International Journal of Modern Mathematics **5**(2) (2010), 157-182

Fast Numerical Algorithms for Euler's Elastica Inpainting Model

Carlos Brito-Loeza and Ke Chen

Received April 17, 2010; Revised May 20, 2010

Abstract

Euler's elastica digital inpainting model of Chan, Kang and Shen introduced in [*SIAM J. Appl. Math.*, 63(2):564–592, 2002] is well known for its attractive features of reconnecting contours along large distances and reconstructing the curvature of missing parts of objects and its ability to denoise outside the inpainting region. Since the underlying Euler Lagrange partial differential equation (PDE) is of fourth order and highly nonlinear, unfortunately, the usual numerical algorithm to find the solution is a very slow time marching method (due to stability restriction). In this paper we address this fast solution issue by progressively proposing first a new unconditionally stable time marching method and then a novel fixed point method. The latter turns out to be two orders of magnitude faster than the time marching method. Moreover, taking this new fixed point method as a smoother, we develop an even faster nonlinear multigrid method for optimal performance. Numerical results will be presented to illustrate the improved results obtained.

Keywords: Image inpainting, variational models, curvature, elastica energy, 4th order PDE, regularization, multilevel methods.

2000 Mathematics Subject Classification: 68U10, 65F10, 65K10.

1 Introduction

Image inpainting offers a very useful technique to restore the missing or damaged parts of an image using available features present in the same image. This technique is basically image interpolation in a nonlinear manner [6,12,31,32,34]. Although its initial purpose was to restore old photographs and disocclusion, nowadays, it has found wide applications in video editing [5], medical imaging [27], super-resolution [16], error concealment in digital communication [18], and satellite image reconstruction [7] just to mention a few examples.

The first author thanks for the support to this work by a CONACYT (El Consejo Nacional de Ciencia y Tecnología) scholarship from México.

In the last few years, a number of PDE-based variational inpainting models (similar to [6]) have appeared. These include the total variation (TV) model [16], the curvature-driven diffusion model [15], Euler's elastica model [14], the Cahn-Hilliard equation based model [7], the complex Ginzburg-Landau equation based model [26], and the Mumford-Shah and Mumford-Shah-Euler models [22].

Of the above mentioned models, only two models respectively in [14] and [22] can do the following at the same time: reconnection of level curves along large distances, curvature reconstruction and noise elimination. The model from [7] is limited to high contrast or binary images while the other models [15, 16, 26] do not recover the curvature of the objects. The associated PDE of Euler's elastica model [14] is of fourth order and highly non-linear and therefore its efficient numerical solution is non-trivial. In this paper, we focus on developing fast numerical algorithms for solving this Euler's elastica model.

The paper is organized as follows. Section 2 introduces the variational inpainting formulation using the TV model. Section 3 reviews Euler's elastica model and its discretization. Section 4 reviews the current numerical methods for its solution. Section 5 introduces our novel numerical methods for Euler's elastica model: two unconditionally stable time marching schemes and a fixed point scheme. Section 6 studies the smoothing property of this fixed point method using a local Fourier analysis before we introduce the framework of a nonlinear multigrid method in Section 7. Section 8 shows through experimental results the excellent efficiency of the multigrid method, and finally Section 9 illustrates the flexibility and easy adaptability of our methods to solve the similar models [2, 3], before our conclusions in Section 10.

2 Inpainting formulation and TV model

Assume that we are given a possibly noisy image $z = z(x, y)$ defined on a domain $\Omega \subseteq \mathbb{R}^2$ and there exists a subset $D \subset \Omega$ (even disconnected) where the pixel intensity values of z are missing or damaged. The purpose of inpainting is to reconstruct those intensity values of z in D from the available information on $\Omega \backslash D$; see Figure 2.1. The image inpainting problem has been studied by many researchers; see [17] and the references therein for a historical account. Most models allow z to have some unknown Gaussian noise $\eta = \eta(x, y)$ present in $\Omega \backslash D$ such that $z = u + \eta$, where u is the image to be restored (or found).

Within the variational formulation, all models achieve the objective of inpainting by minimizing some suitable energy functional. For instance, the TV model minimizes

$$\int_{\Omega} |\nabla u| \, dxdy + \frac{\lambda}{2} \int_{\Omega \backslash D} (u - z)^2 \, dxdy$$

on u, which leads to solving a second order nonlinear PDE [16]. This model has the

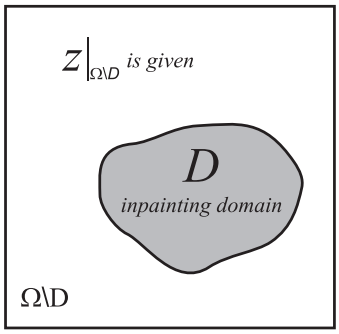


Figure 2.1: Illustration of a typical inpainting problem.

unfortunate properties of creating artificial corners and not reconnecting parts of broken objects along large distances [14, 16]. This is illustrated in Figures 3.2 and 3.3.

3 Euler's elastica inpainting model and its discretization

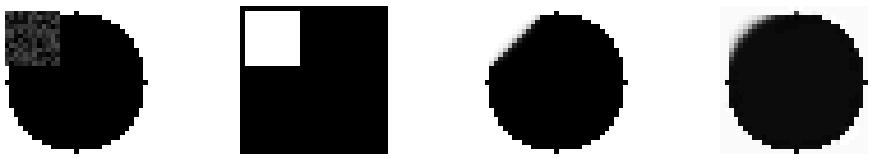
Euler's elastica model by Chan, Kang, and Shen [14] reviewed here is a better variational model than the above TV model as it offers us the capability of inpainting delicate fine features by respecting curvature. We shall refer to this model from now on as the elastica model for simplicity.

3.1 The elastica model

The elastica model [14] minimizing the Euler elastica energy is the following

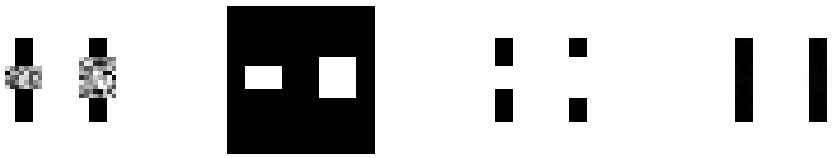
$$\min_u \left\{ J(u) = \int_\Omega \left(a + b\left|\kappa\right|^p\right) \left|\nabla u\right| \, dxdy + \frac{\lambda}{2} \int_{\Omega\backslash D} (u-z)^2 \, dxdy \right\} \tag{3.1}$$

where a and b are arbitrary positive constants, $\lambda > 0$ is a penalty parameter, $p = 2$ is usually chosen, $u = u(x, y)$ is the true image to be restored and $\kappa = \kappa(x, y) \equiv \nabla \cdot \frac{\nabla u}{|\nabla u|}$ is the curvature. The virtue of (3.1) is that the regularization using the Euler elastica energy [29] penalizes the integral of the square of the curvature along edges instead of only penalizing the length of edges as the TV model (if taking $b = 0$) does [9, 17]. Consequently the model can reconnect contours along large distances and recover the curvature of objects at the



(a) (b) (c) (d)

Figure 3.2: The circle problem. (a) A black circle occluded by a noisy square. (b) The mask of the inpainting domain. (c) The TV inpainting result (d) Euler's Elastica inpainting result.



(a) (b) (c) (d)

Figure 3.3: The broken bars problem. (a) Two broken bars with different sizes for the occluding noisy squares. (b) The mask of the inpainting domain. (c) The TV inpainting result (d) Euler's Elastica inpainting result.

same time. The associated Euler-Lagrange (EL) equation is given [14] by

$$\nabla \cdot \left[\left(a + b\kappa^2\right) \frac{\nabla u}{|\nabla u|} - \frac{2b}{|\nabla u|^3} \nabla^{\perp} u \nabla(\kappa |\nabla u|) \nabla^{\perp} u \right] + \lambda_E (z - u) = 0 \ \text{ in } \Omega, \tag{3.2}$$

$$\text{with } \frac{\partial u}{\partial \vec{n}} = 0 \quad \text{and} \quad \frac{\partial((a + b\kappa^2)|\nabla u|)}{\partial \vec{n}} = 0 \ \text{ on } \partial\Omega, \tag{3.3}$$

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

where $\vec{n}$ is the unit outward normal on $\partial\Omega$, $\nabla u = (u_x, u_y)$ and $\nabla^{\perp} u = (-u_y, u_x)$ are the normal and tangential vectors to the level sets of u respectively.

3.2 Discretization by finite differences

To discretizes (3.2) along with (3.3), we use the finite difference method as in [14]. For brevity, we define a vector field $V = \langle V^1, V^2 \rangle$ by $V = \left(a + b\kappa^2\right) \frac{\nabla u}{|\nabla u|} -$

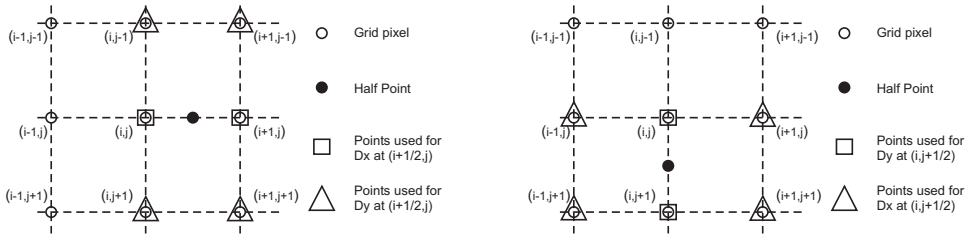


Figure 3.4: On the left side an *x*-half-point and on the right side a *y*-half-point.

