z_{1,1}, \chi z_{2,1} \ldots, \chi z_{n,1}, \chi z_{1,2}, \ldots, \chi z_{n,m}]$.

The selection of the diffusion coefficient G in D plays a crucial role on the feasibility of the implementation of the numerical scheme. According to Chan and Shen [10], on the inpainting domain, G must obey equations (2.5) and (2.6). Therefore it must be

$$G = \begin{cases} 0 & \kappa = 0, \\ \infty & \kappa = \infty, \\ |\kappa|^p, & 0 < |\kappa| < \infty \ \text{ with } \ p \geq 1. \end{cases} \tag{3.17}$$

Since we allow G to be 0 when $\kappa = 0$, matrix $A(\mathbf{u}^k)$ is singular i.e. whenever $\kappa = 0$ at one (i, j) pixel of the image then $A(\mathbf{u}^k)$ losses one degree of its rank. Therefore the fixed point (FP) method (3.16) does not work for the CDD model with (3.17).

One solution (motivated by numerical consideration) is to modify the above G to $\bar{G} = G + \epsilon$, where ϵ is a small and positive parameter. This idea will be tested shortly.

4. Nonlinear Multigrid for a Modified CDD Model

Multigrid methods (MG) have proved to be very useful when solving many linear (and some nonlinear) partial differential equations (PDEs) such as those arising from image restoration problems and others, see [7,8,15,24–26] for successful examples. Usually for a multigrid method to converge, a suitable smoother is the key and the task of finding one is nontrivial for a nonlinear problem.

We now proceed to develop a multigrid algorithm for the CDD formulation (2.7):

$$\nabla \cdot \left(G_{i,j} \frac{(\nabla u)_{i,j}}{|\nabla u|_{i,j}} \right) + \lambda_E (z_{i,j} - u_{i,j}) = 0. \tag{4.1}$$

4.1. The generic algorithm

First of all, we start by introducing new notation and rewriting the equation for the purpose of making it more tractable for computing implementation. Write (4.1) as

$$(Nu)_{i,j} = -\alpha \nabla \cdot \left(G_{i,j} \frac{(\nabla u)_{i,j}}{|\nabla u|_{i,j}} \right) + \chi u_{i,j} = \chi z_{i,j}, \tag{4.2}$$

after we have denoted by $Nu = \chi z$ the main nonlinear operator equation; see (2.8). Since we have to approximate this equation on grids of different sizes we will denote by $N_h u_h = \chi z_h$ the discrete equation (4.2) defined on the finest grid Ω_h of size h and similarly by $N_{2h} u_{2h} = \chi z_{2h}$ the same on the coarser grid Ω_{2h} which is obtained by standard coarsening, i.e., the nonlinear operator N_{2h} which results from defining equation (4.2) on the cell-centered grid Ω_{2h} with grid spacing $2h$. Likewise we can generate a sequence of L coarse levels $2h, 4h, 8h, \ldots, 2^L h$.

Next we briefly mention the standard intergrid transfer operators. Denote by R^{2h}_h (restriction) and I^h_{2h} (interpolation) respectively two transfer operators between Ω_h and Ω_{2h} which

on cell-centered grids are defined by the following equations [26]: The Restriction operator is defined by $R_h^{2h}u_h = u_{2h}$ where, $1 \le i \le n/2,\ \ 1 \le j \le m/2$,

$$(u_{2h})_{i,j} = \frac{1}{4}\Big[(u_h)_{2i-1,2j-1} + (u_h)_{2i-1,2j} + (u_h)_{2i,2j-1} + (u_h)_{2i,2j}\Big] \tag{4.3}$$

and the Interpolation operator is defined by $I_{2h}^h u_{2h} = u_h$ where, $1 \le i \le n/2,\ \ 1 \le j \le m/2$,

$$\begin{aligned}
(u_h)_{2i,2j} &= \big[9(u_{2h})_{i,j} + 3[(u_{2h})_{i+1,j} + (u_{2h})_{i,j+1}] + (u_{2h})_{i+1,j+1}\big]/16,\\
(u_h)_{2i-1,2j} &= \big[9(u_{2h})_{i,j} + 3[(u_{2h})_{i-1,j} + (u_{2h})_{i,j+1}] + (u_{2h})_{i-1,j+1}\big]/16,\\
(u_h)_{2i,2j-1} &= \big[9(u_{2h})_{i,j} + 3[(u_{2h})_{i+1,j} + (u_{2h})_{i,j-1}] + (u_{2h})_{i+1,j-1}\big]/16,\\
(u_h)_{2i-1,2j-1} &= \big[9(u_{2h})_{i,j} + 3[(u_{2h})_{i-1,j} + (u_{2h})_{i,j-1}] + (u_{2h})_{i-1,j-1}\big]/16.
\end{aligned} \tag{4.4}$$

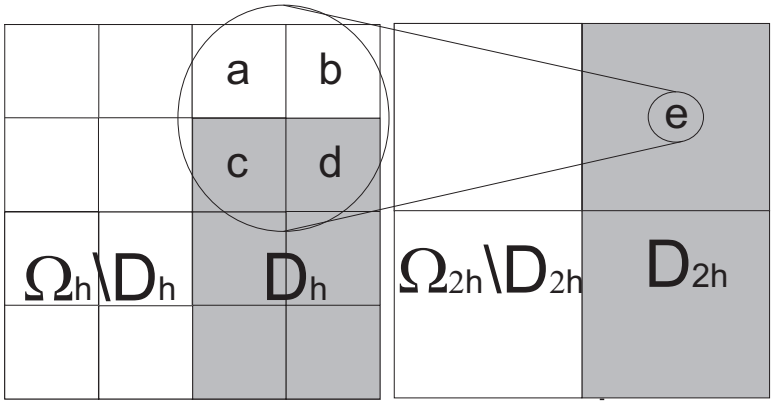

Fig. 4.1. When applying standard coarsening to Ω_h, some points in Ω_{2h} are computed using partial information coming from D_h and thus we need to include those points in D_{2h}