$\frac{2b}{|\nabla u|^3}\nabla^\perp u \nabla(\kappa|\nabla u|)\nabla^\perp u$, i.e.

$$V^1 = (a+b\kappa^2)\frac{\partial_x u}{|\nabla u|} + \frac{2b}{|\nabla u|^3}\left[-\partial_y u\ \partial_x(\kappa|\nabla u|) + \partial_x u\ \partial_y(\kappa|\nabla u|)\right]\ \partial_y u,$$
$$V^2 = (a+b\kappa^2)\frac{\partial_y u}{|\nabla u|} - \frac{2b}{|\nabla u|^3}\left[-\partial_y u\ \partial_x(\kappa|\nabla u|) + \partial_x u\ \partial_y(\kappa|\nabla u|)\right]\ \partial_x u,$$

where ∂_ω denotes the derivative with respect to ω. Then with the above notation and after defining $\alpha = 1/\lambda$ and χ as the indicator function of $\Omega\backslash D$, (3.2) becomes

$$\alpha\nabla\cdot V + \chi(z-u) = 0.$$

Next to approximate $\nabla\cdot V = \frac{\partial}{\partial x}V^1 + \frac{\partial}{\partial y}V^2$ at pixel (i,j), we use central differences between ghost half-points; see Figure 3.4. That is

$$\nabla\cdot V_{i,j} = \frac{\left(V^1_{i+\frac{1}{2},j} - V^1_{i-\frac{1}{2},j}\right)}{h} + \frac{\left(V^2_{i,j+\frac{1}{2}} - V^2_{i,j-\frac{1}{2}}\right)}{h},$$

where h is the spatial step-size that for convenience was chosen to be the same in both Cartesian directions. The next step is to approximate all the involved quantities at half-points. First, to approximate $V^1_{i+\frac{1}{2},j}$ and $V^1_{i-\frac{1}{2},j}$, we do the computations as follows :

Curvature terms. These are approximated by the *min-mod* of two adjacent *whole* pixels.

$$\kappa_{i+\frac{1}{2},j} = \textit{min-mod}(\kappa_{i+1,j},\kappa_{i,j}) \ \text{ and } \ \kappa_{i-\frac{1}{2},j} = \textit{min-mod}(\kappa_{i,j},\kappa_{i-1,j})$$
$$\text{where } \ \textit{min-mod}\ (\alpha,\beta) = \left(\tfrac{sgn\ \alpha + sgn\ \beta}{2}\right)\min(|\alpha|,|\beta|).$$

Partial Derivatives in x. By the central differencing of two adjacent *whole* pixels

$$\partial_x(u_{i+\frac{1}{2},j}) = \tfrac{1}{h}(u_{i+1,j} - u_{i,j}),$$
$$\partial_x(u_{i-\frac{1}{2},j}) = \tfrac{1}{h}(u_{i,j} - u_{i-1,j}),$$
$$\partial_x(\kappa|\nabla u|)_{i+\frac{1}{2},j} = \tfrac{1}{h}\left(\kappa_{i+1,j}|\nabla u|_{i+1,j} - \kappa_{i,j}|\nabla u|_{i,j}\right) \ \text{ and}$$
$$\partial_x(\kappa|\nabla u|)_{i-\frac{1}{2},j} = \tfrac{1}{h}\left(\kappa_{i,j}|\nabla u|_{i,j} - \kappa_{i-1,j}|\nabla u|_{i-1,j}\right).$$
$$\text{Here } |\nabla u|_{i,j} = \tfrac{1}{2h}\sqrt{(u_{i+1,j}-u_{i-1,j})^2 + (u_{i,j+1}-u_{i,j-1})^2 + 4h^2\varepsilon}$$

where a small regularization parameter $\varepsilon > 0$ is used to avoid division by zero when $|\nabla u|_{i,j}$ is in the denominator.

Partial Derivatives in y. By the *min-mod* of ∂_y's at two adjacent *whole* points

$$\partial_y(u_{i+\frac{1}{2},j}) = \textit{min-mod}\left(\tfrac{1}{2h}(u_{i+1,j+1}-u_{i+1,j-1}), \tfrac{1}{2h}(u_{i,j+1}-u_{i,j-1})\right),$$
$$\partial_y(u_{i-\frac{1}{2},j}) = \textit{min-mod}\left(\tfrac{1}{2h}(u_{i,j+1}-u_{i,j-1}), \tfrac{1}{2h}(u_{i-1,j+1}-u_{i-1,j-1})\right),$$

$$\partial_y(\kappa|\nabla u|)_{i+\frac{1}{2},j} = \textit{min-mod}\,(\alpha,\beta) \quad \text{with}$$
$$\alpha = \tfrac{1}{2h}\left(\kappa_{i+1,j+1}|\nabla u|_{i+1,j+1} - \kappa_{i+1,j-1}|\nabla u|_{i+1,j-1}\right) \quad \text{and}$$
$$\beta = \tfrac{1}{2h}\left(\kappa_{i,j+1}|\nabla u|_{i,j+1} - \kappa_{i,j-1}|\nabla u|_{i,j-1}\right).$$

$$\partial_y(\kappa|\nabla u|)_{i-\frac{1}{2},j} = \textit{min-mod}\,(\alpha,\beta) \quad \text{with}$$
$$\alpha = \tfrac{1}{2}\left(\kappa_{i,j+1}|\nabla u|_{i,j+1} - \kappa_{i,j-1}|\nabla u|_{i,j-1}\right) \quad \text{and}$$
$$\beta = \tfrac{1}{2}\left(\kappa_{i-1,j+1}|\nabla u|_{i-1,j+1} - \kappa_{i-1,j-1}|\nabla u|_{i-1,j-1}\right).$$

Here $|\nabla u|$ is approximated using
$$|\nabla u|_{i+\frac{1}{2},j} = \sqrt{(\partial_x(u_{i+\frac{1}{2},j}))^2 + (\partial_y(u_{i,j+\frac{1}{2}}))^2 + \varepsilon} \quad \text{and}$$
$$|\nabla u|_{i-\frac{1}{2},j} = \sqrt{(\partial_x(u_{i-\frac{1}{2},j}))^2 + (\partial_y(u_{i,j-\frac{1}{2}}))^2 + \varepsilon}.$$

Then by a similar procedure we can obtain approximations for $V^2_{i,j+\frac{1}{2}}$ and $V^2_{i,j-\frac{1}{2}}$. Finally, Neumann's boundary condition on $\partial\Omega$ is treated as

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

We remark that the inpainting domain D is mathematically understood as an open set, i.e., not including its boundary and therefore located away from $\partial\Omega$.

4 Review of numerical methods for the solution of the elastica model

Whilst the quality of reconstruction of the elastica model is out of discussion, its efficient numerical realization is still an open problem. In this section, we review the only two methods proposed so far for its solution: a time marching scheme from [14] and a texture-synthesis-based algorithm from [33].

4.1 An accelerated time marching method

An easy method of solving (3.2) is by solving its associated parabolic equation to steady-state

$$\frac{\partial u}{\partial t} = r(u), \qquad r(u) = \alpha\nabla\cdot V + \chi(z-u), \tag{4.1}$$

with initial condition $u(x,y,0) = z(x,y)$ and an explicit Euler method for the time derivative i.e. $u^{k+1}_{i,j} = u^k_{i,j} - \Delta t\, r(u^k_{i,j})$ with $k = 0, 1, \ldots$, and Δt the time-step. This numerical

scheme, accurate to $O(\Delta t)$, is expected to have a Courant-Friedrichs-Lewy (CFL) stability condition $\Delta t \sim O(h^4)$, see [39], for fourth order equations such as (3.2). In fact, for (3.2) due to its high nonlinearity, we have to use much smaller time-steps than h^4.

The idea of how to speed up a time marching scheme from [30] may be applied here by multiplying $r(u)$ by $|\nabla u|$ with the purpose of reducing stiffness [14] i.e.

$$\frac{\partial u}{\partial t} = |\nabla u| r(u), \tag{4.2}$$

which is further solved by an explicit Euler method. We will refer to this method as the accelerated time marching (ATM) method. Although (4.2) is better than (4.1), still a huge number of iterations is required to converge up to a prescribed accuracy. Hence, this method is not appropriate for large images. As an example, for the *circle problem* of Figure 3.2, the maximum stable time-step we were able to use was $\Delta t = 10^{-5}$ (using $\varepsilon = 10^{-2}$, $\lambda = 100$, $a = 1$, $b = 20$) for which convergence was obtained in 1.4×10^6 iterations and 20.7 hours of CPU-time for an image sized 32×32 pixels only. Here the inpainting domain D is a block of 12×12 pixels, initialized with random noise.

4.2 Texture-synthesis-based algorithm

The second method we review is an algorithm which approximates the solution of the elastica model by turning the underlying EL equation into a constrained combinatorial optimization problem [33]. In [33], the proposed algorithm is based on the assumption that the intensity values of missing pixels in D are likely to exist in the known region of the image $\Omega \backslash D$ and therefore some of these known values can be used to fill in a missing pixel. Similar to texture synthesis methods, the algorithm uses confidence and priority maps and proceeds from the boundary of the inpainting domain inward by filling in the missing pixels with the candidate that minimizes the elastica energy of a local neighborhood of the missing pixel. This process carries on until a steady-state solution is reached. We refer the interested reader to Ni's thesis [33] for a detailed explanation of this method.

In [33], it was reported that this algorithm is two orders of magnitude faster than the ATM scheme. Nevertheless, we see two disadvantages of this algorithm: 1) it is not robust to noisy levels; 2) since the obtained local solution is only a rough approximation to the real global solution of the elastica energy, it is likely to fail to deliver a good reconstruction for large D.

5 New numerical methods for the elastica model

We now proceed to present three unilevel, new and fast numerical schemes for the solution of (3.2). They are two unconditionally stable time marching (USTM) schemes and a fixed point method. USTM schemes for high order inpainting have already been tested for

the Cahn-Hilliard model [7] and very recently extended to other high order models in the excellent work of Schönlieb and Bertozzi [37] where a rigorous mathematical analysis of this technique is carried out. A given discrete time stepping algorithm is said to be *gradient stable* if the free energy $E(u)$ of the system is nonincreasing i.e. $E(u^{k+1}) \leq E(u^k)$ for any k. Therefore *unconditionally gradient stable* algorithms are those for which gradient stability holds for any size time step Δt .

5.1 USTM schemes

USTM schemes are designed to solve gradient systems of the form

$$\frac{\partial u}{\partial t} = -\nabla E(u), \;\; u(0) = u_0$$

where $u : \mathbb{R}^+ \to \mathbb{R}^p$ is in class C^1, $E(u) : \mathbb{R}^p \to \mathbb{R}$ is in class C^2, $\nabla E(u)$ is the gradient of $E(u)$ and the energy functional $E(u)$ satisfies the following conditions:

$$\begin{cases} E(u) > 0 & \forall\, u \in \mathbb{R}^p \\ E(u) \to \infty & \text{as } \|u\| \to \infty \\ \langle H(u)u, u\rangle \geq \mu & \forall\, u \in \mathbb{R}^p \end{cases} \tag{5.1}$$

where $\|u\| = \sqrt{\langle u, u\rangle}$ is the usual norm in $\mathbb{R}^p$ defined by the inner product $\langle \cdot, \cdot \rangle$, $H(u)$ is the Hessian matrix of $E(u)$ and $\mu \in \mathbb{R}$. We note that the numerical approximation of the elastica energy $J(u)$ satisfies (5.1) when using $|\nabla u|_\varepsilon = \sqrt{|\nabla u|^2 + \varepsilon}$.

The Eyre's technique [23, 24] called *convexity splitting* (CS) consists of first splitting the above given energy $E(u)$ into $E_1(u) - E_2(u)$ where both $E_1(u), E_2(u)$ are strictly convex and second evolving the gradient flow for the EL equation using a semi-implicit time stepping scheme in which the convex part of the energy is implicit and the concave part explicit. In simple words, the equation

$$\frac{\partial u}{\partial t} = -\nabla E(u) = -\nabla E_1(u) + \nabla E_2(u), \;\; u(0) = u_0$$

is solved in a semi-implicit way

$$\frac{u^{k+1} - u^k}{\Delta t} = -\nabla E_1(u^{k+1}) + \nabla E_2(u^k).$$

Under suitable conditions, this CS scheme leads to an unconditionally stable time-discretization scheme, allowing for arbitrarily large time steps, as previously applied in [7] for a different inpainting model based on the Cahn-Hilliard equation. Of course finding a suitable decomposition of the energy $E(u)$ is the essential part of the CS technique and usually requires a deep understanding of the energy under consideration.

5.1.1 USTM1

The first USTM scheme we present is a straightforward application of the Eyre's technique. Our contribution here is a working splitting of (3.1) leading to a USTM scheme.

The elastica energy (3.1) can be naturally divided (depending upon the norm) in two parts as $J(u) = J_1(u) + J_2(u)$ with

$$J_1(u) = \int_\Omega (a + b\kappa^2)|\nabla u| \; dxdy \quad \text{and} \quad J_2(u) = \frac{\lambda}{2}\int_{\Omega\setminus D} (u-z)^2 \; dxdy.$$

We note that $J_1(u)$ yields the gradient flow

$$\langle \frac{\partial u}{\partial t}, v\rangle_{TV} = -\langle \nabla J_1(u), v\rangle_{TV} \equiv -\frac{\partial}{\partial t} J_1(u+tv) \mid_{t=0}$$

where $-\frac{\partial}{\partial t} J_1(u+tv) \mid_{t=0} = -\nabla \cdot \left(\left(a + b\kappa^2\right) \frac{\nabla u}{|\nabla u|} - \frac{2b}{|\nabla u|^3} \nabla^\perp u \nabla(\kappa |\nabla u|) \nabla^\perp u \right)$ and $\langle \cdot, \cdot \rangle_{TV}$ denotes the inner product in the total variation space of functions. Similarly using the L^2 norm in $J_2(u)$ yields the gradient flow

$$\langle \frac{\partial u}{\partial t}, v\rangle_{L^2} = -\langle \nabla J_2(u), v\rangle_{L^2} \equiv -\frac{\partial}{\partial t} J_2(u+tv) \mid_{t=0}$$

where $-\frac{\partial}{\partial t} J_2(u+tv) \mid_{t=0} = -\lambda(u-z)$. However $J(u)$ by itself is not strictly a gradient flow under any norm. Nevertheless, as we shall show, we still can apply the CS method successfully. As remarked, for the Cahn-Hilliard inpainting model which is also not strictly a gradient flow, it was found that the CS method was successful [7].

To apply the CS method to here, we start by splitting $J_1(u) = J_{11}(u) - J_{12}(u)$ with

$$J_{11}(u) = a\int_\Omega |\nabla u| \; dxdy + C_1 \int_\Omega |\nabla u| \; dxdy \tag{5.2}$$

$$\text{and} \quad J_{12}(u) = -b\int_\Omega \kappa^2 |\nabla u| \; dxdy + C_1 \int_\Omega |\nabla u| \; dxdy, \tag{5.3}$$

and likewise we split $J_2(u) = J_{21}(u) - J_{22}(u)$ with

$$J_{21}(u) = \frac{C_2}{2}\int_{\Omega\setminus D} |u|^2 \; dxdy \quad \text{and} \tag{5.4}$$

$$J_{22}(u) = -\frac{\lambda}{2}\int_{\Omega\setminus D} (u-z)^2 \; dxdy + \frac{C_2}{2}\int_{\Omega\setminus D} |u|^2 \; dxdy. \tag{5.5}$$

Further, our USTM1 scheme is to solve

$$u^{k+1} = u^k + \Delta t \left(-\nabla(J_{11}^{k+1} - J_{12}^k) - \nabla(J_{21}^{k+1} - J_{22}^k) \right)$$

which leads to the numerical scheme

$$\frac{u^{k+1}-u^k}{\Delta t} = \left(\alpha \nabla \cdot \left((a + C_1) \frac{\nabla u^{k+1}}{|\nabla u^{k+1}|} \right) - C_2 u^{k+1} \right) +$$
$$\alpha \nabla \cdot \left(\frac{(b\kappa^2 - C_1)\nabla u^k}{|\nabla u^k|} - \frac{2b\nabla^\perp u \nabla(\kappa|\nabla u^k|)\nabla^\perp u^k}{|\nabla u^k|^3} \right) + \chi(z - u^k) + C_2 u^k \tag{5.6}$$

Here the positive and stabilizing constants C_1 and C_2 are needed to ensure that the energies $J_{11}(u)$ and $J_{21}(u)$ are convex. Experimentally we found that a good choice is $50 \leq C_1 \leq 200$ and $C_2 \sim \lambda$. Actually for convexity splitting to work on (3.1) we might not need to carry out the last splitting since $J_2(u)$ is already a convex functional; however it is beneficial to do so. Experimentally we found that if the splitting of $J_2(u)$ is not made effective, i.e. $C_2 = 0$, then C_1 needs to be increased to such large values that USTM1 would be much slower to converge even though the scheme USTM1 remains unconditionally stable for any C_1 and any Δt.

To solve (5.6) at each time-step, we construct a fixed-point type method by linearizing the term $|\nabla u|^{-1}$ on the left-hand side of (5.6). Therefore, after applying the discretization process (Section 3.2) we obtain the following linear system of equations

$$\begin{aligned} u_{i,j}^{k+1}\mathbb{S}_{i,j} - u_{i+1,j}^{k+1}\left(C_{i+\frac{1}{2},j}^{k}\right) - u_{i-1,j}^{k+1}\left(C_{i-\frac{1}{2},j}^{k}\right) - u_{i,j+1}^{k+1}\left(C_{i,j+\frac{1}{2}}^{k}\right) \\ -u_{i,j-1}^{k+1}\left(C_{i,j-\frac{1}{2}}^{k}\right) = f_{i,j}(u^k, \lambda, \Delta t, C_1, C_2) \end{aligned} \tag{5.7}$$

where

$$C_{(i+\frac{1}{2},j)}^{k} = \frac{\alpha\Delta t\ (a + C_1)}{h^2|\nabla u^k|_{(i+\frac{1}{2},j)}}$$

and so on, and

$$\mathbb{S}_{i,j} = (1 + \Delta t\ C_2) + C_{i+\frac{1}{2},j}^{k} + C_{i-\frac{1}{2},j}^{k} + C_{i,j+\frac{1}{2}}^{k} + C_{i,j-\frac{1}{2}}^{k}.$$

Then such a fixed point method amounts to solving (5.7) as a linear system

$$A(\boldsymbol{u}^k)\boldsymbol{u}^{k+1} = f(\boldsymbol{u}^k),$$

where $\boldsymbol{u}^k = [u_{1,1}^k, u_{2,1}^k \ldots, u_{n,1}^k, u_{1,2}^k, \ldots, u_{n,m}^k]$ and f contains all explicit terms in (5.6). In this case, $A(\boldsymbol{u}^k)$ is a sparse, symmetric and positive definite matrix.