Finally we discuss how to coarsen the interfaces, unique to inpainting problems. It suffices to discuss 2 consecutive grids. First define as D_h the inpainting domain in the finest grid Ω_h and D_{2h} is the coarse grid counterpart to be constructed on Ω_{2h}. Basically, D_h is identified using a binary mask matrix M_h composed of 1's for points in D_h and 0's for points in $\Omega_h \setminus D_h$. The question is then how to construct a similar M_{2h} on D_{2h}. A simple way is by restriction $M_{2h} = R_h^{2h} M_h$. The problem in doing so is that M_{2h} will have some orphan entries due to the action of the operator R_h^{2h} on M_h. Those entries precisely identify those interface points in Ω_{2h} which were not aligned to the coarse grid. In Figure 4.1 we give an example of such a case. There the points a, b, c, d in Ω_h are used to compute the coarse point e in Ω_{2h}, however c and d belong to the inpainting domain D_h and therefore e will be an orphan. For that reason we decided to include such type of points into the inpainting domain D_{2h} by setting their respective entries in M_{2h} to 1 (i.e., for M_{2h} to take all orphan points if any). The problem in applying this strategy is that the mask starts expanding when moving in the coarsest direction and eventually (depending on the size and topology of D_h) may reach the physical boundary $\partial\Omega$ at some coarse level; in other cases where D_h is disconnected some of its disconnected parts can merge themselves as well. Clearly for inpainting we have a very difficult interpolation problem and it is by no means an easy task to approximate the error at coarse levels. Furthermore we observed that for some problems additional errors were incorporated by coarse levels. Hence we decided to stop the mask generation at the level for which the mask is just one pixel away

from the boundary. This decision obviously prevents using those coarsest levels with only a few points; however this was the one that worked best.

We have also tried another strategy based on the idea of trying to reach the coarsest level using an adaptive mask generation. In this strategy the mask first starts growing till close to the boundary (like in our above idea) but then starts contracting or at least keeping one pixel away from the boundary. Unfortunately, this seemingly reasonable idea gave worst results than the above one (particularly for inpainting binary images).

To proceed, denote by FPS a general fixed-point type smoother; we shall define it shortly. Now we state our V-cycling nonlinear MG below, meaning that just one recursive call to the algorithm is made on each level to approximately solve a coarse grid problem.

Algorithm 1 (Nonlinear Multigrid Method)
Select an initial guess u_h *on the finest grid* h.
Set $k = 0$ *and* $err = tol{+}1$.
While $err < tol$
$u_h^{k+1} \leftarrow FAS(u_h^k, N_h^k, z_h, M_h, \nu_0, \nu_1, \nu_2, gsiter)$
$err = \|u_h^{k+1} - u_h^k\|_2, \quad k = k + 1$
end

Here the full approximation scheme (FAS) algorithm is defined [24, 26] recursively as follows:

Algorithm 2 (FAS) $u_h \leftarrow FAS(u_h, N_h, z_h, M_h, \nu_0, \nu_1, \nu_2, gsiter)$

1. *If* Ω^h = *coarsest grid, solve* $N_h u_h = \chi z_h$ *accurately (i.e.* ν_0 *iterations by FPS) and return Else continue with step 2.*
2. *Pre-smoothing: For* $l = 1$ *to* ν_1, $\quad u_h \leftarrow FPS(u_h, z_h, gsiter, M_h)$.
3. *Restrict to the coarse grid,* $M_{2h} \leftarrow R_h^{2h} M_h$ *and* $u_{2h} \leftarrow R_h^{2h} u_h$.
4. *Set the initial solution for the next level,* $\bar{u}_{2h} \leftarrow u_{2h}$.
5. *Compute the new right hand side* $\chi z_{2h} \leftarrow R_h^{2h}(\chi z_h - N_h u_h) + N_{2h} u_{2h}$.
6. *Implement* $u_{2h} \leftarrow FAS_{2h}(u_{2h}, N_{2h}, z_{2h}, M_{2h}, \nu_0, \nu_1, \nu_2, gsiter)$.
7. *Add the residual correction,* $u_h \leftarrow u_h + I_{2h}^h(u_{2h} - \bar{u}_{2h})$.
8. *Post-smoothing: For* $l = 1$ *to* ν_2, $\quad u_h \leftarrow FPS(u_h, z_h, gsiter, M_h)$.

Here *gsiter* represents the number of inner Gauss-Seidel iterations at each pre or post-smoothing FPS step.

4.2. The modified CDD and the FP smoother

Although the above algorithm appears applicable to solving (4.2), it requires a suitable smoother FPS. To this end, we have tried various choices of FPS but failed to find a working smoother; the fixed-point method from §3.3 cannot be used as an efficient smoother.