The USTM1 scheme (5.6) is tested for a model problem, with comparative results with ATM shown in Table 5.1; it is evident that it is many times faster than the ATM.

5.1.2 USTM2

Convexity splitting guarantees unconditional stability of USTM1 by treating the expanding term of the elastica energy i.e. $-\nabla J_{12}(u)$ explicitly. In this section, we shall show that further partitioning

$$J_{12}(u) = J_{12}^{A}(u) + J_{12}^{B}(u) \tag{5.8}$$

in such a way that

$$J_{12}^{A}(u) = \int_{\Omega} g_1(u)g_2(u)\ dxdy \tag{5.9}$$

with $g_1(u) = -b\kappa^2$ and $g_2(u) = C_1|\nabla u|$ can be implicitly treated, allows us to treat $J_2(u)$ fully implicitly without losing the property of unconditional stability and to eliminate C_2 from the scheme. Numerical experiments showed that the resulting new scheme which we name as USTM2 is as fast as USTM1 to converge but more importantly there is no need to use a second parameter C_2.

Theorem 5.1. *If $J_{11}(u)$ and $J_{12}(u)$ are strictly convex functionals satisfying (5.2)-(5.3) and $J_{12}(u)$ satisfies (5.1) with $\langle H_{J_{12}}(u)u, u\rangle > \mu \geq 0$, i.e. all the eigenvalues of the Hessian matrix $H_{J_{12}}$ are strictly nonnegative values, then for any initial condition, and provided $C_1 > 0$ is sufficiently large, the numerical scheme*

$$\frac{u^{k+1} - u^k}{\Delta t} = \nabla J_{11}(u^{k+1}) - \nabla g_1(u^k)g_2(u^k) - g_1(u^{k+1})\nabla g_2(u^{k+1}) - \nabla J_{12}^B(u^k), \tag{5.10}$$

is gradient stable for all $\Delta t > 0$.

Proof. Assuming $J_1(u)$ satisfies (5.1) and expanding it about u^{k+1} up to second order using Taylor's theorem, the following inequality holds

$$\delta J \equiv J_1(u^{k+1}) - J_1(u^k) \leq \langle \nabla J_1(u^{k+1}), u^{k+1} - u^k\rangle + |\mu|\|u^{k+1} - u^k\|^2.$$

Now, first we replace $\nabla J_1(u^{k+1})$ above by $\nabla J_{11}(u^{k+1}) - \nabla J_{12}(u^{k+1})$ to get

$$\delta J \leq \langle \nabla J_{11}(u^{k+1}) - \nabla J_{12}(u^{k+1}), u^{k+1} - u^k\rangle + |\mu|\|u^{k+1} - u^k\|^2$$

and then add the term $-\langle \frac{1}{\Delta t}(u^{k+1} - u^k) + \nabla J_{11}(u^{k+1}) - \nabla g_1(u^k)g_2(u^k) - g_1(u^{k+1})\nabla g_2(u^{k+1}) - \nabla J_{12}^B(u^k), u^{k+1} - u^k\rangle$ from (5.10) which accounts for nothing but helps in the proof. This way,

$$\begin{aligned}\delta J \leq\ & \langle \nabla J_{11}(u^{k+1}) - \nabla J_{12}(u^{k+1}), u^{k+1} - u^k\rangle + |\mu|\|u^{k+1} - u^k\|^2 \\ & - \langle \frac{1}{\Delta t}(u^{k+1} - u^k) + \nabla J_{11}(u^{k+1}) - \nabla g_1(u^k)g_2(u^k) \\ & - g_1(u^{k+1})\nabla g_2(u^{k+1}) - \nabla J_{12}^B(u^k), u^{k+1} - u^k\rangle.\end{aligned}$$

It is immediate to see that the terms $\nabla J_{11}(u^{k+1})$ cancel out. The next step is to use $\nabla J_{12}(u^{k+1}) = \nabla J_{12}^A(u^{k+1}) + \nabla J_{12}^B(u^{k+1})$ and further $\nabla J_{12}^A(u^{k+1}) = \nabla g_1(u^{k+1})g_2(u^{k+1}) + g_1(u^{k+1})\nabla g_2(u^{k+1})$ which can be derived from (5.8) and (5.9)

to obtain

$$\begin{aligned}\delta J \leq & -\langle \nabla g_1(u^{k+1})g_2(u^{k+1}) + g_1(u^{k+1})\nabla g_2(u^{k+1}) + \nabla J_{12}^B(u^{k+1}), u^{k+1} - u^k\rangle \\ & + \langle \nabla g_1(u^k)g_2(u^k) + g_1(u^{k+1})\nabla g_2(u^{k+1}) + \nabla J_{12}^B(u^k), u^{k+1} - u^k\rangle \\ & + \left(|\mu| - \frac{1}{\Delta t}\right)\|u^{k+1} - u^k\|^2 \\ & = -\langle \nabla g_1(u^{k+1})g_2(u^{k+1}) - \nabla g_1(u^k)g_2(u^k), u^{k+1} - u^k\rangle \\ & - \langle \nabla J_{12}^B(u^{k+1}) - \nabla J_{12}^B(u^k), u^{k+1} - u^k\rangle + \left(|\mu| - \frac{1}{\Delta t}\right)\|u^{k+1} - u^k\|^2\end{aligned}$$

after cancelation of some terms. Now from the convexity of $J_{12}^B(u) = C_1 \int_\Omega |\nabla u|\ dxdy$ there exists $\hat{\mu}$ such that $\langle \nabla J_{12}^B(u^{k+1}) - \nabla J_{12}^B(u^k), u^{k+1} - u^k\rangle \geq \hat{\mu}(C_1)\|u^{k+1} - u^k\|^2$ where $\hat{\mu}(C_1)$ indicates that $\hat{\mu}$ depends on C_1. Hence we have that

$$\begin{aligned}\delta J \leq & -\langle g_1(u^{k+1})\nabla g_2(u^{k+1}) - g_2(u^k)\nabla g_1(u^k), u^{k+1} - u^k\rangle \\ & + \left(|\mu| - |\hat{\mu}(C_1)| - \frac{1}{\Delta t}\right)\|u^{k+1} - u^k\|^2 \leq 0,\end{aligned}$$

where the last inequality is satisfied provided C_1 is chosen large enough. □

Using Theorem 5.1 we obtain our USTM2 scheme

$$\begin{aligned}& u^{k+1} - \Delta t\left(\alpha\nabla\cdot\left((a + C_1 + b\kappa^2)\frac{\nabla u^{k+1}}{|\nabla u^{k+1}|}\right) + \chi u^{k+1}\right) \\ & = u^k + \Delta t\left(\alpha\nabla\cdot\left(-C_1\frac{\nabla u^k}{|\nabla u^k|} - \frac{2b}{|\nabla u^k|^3}\nabla^\perp u\nabla(\kappa|\nabla u^k|)\nabla^\perp u^k\right) + \chi z\right). \qquad (5.11)\end{aligned}$$

We remark that USTM2 is only slightly faster than USTM1 as can be seen from Table 5.1 if the latter is supplied with an optimized C_2; however in other cases USTM2 works fine while USTM1 having a wrong parameter C_2 slows down its convergence or is even divergent.

5.2 A working fixed point method

The main virtue of the above two USTM schemes is that the value of Δt is not an issue due to the unconditional stability property they share. Of course selecting very small Δt will yield a very slow to converge algorithm and selecting very large Δt will be prohibited if the accuracy of the solution needs to be bounded at every iteration, see for instance [25, 38, 41]. For the elastica formulation (3.1) the aim is to find its minimum, therefore the value of Δt is less relevant since only the accuracy of the final solution is important and not at every iteration. This observation prompts us to construct a fixed point method from

(5.11) for a steady state solution (i.e. taking an infinity time-step Δt). This suggests to solve

$$\begin{aligned} &-\alpha\nabla\cdot\left((a+C_1+b\kappa^2)\frac{\nabla u^{k+1}}{|\nabla u^{k+1}|}\right)+\chi u^{k+1}\\ &=\alpha\nabla\cdot\left(-C_1\frac{\nabla u^k}{|\nabla u^k|}-\frac{2b}{|\nabla u^k|^3}\nabla^\perp u\nabla(\kappa|\nabla u^k|)\nabla^\perp u^k\right)+\chi z. \end{aligned} \tag{5.12}$$

At the present time a rigorous mathematical proof of the convergence of this fixed point scheme is not at hand although we expect this to be true since (5.12) inherits this property from the USTM2 scheme. Nonetheless numerical experiments over a wide range of problems have shown (5.12) to be a convergent and fast algorithm provided C_1 is properly selected.

To solve (5.12), we linearize the left-hand side and obtain a linear system of equations:

$$\begin{aligned} u_{i,j}^{k+1}\mathbb{S}_{i,j}-u_{i+1,j}^{k+1}\left(C_{i+\frac{1}{2},j}^k\right)-u_{i-1,j}^{k+1}\left(C_{i-\frac{1}{2},j}^k\right)-u_{i,j+1}^{k+1}\left(C_{i,j+\frac{1}{2}}^k\right)\\ -u_{i,j-1}^{k+1}\left(C_{i,j-\frac{1}{2}}^k\right)=f_{i,j}(u^k,\lambda,C_1) \end{aligned} \tag{5.13}$$

where $C_{(i+\frac{1}{2},j)}^k=\frac{\alpha(a+C_1+b\kappa^2)}{h^2|\nabla u^k|_{(i+\frac{1}{2},j)}}$ and so on, and $\mathbb{S}_{i,j}=\chi+C_{i+\frac{1}{2},j}^k+C_{i-\frac{1}{2},j}^k+C_{i,j+\frac{1}{2}}^k+C_{i,j-\frac{1}{2}}^k$. Note (5.13), also representable by $A(\boldsymbol{u}^k)\boldsymbol{u}^{k+1}=f(\boldsymbol{u}^k)$, has similar representation to (5.7). Now since the new matrix $A(\boldsymbol{u}^k)$ is semi-positive definite and weakly diagonally dominant, we use a lexicographic Gauss-Seidel (GSLEX) method to partially solve the system and call this fixed point method based on GSLEX inner iterations the FPGS algorithm.

A remark is due here. Re-write (5.12) as

$$-\alpha\left(\nabla\cdot\mathcal{N}^{k+1}+C_1\mathcal{TV}(u^{k+1})\right)+\chi u^{k+1}=\alpha\left(\nabla\cdot\mathcal{T}^k+C_1\mathcal{TV}(u^k)\right)+\chi z, \tag{5.14}$$

after defining $\mathcal{TV}(u)\equiv\nabla\cdot\frac{\nabla u}{|\nabla u|}$, and splitting $V=\mathcal{N}+\mathcal{T}$ further (along normal and tangent directions) with $\mathcal{N}=(a+b\kappa^2)\frac{\nabla u}{|\nabla u|}$ and $\mathcal{T}=-\frac{2b}{|\nabla u|^3}\nabla^\perp u\nabla(\kappa|\nabla u|)\nabla^\perp u$. Then taking $C_1=0$ yields the natural fixed point for (3.2) in the spirit of [4, 20, 42]. However, experimental tests showed that this scheme with $C_1=0$ simply does not work. Possible refinements (still with $C_1=0$) from taking some part from $\nabla\cdot\mathcal{N}$ to the right-hand side or taking some contribution from $\nabla\cdot\mathcal{T}$ into $A(\boldsymbol{u}^k)$ failed to make any improvement. Our experiments showed that the first option is not helpful, and the second is not much better because diagonal dominance cannot be guaranteed due to $\nabla(\kappa|\nabla u|)$ changing its sign and consequently $A(\boldsymbol{u}^k)$ may not be even weakly diagonally dominant. One explanation of the failure of (5.14) when $C_1=0$ may be that the domain of convergence of the scheme is very small and therefore it fails to converge for a bad initial guess. On the other hand, when the positive constant C_1 is large enough , the $\mathcal{TV}$ term drives to convergence the algorithm when $\|u^{k+1}-u^k\|$ is large (at the beginning of iterations), however once $\|u^{k+1}-u^k\|$

Method	# iterations	CPU	PSNR
ATM	1.4×10^6	$74,578$	55
USTM1	6,500	120	60
USTM2	6,000	112	60
FPGS	3,500	73	60

Table 5.1: Performance of the two algorithms for the circle problem of Figure 3.2 with $m = n = 32$. The following parameters were used for each one of them $a = 1$, $b = 20$, $\beta = 10^{-2}$, $\lambda = 100$. For ATM $\Delta t = 10^{-5}$ and for USTM $\Delta t = 1$. Finally, 10 GS steps were used in FPGS.

starts decreasing then the $\mathcal{TV}$ terms on both sides of (5.14) tend to cancel each other and gradually (5.14) with $C_1 = 0$ takes over.

5.3 Comparisons

In Table 5.1 we present a performance summary of the three unilevel algorithms we have presented so far (USTM1, USTM2, FPGS) against the ATM method. Here the Peak-Signal-to-Noise-Ratio (PSNR) between images u and u^0 of size $m \times n$ is defined as

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

for a model problem where the true image u^0 is known and u is the restored image. The larger a PSNR is, the better the restored image is. Clearly, even for this small image of 32×32, the FPGS algorithm is many times faster than ATM. However it still can take long time to inpaint large images. As an optimal algorithm, the multigrid method has proved to be successful in solving a number of different image processing problems; see [4, 9, 20, 35, 36]. In the following, first we shall show that FPGS is a good smoother and then we shall proceed to develop a multigrid method for the elastica model (3.2).

6 Local Fourier analysis for the fixed point method

To construct a convergent multigrid (MG) method for a nonlinear problem such as the elastica PDE (3.2), the key is to find a good smoother and this is by no means trivial. For an iterative algorithm to be a good smoother, it needs to reduce the high frequencies of the underlying error. To evaluate this, a very useful tool is the local Fourier analysis (LFA); see [40]. Theoretically LFA is designed to study linear problems with constant coefficients on an infinite grid. Regardless of this strong limitation, by extending the grid to an infinite grid to neglect boundary conditions and locally linearizing the discrete operator, LFA is still

a recommended tool [1, 8, 28, 43] for analysis of discrete nonlinear operators. Examples of image processing problems where LFA was used for this purpose can be found in [4, 9, 13]. Hence in this section we use LFA to show that the linearized fixed point algorithm (5.13) certainly reduces the high frequency components with a good rate and therefore is a suitable smoother for a MG method.

For simplicity we consider the case of a square image of size $m \times m$. Let k represent the superscript for the outer iterations and p the same for the inner (GSLEX) ones of the FPGS scheme respectively. Let the inner local error functions $e_{i,j}^{p+1}$ and $e_{i,j}^{p}$ be defined as $e_{i,j}^{p+1} = \bar{u}_{i,j} - u_{i,j}^{p+1}$ and $e_{i,j}^{p} = \bar{u}_{i,j} - u_{i,j}^{p}$ where $\bar{u}_{i,j}$ is the exact solution at grid point (i, j). LFA involves expanding

$$e_{i,j}^{(p+1)} = \sum_{\theta_1,\theta_2=-m/2}^{m/2} \psi_{\theta_1,\theta_2}^{p+1} B_{\theta_1,\theta_2}(x_i, y_j), \qquad e_{i,j}^{(p)} = \sum_{\theta_1,\theta_2=-m/2}^{m/2} \psi_{\theta_1,\theta_2}^{p} B_{\theta_1,\theta_2}(x_i, y_j)$$

in Fourier components $B_{\theta_1,\theta_2}(x_i, y_j)$, with $\alpha_1 = 2\theta_1\pi/m$, $\alpha_2 = 2\theta_2\pi/m \in [-\pi, \pi]$, defined as

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

Since the local inner iterations of (5.13) can be represented by (for $p \geq 0$)

$$\frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ -C^k_{i-\frac{1}{2},j} & \mathbb{S}^k_{i,j} & 0 \\ 0 & -C^k_{i,j+\frac{1}{2}} & 0 \end{bmatrix} u^{p+1} = f^k_{i,j} - \frac{1}{h^2}\begin{bmatrix} 0 & -C^k_{i,j-\frac{1}{2}} & 0 \\ 0 & 0 & -C^k_{i+\frac{1}{2},j} \\ 0 & 0 & 0 \end{bmatrix} u^{p},$$

we have that

$$\begin{gathered} -(\chi + C_{i+\frac{1}{2},j} + C_{i-\frac{1}{2},j} + C_{i,j+\frac{1}{2}} + C_{i,j-\frac{1}{2}}) e_{i,j}^{p+1} + C_{i+\frac{1}{2},j}\, e_{i+1,j}^{p} + C_{i-\frac{1}{2},j}\, e_{i-1,j}^{p+1} \\ + C_{i,j+\frac{1}{2}}\, e_{i,j+1}^{p} + C_{i,j-\frac{1}{2},j}\, e_{i,j-1}^{p+1} = 0. \end{gathered}$$

Therefore the local amplification factor for $\mu_{i,j} = \left|\psi_{\theta_1,\theta_2}^{p+1}/\psi_{\theta_1,\theta_2}^{p}\right| = \mu_{i,j}(\alpha_1, \alpha_2)$ is defined by

$$\mu_{i,j} = \frac{\left|C_{i+\frac{1}{2},j}\, e^{\mathbf{i}\alpha_1} + C_{i,j+\frac{1}{2}}\, e^{\mathbf{i}\alpha_2}\right|}{\left|\chi + C_{i+\frac{1}{2},j} + C_{i-\frac{1}{2},j} + C_{i,j+\frac{1}{2}} + C_{i,j-\frac{1}{2}} - C_{i-\frac{1}{2},j}\, e^{-\mathbf{i}\alpha_1} - C_{i,j-\frac{1}{2}}\, e^{-\mathbf{i}\alpha_2}\right|}$$

and the maximum rate at this kth step $\bar{\mu}_{i,j} = \max_{\alpha_1,\alpha_2} \mu_{i,j}(\alpha_1, \alpha_2)$ in the high frequency range $(\alpha_1, \alpha_2) \in [-\pi, \pi] \setminus [-\pi/2, \pi/2]$. Note that at the kth outer step, the nonlinear coefficients $C_{(\cdot,\cdot)}$ remain frozen (fixed) across all inner iterations.