In trying to find a smoother that would work with Algorithm 2, our idea is to address the problem described in §3.3 by overcoming the singularity associated with the FP method. Motivated by the Euler's Elastica model [9], we decided to introduce two positive parameters a and b in the function g. Specifically our proposal for the diffusion coefficient G is

$$G = \begin{cases} a & \kappa = 0 \\ \infty & \kappa = \infty \\ a + b|\kappa|^p, & \text{with } p \geq 1 \text{ and } 0 < |\kappa| < \infty. \end{cases} \tag{4.5}$$

For a general p, our modified CDD takes the form

$$\nabla \cdot (a + b|\kappa|^p) \frac{\nabla u}{|\nabla u|} + \lambda_E (z - u) = 0. \tag{4.6}$$

Since $\kappa \to 0$ in flatter regions, it is speculated in [10] that the connectivity principle is put at risk if the diffusion coefficient is not zero. We shall demonstrate via our numerical experiments that for the modified model (4.6) as long as a is sufficiently small we can satisfy the connectivity principle, i.e., reconnect the contours across large distances. However if we select a too small compared with b we introduce instability to the linear system (3.7). Experimentally we found that $20 < \frac{b}{a} < 250$ works well.

Note that, setting $b = 1$ and $a = 0$ reduces the modified model to the original CDD model [10]. On the other hand, setting $a = 1$ and $b = 0$ transforms the modified model to the TV model [11]. Therefore our inpainting model (4.6) is half way between the CDD model and the TV model. Moreover, as we shall see, it inherits the virtue of the original CDD model in reconstructing large scale missing parts while sharing with the TV inpainting model the advantage of having a working fixed point (FP) method which in turn can provide a fast and efficient smoother for a multilevel method.

4.2.1. A global smoother

Now we can similarly consider a fixed point method for numerically solving the discretized equation of (4.6) at each (i, j) point. To do this we fix the nonlinear terms G and $|\nabla u|$ at the current k-step and solve for the new $k+1$ step. Thus we again obtain that

$$\begin{aligned} u_{i,j}^{k+1} \mathbb{S}_{i,j} - u_{i+1,j}^{k+1} \left(C_{i+\frac{1}{2},j}^k \right) - u_{i-1,j}^{k+1} \left(C_{i-\frac{1}{2},j}^k \right) \\ - u_{i,j+1}^{k+1} \left(C_{i,j+\frac{1}{2}}^k \right) - u_{i,j-1}^{k+1} \left(C_{i,j-\frac{1}{2}}^k \right) = \chi z_{i,j} \end{aligned} \tag{4.7}$$

where

$$\begin{aligned} C_{(i+\frac{1}{2},j)}^k &= \frac{\alpha \, G_{(i+\frac{1}{2},j)}^k}{h |\nabla u^k|_{(i+\frac{1}{2},j)}}, \\ \mathbb{S}_{i,j} &= C_{i+\frac{1}{2},j}^k + C_{i-\frac{1}{2},j}^k + C_{i,j+\frac{1}{2}}^k + C_{i,j-\frac{1}{2}}^k + \chi, \end{aligned} \tag{4.8}$$

whose corresponding linear system $A(\mathbf{u}^k)\mathbf{u}^{k+1} = \chi \mathbf{z}$ is now never singular and hence solvable.

We can check that in this case $A(\mathbf{u}^k)$ is a symmetric and sparse block-tridiagonal matrix with the important feature that it is weakly diagonally dominant. To show this, we can choose and arbitrary row of it and see that all the $a_{i,j}$ with $i \neq j$ entries but at most four are equal to zero. The nonzero entries in this row are given by $C_{i+\frac{1}{2},j}^k$, $C_{i-\frac{1}{2},j}^k$, $C_{i,j+\frac{1}{2}}^k$ and $C_{i,j-\frac{1}{2}}^k$ respectively

and all are positive. The diagonal entry $a_{i,i}$ on the other hand is computed using (4.8). Thus, we have that

$$a_{i,i} \geq \sum_{i \neq j}^{n} |a_{i,j}| \ \text{ in } \ \Omega$$

with strict inequality in $\Omega \backslash D$ and therefore $A(\mathbf{u}^k)$ is weakly diagonally dominant. Furthermore, because

$$G \equiv 1 \ \text{ in } \ \Omega \backslash D \ \text{ and } \ G = a + b|\kappa|^p \ \text{ with } \ p \geq 1 \ \text{ in } \ D$$

we can deduce by using the Gerschgorin theorem that $A(\mathbf{u}^k)$ is Positive Semi-Definite.

Therefore, with $A(\mathbf{u}^k)$ having such a property, we can apply the Gauss-Seidel (GS) iterations to solve the linear system (4.7):

$$A(\mathbf{u}^k)\mathbf{u}^{k+1} = \chi \mathbf{z}.$$

As a smoother, we shall name this fixed-point method with GS iterations the FPGS smoother, which can be stated as follows.

Algorithm 3 (FPGS Smoother) $u_h \leftarrow FPGS(u_h, z_h, gsiter, M_h)$
Choose an initial guess $\boldsymbol{u}_h$.
for $k = 1$ *to gsiter*
Apply gsiter Gauss Seidel iterations to the linear system $A_h(\boldsymbol{u}_h^k)\boldsymbol{u}_h^{k+1} = \boldsymbol{z}_h$
end

4.2.2. A local smoother

Following from [1, 15, 25] for solving other PDEs, we also considered a local-type fixed point smoother. The difference of this smoother from the global one is that we apply the relaxation steps *locally*. The scheme is the same as that of (4.7) with k representing the kth step. The idea is to update $u_{i,j}$ with the nonlinear terms C^k fixed at each k local step meaning that when we apply relaxation to the next (i, j) point, some C nonlinear terms have been updated. We call this smoother FPLS and is defined as follows:

Algorithm 4 (FPLS Smoother) $u_h \leftarrow FPLS(u_h, z_h, gsiter, M_h)$
for $i = 1$ *to* m
for $j = 1$ *to* n
for $k = 1$ *to gsiter*
$\bar{u}_h \leftarrow u_h$ *and update* $u_{i,j}$ *by solving the linear equation*

$$\begin{aligned} u_{i,j}^{k+1}\bar{\mathbb{S}}_{i,j} - u_{i+1,j}^{k+1}\left(\bar{C}_{i+\frac{1}{2},j}^{k}\right) - u_{i-1,j}^{k+1}\left(\bar{C}_{i-\frac{1}{2},j}^{k}\right) \\ - u_{i,j+1}^{k+1}\left(\bar{C}_{i,j+\frac{1}{2}}^{k}\right) - u_{i,j-1}^{k+1}\left(\bar{C}_{i,j-\frac{1}{2}}^{k}\right) = \chi z_{i,j} \end{aligned} \tag{4.9}$$

end
end
end

Here $\bar{C}_{i,j}^k$ means C^k evaluated at $\bar{u}_{i,j}$ and the same applies to $\bar{\mathbb{S}}_{i,j}$, as in (4.8).

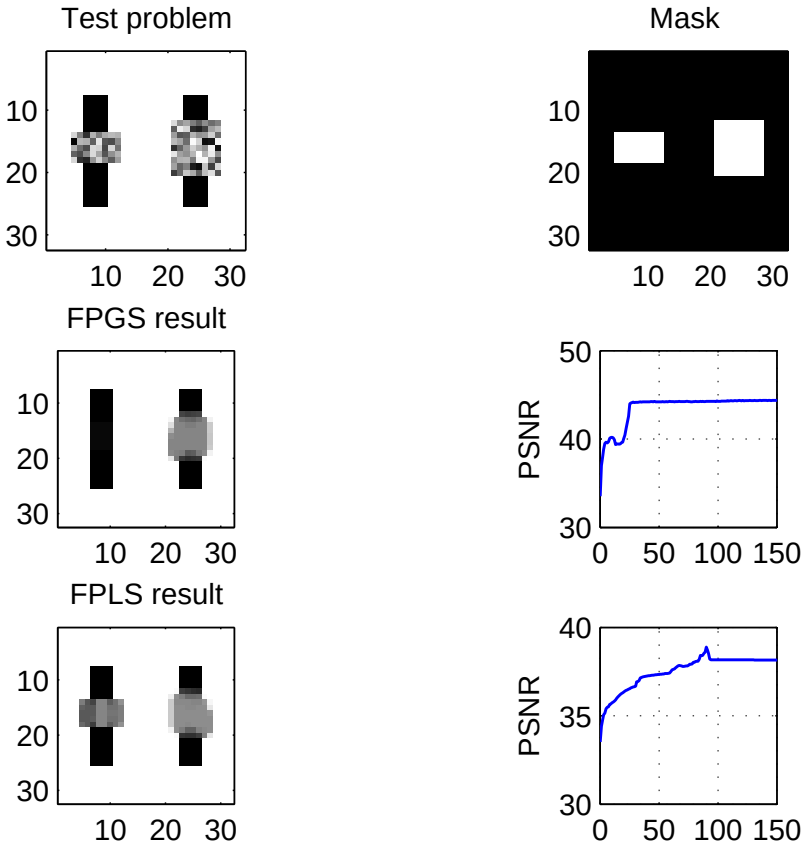

Fig. 4.2. *Test problem.*

4.2.3. Use of the smoothers as independent methods

On their own, both smoothers are not always convergent; this situation is different from the image denoising case [27]. Consider the *test problem* illustrated at the top of Figure 4.2. In the middle and at the bottom of the same Figure we show the results obtained from using FPGS and FPLS as independent methods. The failure of both in trying to solve the problem is evident. The PSNR measure used is as defined in §5.

However our main interest is not on the convergence of both smoothers but in their smoothing capabilities. In this case as we shall show, FPGS has better smoothing rates than FPLS and therefore it is our preferred smoother for a nonlinear MG algorithm.

4.2.4. Local Fourier analysis

We just have seen that both FPGS and FPLS are not always convergent. However to be used in a multigrid algorithm we only need them to reduce (smooth) as much as possible the high frequency components of the error regardless the overall error itself.

When dealing with nonlinear problems the local Fourier analysis (LFA) can still be used as a tool to check if a given smoother is effective in reducing the high frequency components;

see [1, 26]. For simplicity we consider the case of a square image of size $m \times m$. Let the local error functions $e_{i,j}^{k+1}$ and $e_{i,j}^{k}$ be defined as $e_{i,j}^{k+1} = u_{i,j} - u_{i,j}^{k+1}$ and $e_{i,j}^{k} = u_{i,j} - u_{i,j}^{k}$. The LFA involves expanding

$$\begin{aligned} e_{i,j}^{(k+1)} &= \sum_{\theta_1,\theta_2=-m/2}^{m/2} \psi_{\theta_1,\theta_2}^{k+1} B_{\theta_1,\theta_2}(x_i, y_j), \\ e_{i,j}^{(k)} &= \sum_{\theta_1,\theta_2=-m/2}^{m/2} \psi_{\theta_1,\theta_2}^{k} B_{\theta_1,\theta_2}(x_i, y_j) \end{aligned} \tag{4.10}$$

in Fourier components. Here $B_{\theta_1,\theta_2}(x_i, y_j)$ is the general Fourier component defined as