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) grid point. As done in [9], in order to evaluate effectiveness,

we evaluate the accumulated rate based on consecutive smoothing rates $\bar{\mu}_{i,j}$. That is to say, suppose we have completed K (accumulated) inner relaxation steps. Let $\bar{M}_k$ denote the corresponding rate matrix (for $1 \leq k \leq K$); then define

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

as the accumulated smoothing rate of a relaxation step (over K accumulated inner iterations). This makes perfect sense when the coefficients are constant which happens to be true towards the end of convergence of the algorithm.

To illustrate the rate for an example, we present in Table 6.2 the smoothing rates $\bar{\mu}$ and accumulated smoothing rates $\hat{\mu}_K$ inside the inpainting domain D and outside it in $\Omega \backslash D$, computed for five outer iterations for the circle problem of Figure 3.2 with 20% of additive Gaussian noise. Clearly, the accumulated rates for FPGS are quite good for $K \leq 50$.

	D		$\Omega \backslash D$	
Outer iterations ν	$\bar{\mu}$	$\hat{\mu}_K$	$\bar{\mu}$	$\hat{\mu}_K$
1	0.6267	0.6267	0.8850	0.8850
2	0.7049	0.4417	0.8882	0.7861
3	0.7559	0.3339	0.8680	0.6823
4	0.8220	0.2745	0.8582	0.5856
5	0.9115	0.2502	0.8445	0.4945

Table 6.2: Illustration of smoothing rates for the FPGS smoother. (Note $K = gsiter\ \nu$, where $gsiter$ is the number of inner iterations). Here $a = 1$, $b = 20$ and $gsiter = 10$ were used.

7 A nonlinear multigrid for Euler's elastica model

Here we shall show how to implement a multigrid algorithm for the elastica formulation (3.2). To begin with, we denote by $(Nu)_{i,j} = \chi z_{i,j}$ the nonlinear operator equation

$$-\alpha\nabla \cdot \left(\frac{\left(a + b\kappa_{i,j}^2\right) \nabla u_{i,j}}{|\nabla u|_{i,j}} - \frac{2b\nabla^{\perp} u_{i,j} \nabla(\kappa_{i,j}|\nabla u|_{i,j})\nabla^{\perp} u_{i,j}}{|\nabla u|_{i,j}^3} \right) + \chi u_{i,j} = \chi z_{i,j},$$

which will be approximated on grids of different sizes. We will denote by $N_h u_h = \chi z_h$ the discrete equation 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 the above equation on the cell-centered grid Ω_{2h} with grid spacing $2h$. Likewise we can generate a sequence of L coarse levels $2h, 4h, 8h, \ldots, 2^L h$.

7.1 The MG components

We briefly discuss how we choose the three main components of a multigrid algorithm. For the restriction and interpolation operators R_h^{2h} and I_{2h}^h respectively, full weighting (FW) and bilinear interpolation operators for cell-centered grids are used; see [40] for details. No difference in convergence was noticed by using higher order operators like bi-cubic interpolation.

To coarsen the interfaces, which is unique for inpainting problems, we use the same method as described in [9]. Briefly we represent the inpainting domain by a binary mask M_h and coarsen this mask similarly to grid coarsening.

As stated before, our chosen smoother is the FPGS algorithm (Section 5.2), that is, (5.13). It is stated in Algorithm 7.1.

Algorithm 7.1 [FPGS Smoother] $\mathbf{u}_h \leftarrow FPGS(\mathbf{u}_h, z_h, gsiter, M_h)$

1: Choose an initial guess $\mathbf{u}_h^0$ for (5.13)
2: **for** $k = 1$ to $gsiter$ **do**
3: Apply $gsiter$ Gauss Seidel iterations to the linear system $A_h(\mathbf{u}_h^k)\mathbf{u}_h^{k+1} = \mathbf{z}_h$
4: **end for**

7.2 The MG algorithm

Multigrid schemes are designed to obtain fast solution of PDE's similar to (3.2). In particular when the PDE to solve is nonlinear as with here, the full approximation scheme (FAS) [40] is highly efficient. This FAS method which we have adapted for the inpainting case and is described in Algorithm 7.3, operates a hierarchy of discretization levels where at each level the error equation is partially solved or smoothed (step 2) and the new approximation transported to next coarser level (step 3). This process is recursively applied until reaching the coarsest level where exact but computationally cheap solution is obtained (step 1). Then the process moves backwards on the hierarchical structure transporting the more accurate error (step 7) and updating the approximate solution at each level (step 8), then taking this new approximate solution as initial guess and smoothing again (step 9) repeating the process until reaching the finest level again. With standard coarsening the number of variables is halved at each coarse level in each spatial dimension.

Now we state our V-cycling nonlinear MG in Algorithm 7.2, meaning that just one recursive call to the algorithm is made on each level to approximately solve a coarse grid problem. Here *gsiter* represents the number of inner Gauss-Seidel iterations at each pre or post-smoothing FPGS step. Finally, we remark that when implementing the MG method, the parameter λ and the balance at coarse levels between $\mathcal{N}$ and $\mathcal{T}$ defined by the parameters a and b need to be kept the same.

Algorithm 7.2 [Nonlinear Multigrid Method]

1: Select an initial guess u_h on the finest grid h
2: Set $k = 0$ and $err = tol + 1$
3: **while** $err < tol$ **do**
4: $\quad u_h^{k+1} \leftarrow FAS(u_h^k, N_h^k, z_h, M_h, \nu_0, \nu_1, \nu_2, gsiter)$
5: $\quad err = \|u_h^{k+1} - u_h^k\|_2, \quad k = k + 1$
6: **end while**

Algorithm 7.3 [FAS] $\quad u_h \leftarrow FAS(h, u_h, N_h, z_h, M_h, \nu_0, \nu_1, \nu_2, gsiter)$

1: If $\Omega^h =$ coarsest grid, solve $N_h u_h = \chi z_h$ accurately (i.e. ν_0 iterations by FPGS) and return Else continue with step 2.
2: Pre-smoothing: Do ν_1 steps of $\quad u_h \leftarrow FPGS(u_h, z_h, gsiter, M_h)$
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: Update the right hand side $\chi z_{2h} \leftarrow R_h^{2h}(\chi z_h - N_h u_h) + N_{2h} u_{2h}$
6: Solve $N_{2h} u_{2h} = \chi z_{2h}$ by implementing
$u_{2h} \leftarrow FAS_{2h}(2h, u_{2h}, N_{2h}, z_{2h}, M_{2h}, \nu_0, \nu_1, \nu_2, gsiter)$
7: Compute the error $e_{2h} = u_{2h} - \bar{u}_{2h}$ and move it back to the next grid by $e_h \leftarrow I_{2h}^h e_{2h}$
8: Add the residual correction, $u_h \leftarrow u_h + e_{2h}$
9: Post-smoothing: Do ν_2 steps of $\quad u_h \leftarrow FPGS(u_h, z_h, gsiter, M_h)$

7.3 Full Multigrid

In order to provide an automatic and yet suitable initial guess, we shall adopt the Full Multigrid method (FMG) as described in [19, 40] which is based on the idea of nested iteration. That is, given the coarse grid Ω_H, one can apply a multigrid cycle (say a V-cycle) to obtain an approximate solution u_H at this level, then this u_H is interpolated to the next finer grid $\Omega_{H/2}$ to be used as an initial guess for another multigrid cycle to solve for $u_{H/2}$ at this fine level. This process is repeated until reaching the finest level.

Notice that in FMG the solution u_H and not the error e_H which is interpolated to the next finer level. Usually the operator used to interpolate the solution is denoted by $\Pi_{H/2}^H$ and is of higher accuracy (third order in our Algorithm 7.4) than the interpolation operators used within the multigrid iteration. Here u_h^{FMG} denotes the resulting FMG approximation on grid Ω_h. As expected, much better (faster) results are obtained as can be seen from the last two columns of Table 8.3 from testing all four inpainting problems.

Algorithm 7.4 Full Multigrid

On the coarsest Ω_H with mesh size $H = 2^L$,
Solve $N_H u_H = z_H$, providing $u_H^{FMG} = u_H$.
for $\ell = 1$ to L **do**
  $u_h^0 \leftarrow \Pi_H^h u_H^{FMG}$ with $h = H/2$
  $u_h^{FMG} \leftarrow FAS(h, u_h^0, N_h, z_h, M_h, \nu_0, \nu_1, \nu_2, gsiter)$.
  Set $H = h$
end for

8 Numerical results

In this section, we test the performance of Algorithm 7.2 to inpaint four problems. The problems and the MG restored results are shown in Figures 10.5, 10.6, 10.7 and 10.8 for $m = 256$.

Quality of reconstruction. Clearly from Figures 10.5-10.8, the restoration obtained with Algorithm 7.2 is visually pleasing in all of them. For instance, Figure 10.5 shows a very good reconstruction of the missing edges in the ear, nose and cheek of the girl. In Figure 10.6 we observe the reconnection of the thin piece of hair initially occluded by the letter "G" and in general the fair recovering of the geometrical structure of the missing regions. Figures 10.7 and 10.8 show the smooth reconstruction of curvy missing regions and in particular the latter illustrates the virtue of the elastica model in denoising and inpainting at the same time; a feature only shared with the TV model.

For completeness, in Table 8.3 we present the PSNR values obtained from the restored images. In the first two problems, the high PSNR values indicate good reconstructions. In the last two, they are not that high due to texture not recovered (requiring a new model) in the first case of grapes problem and strong noise present in the second one of noisy circles.