$$B_{\theta_1,\theta_2}(x_i, y_j) = \exp\left(\mathbf{i}\alpha_1 \frac{x_i}{h} + \mathbf{i}\alpha_2 \frac{y_j}{h}\right) = \exp\left(\frac{2\mathbf{i}\theta_1 i\pi}{m} + \frac{2\mathbf{i}\theta_2 j\pi}{m}\right) \tag{4.11}$$

with $\alpha_1 = 2\theta_1\pi/m$, $\alpha_2 = 2\theta_2\pi/m \in [-\pi, \pi]$. From (4.7) and (4.9) we obtain

$$\begin{aligned} -\Big(C_{i+\frac{1}{2},j} + C_{i-\frac{1}{2},j} + C_{i,j+\frac{1}{2}} + C_{i,j-\frac{1}{2}}\Big) e_{i,j}^{k+1} + C_{i+\frac{1}{2},j}\, e_{i+1,j}^{k} \\ + C_{i-\frac{1}{2},j}\, e_{i-1,j}^{k+1} + C_{i,j+\frac{1}{2}}\, e_{i,j+1}^{k} + C_{i,j-\frac{1}{2},j}\, e_{i,j-1}^{k+1} = 0. \end{aligned}$$

Therefore the local amplification factor $\mu_{i,j} = \left|\psi_{\theta_1,\theta_2}^{k+1}/\psi_{\theta_1,\theta_2}^{k}\right|$ is defined by

$$\mu_{i,j}(\theta_1, \theta_2) = \frac{\left|C_{i+\frac{1}{2},j}\, e^{\mathbf{i}\alpha_1} + C_{i,j+\frac{1}{2}}\, e^{\mathbf{i}\alpha_2}\right|}{\left|C_{i+\frac{1}{2},j} + C_{i-\frac{1}{2},j} + C_{i,j+\frac{1}{2}} + C_{i,j-\frac{1}{2}} - C_{i-\frac{1}{2},j}\, e^{-\mathbf{i}\alpha_1} - C_{i,j-\frac{1}{2}}\, e^{-\mathbf{i}\alpha_2}\right|}. \tag{4.12}$$

What follows is to compute the nonlinear coefficients $C_{(\cdot,\cdot)}$ at each (i, j) point and find the maximum for the kth step

$$\bar{\mu}_{i,j} = \max_{\theta_1,\theta_2} \mu_{i,j}(\theta_1, \theta_2)$$

in the high frequency interval $(\alpha_1, \alpha_2) \in [-\pi, \pi] \setminus [-\pi/2, \pi/2]$. Recall that FPGS and FPLS smoothers only differ in when to update the C coefficients.

Since our FPGS and FPLS smoothers are linearized at each step, their smoothing rates will change at every outer iteration. Then we have an $m \times m$ rate matrix $\bar{M}_k$, for the kth step, with entry $\bar{\mu}_{i,j}$ representing the local smoothing rates at (i, j) point. Unfortunately, our initial tests showed that the maximum is close to 1 and yet the practical performance of the smoothers appears quite good. This prompted us to look for a better explanation.

In order to evaluate their effectiveness, it turned out that the (new) accumulated rate based on (old) consecutive smoothing rates is well below 1. That is to say, the above maxima are not achieved at the same location (otherwise LFA is not helpful)! For either smoother, suppose we have completed K (accumulated) inner relaxation steps. Let $\bar{M}_k$ denote the corresponding rate matrix (for $1 \le k \le K$); then define

$$\hat{\mu}_K = \max_{i,j} (\bar{M}_1)_{i,j} (\bar{M}_2)_{i,j} \cdots (\bar{M}_K)_{i,j} \tag{4.13}$$

as the accumulated smoothing rate of a relaxation step (over K iterations). Clearly this is a reasonable definition as it takes care of all iterations within a relaxation step; we expect $\hat{\mu}_K < 1$ and of course $\hat{\mu}_K \ll 1$ if the underlying smoother is good. For linear problems, we have a constant value $\bar{\mu}_{i,j} = \bar{\mu}$ so $\hat{\mu}_K = \bar{\mu}^K$.

Finally for completeness, we shall name the resulting global smoother based on modifying (3.17) by $\bar{G} = G + \epsilon$ as FPGS(ϵ). Likewise we shall denote the corresponding multigrid algorithm based on FPGS(ϵ) as MG(ϵ).

As an example, we present in Table 4.1 the accumulated local smoothing rates computed for the first five iterations for the test problem of Figure 4.2.

Table 4.1: Illustration of accumulated smoothing rates for FPLS, FPGS and FPGS(ϵ) smoothers with $gsiter = 10$ used for all tests. Here F4 means FPGS(10^{-4}) and F8 means FPGS(10^{-8}). Note $K = gsiter\ \nu$, where $gsiter$ is the number of inner iterations

Up to outer	$\hat{\mu}_K$ Left bar region				$\hat{\mu}_K$ Right bar region			
iterations ν	FPGS	FPLS	F4	F8	FPGS	FPLS	F4	F8
1	0.5891	0.9620	0.7898	0.7911	0.8354	0.9690	0.8558	0.8568
2	0.5492	0.9448	0.7528	0.7538	0.2567	0.8986	0.7624	0.7651
3	0.5183	0.9333	0.7160	0.7201	0.0420	0.8861	0.7465	0.7506
4	0.4863	0.9327	0.6920	0.6958	0.0037	0.8804	0.6382	0.6337
5	0.3867	0.9320	0.6445	0.6483	0.0016	0.8633	0.5622	0.6216

From Table 4.1 we can argue that FPGS reduces the high frequency modes much faster than FPLS. Clearly the FPGS(ϵ) method is not as effective. We also tested on other inpaiting problems with various inpainting domains and noticed a similar behavior; therefore FPGS is our preferred smoother. Furthermore Table 4.1 also suggests that 3 to 5 fixed point iterations are sufficient to get an smoothing rate comparable to GS for the Poisson equation [26].

5. Numerical Results

In this section, we shall first give results of five different inpainting problems designed to test the performance of the multigrid algorithm, as illustrated in Figures 5.1-5.5. We use the problems in Figures 5.1 and 5.2 (tested in the original CDD paper [10]) to compare the performance of our MG algorithm with the method used in [10] and the MG(ϵ) method.

To measure the restoration quality, we found it useful to use the Peak-Signal-To-Noise-Ratio (PSNR) to check how similar two images u and u^0 of size $m \times n$ are each other. The PSNR is defined as

$$PSNR = 20\ log_{10}\left(\frac{255}{RMSE(u,u^0)}\right), \quad RMSE(u,u^0) = \sqrt{\frac{\sum_{i,j}(u_{i,j}-u^0_{i,j})^2}{mn}}. \tag{5.1}$$

The larger a PSNR is, the better is the restored image. In real life situation, such a measure is not possible because u^0 is not known.

In Table 5.1 we show quantitative measure of the PSNR values for the damaged and restored images, and in Figures 5.1-5.5 the respective images. Here we only used a few MG cycles to obtain the results. Clearly the restored images are good.

Comparison. We now address the efficiency gains. It turns out that our multigrid algorithm is many magnitudes faster, even for inpainting small images. For instance for the inpainting problem in Figure 5.1 with only a 128×128 image, to reach the same accuracy, time marching requires $160,000$ iterations and 7380 CPU seconds to converge whilst our MG only requires 3 V-cycles and 9.6 CPU seconds. Therefore, it is not necessary to do extensive comparisons.

Table 5.1: Initial and Final PSNR values after applying the modified CDD algorithm to the problems presented in this work

Problem	Image Size	initial PSNR	final PSNR
Child, Figure 5.1	128×128	46	95
Bars, Figure 5.3	512×512	50	102
Ring, Figure 5.4	512×512	27	82
Lena, Figure 5.5	512×512	53	95

Table 5.2: Further improvements of MG results by the FMG and comparisons with MG(ϵ). Here '#' denotes the number of MG cycles and ν the number of smoothing steps per grid needed

	Image	MG			FMG			MG(10^{-4})			MG(10^{-8})		
Problem	Size	ν	#	CPU	ν	#	CPU	ν	#	CPU	ν	#	CPU
Lena	128	6	2	7	6	1	6	12	2	17	12	2	17
	256	8	3	47	8	1	22	15	3	87	15	4	116
	512	8	4	290	8	2	159	15	5	635	15	5	640
Bars	128	15	4	79	15	2	12	40	5	71	50	5	81
	256	20	6	350	20	3	193	50	7	667	50	8	760
	512	20	6	978	20	4	696	50	7	2871	50	8	3304
Ring	128	15	4	61	15	1	16	30	4	69	40	5	96
	256	15	5	192	15	1	46	50	6	606	50	8	813
	512	15	5	1165	15	1	241	50	7	4151	50	8	4732

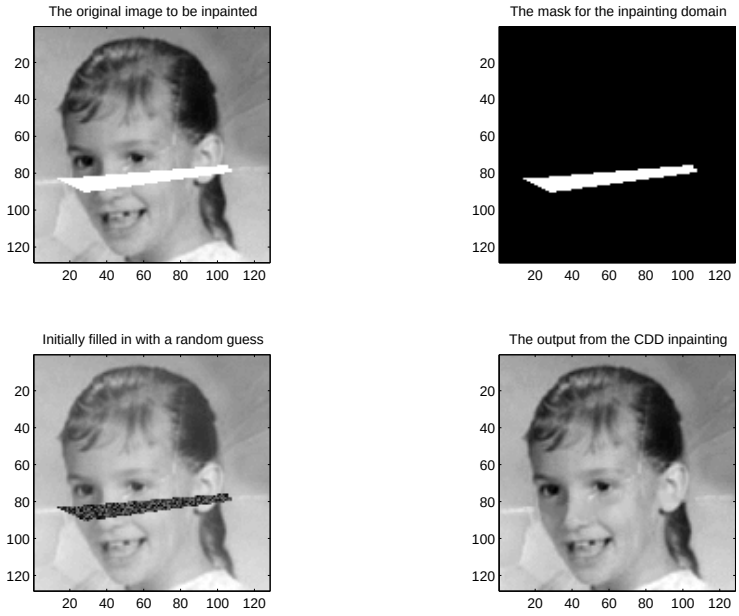

Fig. 5.1. An example of CDD inpainting for scratch removal in a real old-photograph. Notice the big size of the scratch and the very good reconstruction. The algorithm performed 3 V-cycles in only 9.60 seconds