In other experiments, we have compared with the TV model using the code from [21]. For problems with a long and thin inpaiting domain D (e.g. Figure 10.7), the TV model gives a good restoration but for other problems with large D (e.g. Figure 10.5) the TV model cannot inpaint at all; see [9, 14] for more discussions on the weakness of the TV model for inpainting.

Speed comparison to other algorithms. We have already shown in Table 5.1 how slow the ATM can be (even for a small image), taking not thousands but millions of iterations to converge. This kind of slow convergence suggests that unilevel algorithms are barely enough for inpainting small images and definitely not suitable for large ones. A quick review of Table 5.1 also reveals that our FPGS as a standalone method is two orders of magnitude faster than ATM for the circle problem and this speedup was confirmed through other experiments and different problems we tested. On the other hand, we present the results obtained with MG in Table 8.3, from which it can be seen that our MG can be used

to obtain very fast inpaintings for large images.

Problem	Image Size	MG		FMG		PSNR
		MG cycles	CPU	MG cycles	CPU	
Child	128×128	10	171	2	27	91
	256×256	7	400	2	130	90
	512×512	6	1538	2	613	90
Lena	128×128	7	59	2	23	95
	256×256	6	268	2	103	94
	512×512	5	998	2	502	96
Grapes (*)	128×128	7	55	2	22	70
	256×256	5	219	2	101	70
	512×512	5	972	2	440	71
Noisy Circles	128×128	6	125	2	57	70
	256×256	4	356	2	193	71
	512×512	4	1514	2	851	70

Table 8.3: MG results and further improvements by the FMG. (*) results from inpainting the first (red) channel of the color image.

Finally, we remark on comparisons to the texture-synthesis-based method [33]. First while [33] claims a speed up of two orders of magnitude over ATM, our FMG can accelerate convergence by three orders of magnitude over ATM so our method is faster. Further, the method of [33] is not robust to inpaint noisy images (while our method is) and only finds an approximate local solution to the problem. Our method finds a global solution which guarantees a true minimization of (3.1) for large inpainting domains.

9 Generalization to other inpainting models

Finally we briefly comment on the generalization of our numerical algorithms. In the literature, there are two closely related works to our studied elastica model, namely, the filling-in and disocclusion model of Ballester, Bertalmio, Caselles, Sapiro and Verdera proposed in [2] and a later model by Ballester, Caselles and Verdera in [3]. For convenience we shall refer to the model of [2] as the BBCSV model and to the model of [3] as BCV. These two models differ from the elastica model in that they diffuse a vector field and gray levels at the same time within the missing regions.

In the BBCSV model, the functional

$$\begin{gathered}\min_{u,\theta}\left\{\int_D (a+b|\nabla K * u|)|\nabla\cdot\theta|^p\,dxdy\right\}\\ \theta\le 1,\ \ \|u\|\le M,\\ |\nabla u|-\nabla u\cdot\theta=0\ \text{ in }\ D,\\ u=z\ \text{ in } B\ \text{ (a band surrounding } D),\\ \theta\cdot\nu^D|_{\partial D}=\theta_0\cdot\nu^D|_{\partial D}\end{gathered}\tag{9.1}$$

where a, b, p, D are defined as before, K a convolution kernel, $M = \|z\|_{L^\infty(B)}$, θ_0 is any vector field satisfying $\theta_0\cdot\nabla z = |\nabla z|$ and ν^D denotes the outer unit normal to D, is minimized by evolving in time a system of PDE's of order three.

In [3], the authors relaxed some conditions and introduced new terms proposing the BCV model which reads

$$\begin{aligned}\min_{u,\theta}\Bigg\{&\int_D (a+b|\nabla K*u|)\,|\nabla\cdot\theta|^p\ dxdy+\alpha\int_D|\nabla u|\,dxdy\\ &-\alpha\int_{\partial D} g_0 u+\gamma\int_B |u-z|^q\,dxdy\Bigg\},\end{aligned}\tag{9.2}$$

$$\begin{gathered}\theta\le 1,\ \ |\nabla u|-\nabla u\cdot\theta=0\ \text{ in }\ D,\\ \theta\cdot\nu^D|_{\partial D}=g_0,\end{gathered}\tag{9.3}$$

where $g_0=\theta_0\cdot\nu^D|_{\partial D}$, $\alpha,\gamma>0$, $q\ge 1$. Then using an elegant approach they proved the convergence of minima of BCV to minima of the functional

$$\begin{aligned}\min_u\int_D (a+b|\nabla K*u|)\left|\nabla\cdot\left(\frac{\nabla u}{\varepsilon^2+|\nabla u|}\right)\right|^p\ dxdy+\alpha\int_D|\nabla u|\,dxdy\\ -\alpha\int_{\partial D} g_0 u+\gamma\int_B|u-z|^q\,dxdy,\\ \frac{\nabla u}{\varepsilon^2+|\nabla u|}\cdot\nu^D=\frac{\nabla z}{\varepsilon^2+|\nabla z|}\cdot\nu^D,\end{aligned}\tag{9.4}$$

as $\varepsilon\to 0$ establishing a connection between the numerical approaches of elastica [14] and BCV models. The major difference between them may be that BCV is equipped with a relatively fast semi-implicit steepest descent method while for elastica only the very slow ATM has been reported for its solution [14]. Here we shall show how easily our numerical methods can be adapted to minimize the BCV energy. This is, we show a feasible stabilized fixed point method (FPGS-type) for the equivalent of the BCV energy (9.4) for the case $p = q = 2$. Thus our FPGS-type algorithm for BCV reads

$$-\alpha\nabla\cdot\left(D_1(u^k)\frac{\nabla u^{k+1}}{|\nabla u^{k+1}|}\right)+\chi u^{k+1}=f(u^k,\alpha),\tag{9.5}$$

where

$$D_1(u) = C_1 + a + b\kappa^2 + \frac{\nabla u \cdot \nabla(2\kappa)}{|\nabla u|^3},$$

$$f(u,\alpha) = \alpha\nabla \cdot \left(-C_1 \frac{\nabla u}{|\nabla u|} - \frac{\nabla(2\kappa)}{|\nabla u|} - \frac{2b}{|\nabla u|^3} \nabla^{\perp} u \nabla(\kappa|\nabla u|) \nabla^{\perp} u \right) + \chi z.$$

This scheme was successfully tested for a number of problems. For instance, the test problem in Figure 3.2 was solved in only $3,700$ iterations which is comparable to that obtained for the elastica model for the same problem and this fast convergence is also observed in other tests. The quality of reconstruction between both models is also very similar and no visual difference between the outcomes from elastica and BCV models was appreciated. A similar scheme can be developed and used to solve the BBCSV model.

10 Conclusions

Euler's elastica inpainting model is attractive due to its ability of reconnecting far apart parts of broken objects and recovering the curvature of the missing parts of objects. So far, the lack of a fast numerical algorithm for this model represented a strong limitation for the range of its applications and wider use. In this paper we first introduced three fast unilevel numerical algorithms (USTM1, USTM2 and FPGS) than the existing accelerated time marching method (ATM). Then adopting our FPGS as a smoother, we were able to develop a fast and efficient nonlinear MG algorithm for solving this Euler's elastica model. A local Fourier analysis was done to demonstrate the effectiveness of our smoother. Numerical results confirmed that our multigrid method is very efficient. For our related work of multigrid methods for curvature denoising, refer to [10, 11].

References

[1] R.E. Alcouffe, A. Brandt, J.E. Dendy and J.W. Painter, *The multigrid method for the diffusion equation with strongly discontinuous coefficients*, SIAM J. Sci. Stat. Comput. **2**(4) (1981), 430–454.

[2] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro and J. Verdera, *Filling-in joint interpolation of vector fields and gray levels*, IEEE Trans. Image Process., **10**(8) (2001), 1200–1211.

[3] C. Ballester, V. Caselles, and J. Verdera, *Disocclusion by joint interpolation of vector fields and gray levels*, Multiscale Model. Simul., **2**(1) (2003), 80–123.

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

[5] 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.

[6] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, *Image inpainting*, in: SIGGRAPH 2000, Computer Graphics Proceedings, ed. K. Akeley, pp.417–424. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

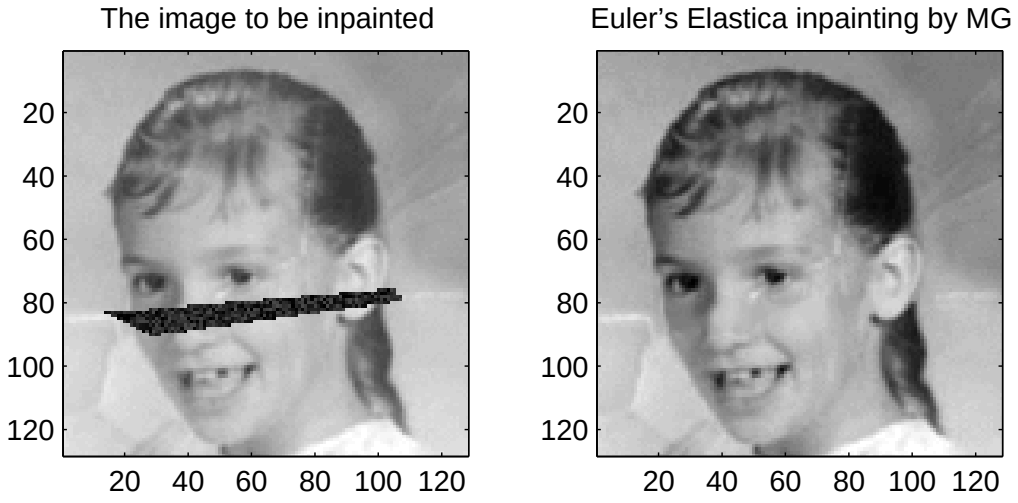


Figure 10.5: Example of reconnection of contours along large distances and inpainting of large domains by MG.