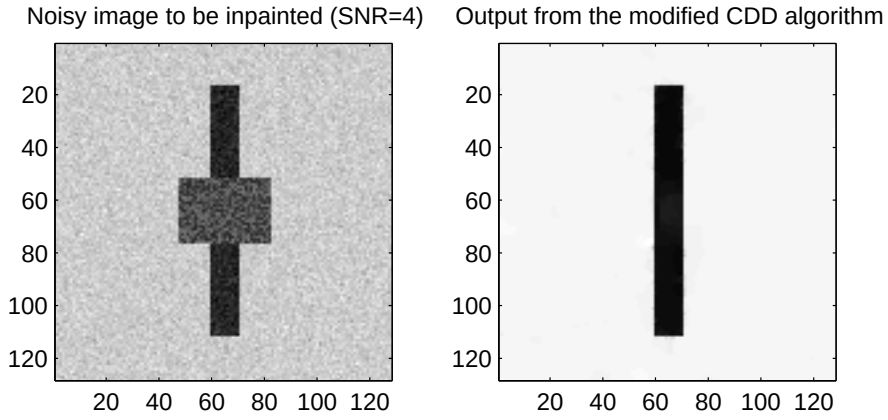

Fig. 5.2. Our algorithm performs very well at noisy images

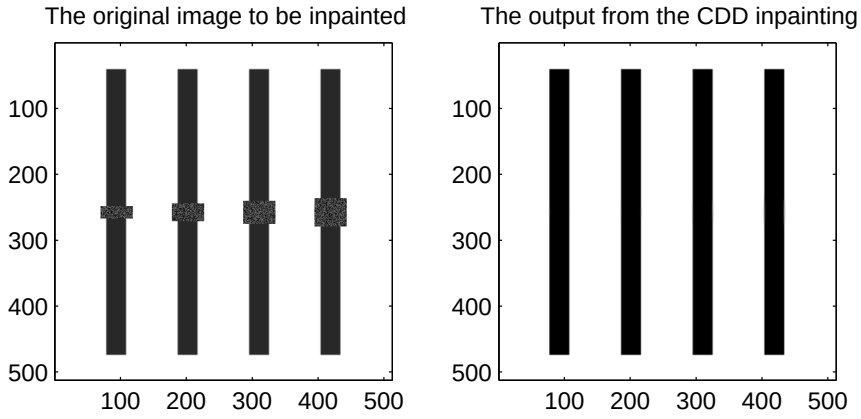

Fig. 5.3. Another example of good reconnection across large distances. Again our MG algorithm performed very well reconstructing this large image in a reasonable amount of time; see Table 4.1

Full Multigrid and MG(ϵ). We observed that for very large scale inpainting domains such as those of the ring problem illustrated in Figure 5.4, the rate of convergence of our MG algorithm is slightly dependent on the initial guess, even though our MG always converges no matter what the initial guess is. In order to reduce this dependence and to improve even more the speed of convergence we adopted the Full Multigrid method (FMG) as described in [26]. Indeed, much better results are obtained and can be seen in Table 5.2, where we also give the results of MG(ϵ). Clearly MG(ϵ) is less efficient than MG and FMG.

6. Conclusions

The original CDD model of [10] improves on the total-variation norm based inpainting model [11, 12]. The only reported time marching scheme is slow in convergence and therefore it is only suitable to process small-sized images. Moreover, the nonlinear MG cannot be easily used to solve the model.

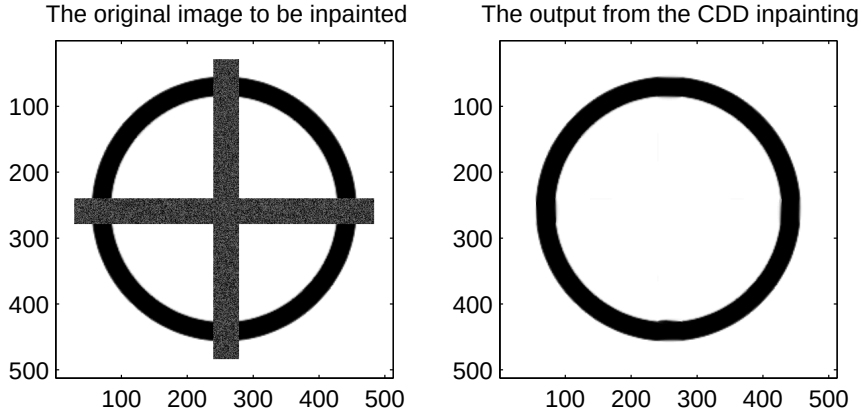

Fig. 5.4. This experiment is presented to emphasize that the CDD model uses straight lines to reconnect the level line contours. The experiment is also particularly important to show the performance of our MG algorithm when an inpainting domain is large

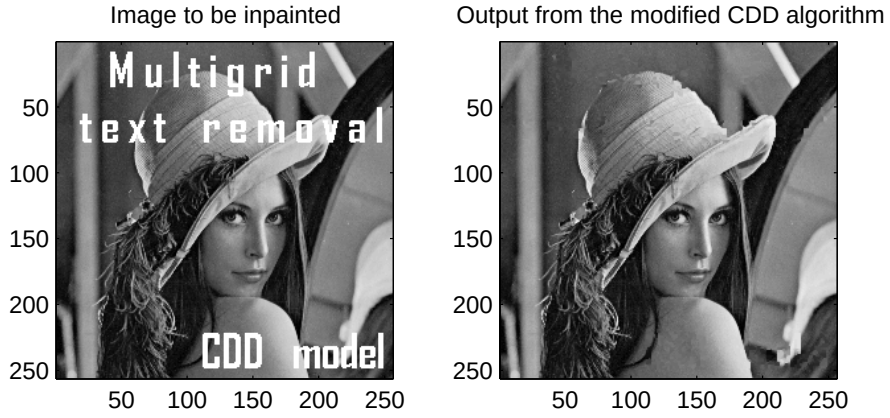

Fig. 5.5. A practical text removal example. Notice the reconnection of the thin piece of hair (2 pixels-width) initially occluded by the thick letter C (4 pixels-width)

In this paper we developed a fast and efficient nonlinear MG algorithm for solving a modified CDD model. By first finding out why a fixed-point method is not feasible for the original CDD model, we then proposed a modified CDD model for which a fixed-point method is feasible and developed a nonlinear MG for the modified model. A local Fourier analysis shows that the global smoother is faster than the local smoother. Numerical results confirmed that the modified model retains the desirable property of the original model in reconnecting the level lines across large distances, and our multigrid method is very efficient.

Acknowledgments. The support to this work by a CONACYT (El Consejo Nacional de Ciencia y Tecnologia) scholarship from Mexico is gratefully acknowledged.

References

[1] N. Badshah and K. Chen, Multigrid method for the Chan-Vese model in variational segmentation, *Commun. Comput. Phys.*, **4**:2 (2008), 294-316.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, Image inpainting, *Siggraph 2000, Proceedings of the 27th Annual Conference On Computer Graphics And Interactive Techniques*, (2000), 417-424.

[3] A.L. Bertozzi, S. Esedoglu, and A. Gillet, Inpainting of binary images using the Cahn-Hilliard equation, *IEEE Transactions on image processing*, **16**:1 (2007), 285-291.

[4] M. Bertalmio, A.L. Bertozzi, and G. Sapiro, Navier-Stokes, fluid dynamics, and image and video inpainting, *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, **1** (2001), 355-362.

[5] V. Caselles, J.M. Morel, and C. Sbert, and A. Gillet, An axiomatic approach to image interpolation, *IEEE Transactions on image processing*, **7**:3 (1998), 376-386.

[6] T.F. Chan and K. Chen, On a nonlinear multigrid algorithm with primal relaxation for the image total variation minimisation, *J. Numer. Algori.*, **41**:4 (2006), 387-411.

[7] T.F. Chan and K. Chen, An optimisation-based multilevel algorithm algorithm for total variation image denoising, *SIAM J. Multi. Mod. Simu*, **5**:2 (2006), 615-645.

[8] T.F. Chan, K. Chen, and Jamylle L. Carter, Iterative methods for solving the dual formulation arisin from image restoration, *Electron. T. Numer. Ana.*, **26** (2007), 299-311.

[9] T.F. Chan, S.H. Kang, and J.-H. Shen, Euler's elastica and curvature based inpaintings, *J. Appl. Math.*, **63**:2 (2002), 564-592.

[10] T.F. Chan and J.-H. Shen, Non-texture inpaintings by curvature-driven diffusions, *J. Vis. Commun. Image R.*, **12**:4 (2001), 436-449.

[11] T.F. Chan and J.-H. Shen, Mathematical models for local nontexture inpaintings, *SIAM J. Appl. Math.*, **62**:3 (2002), 1019-1043.

[12] T.F. Chan and J.-H. Shen, Image Processing and Analysis Variational, PDE, wavelet, and stochastic methods, *SIAM*, (2005).

[13] T.F. Chan and J.-H. Shen, Variational image inpainting, *Commun. Pur. Appl. Math.*, **58**:5 (2005), 579-619.

[14] T.F. Chan, J.-H. Shen, and H.-M. Zhou, Total variation wavelet inpainting, *J. Math. Imaging Vis.*, **25**:1 (2006), 107-125.

[15] K. Chen and J. Savage, An accelerated algebraic multigrid algorithm for total variation denoising, *BIT Numer. Math.*, **47**:2 (2007), 277-296.

[16] S. Esedoglu and J.-H. Shen, Digital inpainting based on the Mumford-Shah-Euler image model, *Eur. J. Appl. Math.*, **13**:4 (2002), 353-370.

[17] H. Grossauer and O. Scherzer, Using the complex Ginzburg-Landau equation for digital inpainting in 2D and 3D, *Scale Space Methods in Computer Vision*, **2695** (2003), 225-236.

[18] J.-W. Gu, L. Zhang, G.-Q. Yu, Y.-X. Xing, and Z.-Q. Chen, X-ray CT metal artifacts reduction through curvature based sinogram inpainting, *Journal of X-Ray Science and Technology*, **14**:2 (2006), 73-82.

[19] A.E.H. Love, A Treatise on the Mathematical Theory of Elasticity, *Dover, New York* 4^{th} *ed.*, (1927).

[20] A. Marquina and S. Osher, Explicit algorithms for a new time dependent model based on level set motion for nonlinear deblurring and noise removal, *SIAM J. Sci. Comput.*, **22**:2 (2000), 387-405.

[21] S. Masnou and J.M. Morel, Level lines based disocclusion, *Proceedings of 5th IEEE Int. Conf. on Image Processing*, **3** (1998), 259-263.

[22] M. Nitzberg, D. Mumford, and T. Shiota, Filtering, Segmentation, and Depth, Springer-Verlag, Lecture Notes in Computer Science, **662** (1993).

[23] L.I. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms,

Physica D, **60**:1-4 (1992), 259-268.

[24] J. Savage and K. Chen, An improved and accelerated non-linear multigrid method for total-variation denoising, *Int. J. Comput. Math.*, **82**:8 (2005), 1001-1015.

[25] J. Savage and K. Chen, On multigrids for solving a class of improved total variation based staircasing reduction models,*In: "Image Processing Based On Partial Differential Equations",* eds. X.-C. Tai, K.-A. Lie, T.F. Chan and S. Osher, Springer-Verlag, (2006), 69-94.

[26] U. Trottenberg, C. Oosterlee, and A. Schuller, Multigrid, Academic Press, New York, 2001.

[27] C. Vogel, Computational Methods For Inverse Problems, SIAM publications, Philadelphia, 2002.