[7] A. L. Bertozzi, S. Esedoglu, and A. Gillet, *Inpainting of binary images using the Cahn-Hilliard equation*, IEEE Trans. Image Process., **16**(1) (2007), 285–291.
[8] A. Brandt, *Multi-level adaptive solutions to BVPs*, Math. Comput. **31**(138) (1977), 333–390.
[9] C. Brito-Loeza and K. Chen, *Multigrid method for a modified curvature driven diffusion model for image inpainting*, J. Comput. Math., **26**(6) (2008), 856–875.
[10] C. Brito-Loeza and K. Chen, *On high-order denoising models and fast algorithms for vector-valued images*. IEEE Trans. Image Process., **19**(6), (2010), to appear.
[11] C. Brito-Loeza and K. Chen, *Multigrid algorithm for high order denoising*, SIAM J. Image Sciences, (2010), to appear.
[12] V. Caselles, J.M. Morel, and C. Sbert, *An axiomatic approach to image interpolation*, IEEE Trans. Image Process., **7** (1998), 376–386.
[13] T. F. Chan, K. Chen and J. L. Carter, *Iterative methods for solving the dual formulation arising from image restoration*, Electron. Trans. Numer. Anal., **26** (2007), 299–311.
[14] T. F. Chan, S. H. Kang, and J. Shen, *Euler's elastica and curvature based inpaintings*, SIAM J. Appl. Math. **63**(2) (2002), 564–592.
[15] T. F. Chan and J. Shen, *Non-texture inpaintings by curvature-driven diffusions*, J. Visual Commun. Image Rep. **12**(4) (2001), 436–449.
[16] T. F. Chan and J. Shen, *Mathematical models for local nontexture inpaintings*, SIAM J. Appl. Math. **62**(3) (2002), 1019–1043.
[17] T. F. Chan and J. Shen, *Image Processing and Analysis – Variational, PDE, Wavelet, and stochastic Methods*, SIAM, Philadelphia, PA, USA, 2005.

Figure 10.6: Text removal example. Note the smooth reconnection of the broken piece of hair occluded by the letter G. The size of the gap is more than twice the size of the hair.

[18] T. F. Chan, J. Shen, and H. Zhou, *Total variation wavelet inpainting*, J. Math. Imaging Vis. **25** (2006), 107–125.

[19] K. Chen, *Matrix Preconditioning Techniques and Applications*, Series: Cambridge Monographs on Applied and Computational Mathematics (No. 19), Cambridge University Press, UK, 2005.

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

[21] J. Dahl, P.C. Hansen, S.H. Jensen and T.L. Jensen, *Algorithms and software for total variation image reconstruction via first-order methods*, Numer. Algor., **53**, (2010), 67–92.

[22] S. Esedoglu and J. Shen, *Digital inpainting based on the Mumford-Shah-Euler image model*, European J. Appl. Math. **13** (2002), 353–370.

[23] D. J. Eyre, *Unconditionally gradient stable time marching the Cahn-Hilliard equation*, Computational and Mathematical Models of Microstructural Evolution, **53** (1998), 1686–1712.

[24] D. J. Eyre, *An unconditionally stable one-step scheme for gradient systems*, Unpublished article (1998).

[25] J. B. Greer, A. L. Bertozzi, and Guillermo Sapiro, *Fourth order partial differential equations on general geometries*, J. Comput. Phys. **216**(1) (2006), 216–246.

[26] 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.

[27] J. Gu, L. Zhang, G. Yu, Y. Xing, and Z. Chen, *X-ray CT metal artifacts reduction through curvature based sinogram inpainting*, J. X-Ray Sci. Tech. **14**(2) (2006), 73–82.

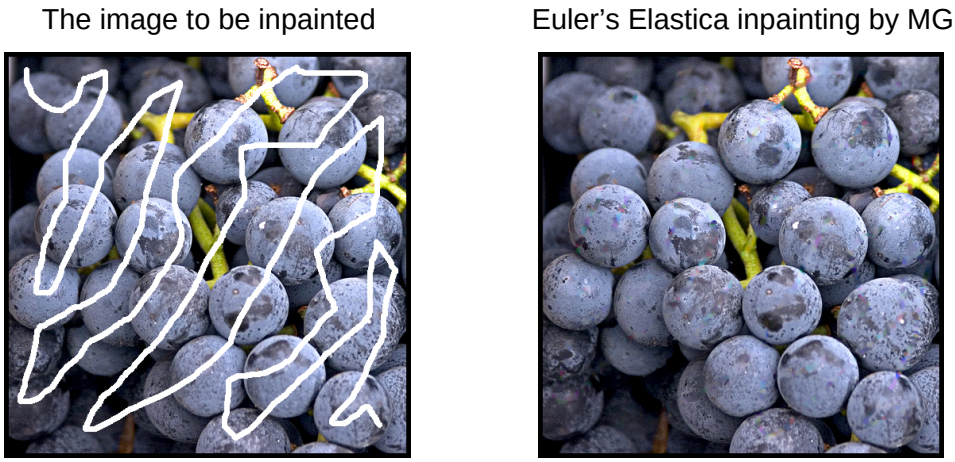


Figure 10.7: This example illustrates the recovering of the rounded parts of the objects (grapes). Also due to the fast processing obtained by using MG, a color image can be inpainted in no extra time; see Table 8.3.

[28] H. Kostler, K. Ruhnau and R. Wienands, *Multigrid solution of the optical flow system using a combined diffusion- and curvature-based regularizer*, Numer. Lin. Alg. Applics. **15** (2008), 201–218.

[29] A.E.H. Love, *A Treatise on the Mathematical Theory of Elasticity*, Dover, New York 4th ed., 1927.

[30] 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.

[31] S. Masnou, *Disocclusion: a variational approach using level lines*, IEEE Trans. Image Process., **11**(2) (2002), 68–76.

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

[33] Kang-Yu Ni, *Variational PDE-based Image Segmentation and Inpainting with Applications in Computer Graphics*, UCLA CAM-report 08–39, (2008), USA.

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

[35] 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.

[36] 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, **82** (2006) 69–94.

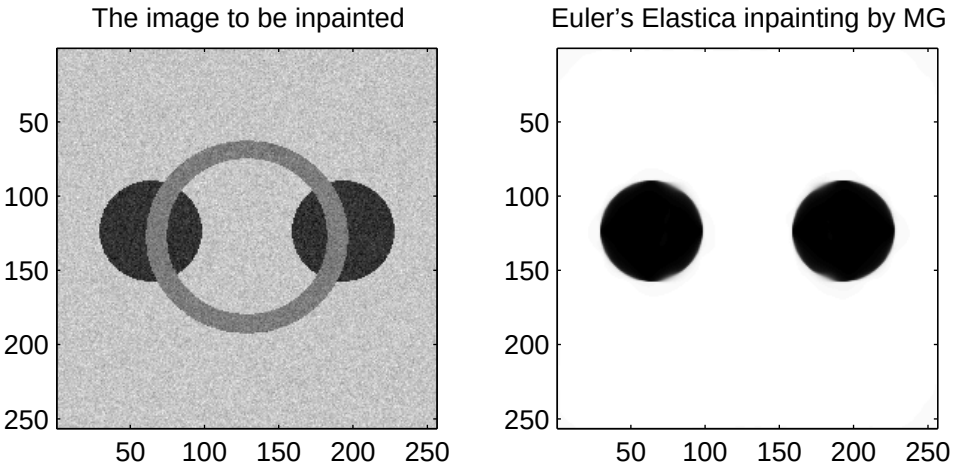


Figure 10.8: Final example of a good performance of MG processing noisy images.

[37] C.-B. Schönlieb, A. Bertozzi, *Unconditionally Stable Schemes for Higher Order Inpainting*, UCLA CAM-report 09–78, (2009), USA.
[38] P. Smereka, *Semi-Implicit level set methods for curvature and surface. diffusion motion*, J. Sci. Comput. **19** (2003), 439–456.
[39] G. Strang, *Introduction to Applied Mathematics*, Wellesley Cambridge Press, 1986.
[40] U. Trottenberg, C. Oosterlee, and A. Schuller, *Multigrid*, Academic Press, 2001.
[41] B. P. Vollmayr-Lee and A. D. Rutenberg, *Fast and accurate coarsening simulation with an unconditionally stable time step*, Phys. Rev. E **68**(2) (2003), 066703.1–066703.13.
[42] C. Vogel, *Computational Methods For Inverse Problems*, Society for Industrial and Applied Math; 1st edition, 2002.
[43] R. Wienands and W. Joppich, *Practical Fourier Analysis For Multigrid Methods*, Chapman and Hall/CRC, Florida USA, 2005.

Carlos Brito-Loeza: Centre for Mathematical Imaging Techniques and Department of Mathematical Sciences, University of Liverpool, Peach Street, Liverpool, L69 7ZL, United Kingdom

E-mail address: cbrito@liverpool.ac.uk

Ke Chen: Centre for Mathematical Imaging Techniques and Department of Mathematical Sciences, University of Liverpool, Peach Street, Liverpool, L69 7ZL, United Kingdom [Web http://www.liv.ac.uk/www/cmit]

E-mail address: k.chen@liverpool.ac.uk