University of California

Los Angeles

Applications of Variational Models and Partial Differential Equations in Medical Image and Surface Processing

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Mathematics

by

Bin Dong

2009

The dissertation of Bin Dong is approved.

Luminita A. Vese

Paul Thompson

Andrea L. Bertozzi

Stanley J. Osher, Committee Chair

University of California, Los Angeles

2009

To my parents,

who always have faith in me;

whose unrelenting and selfless love

keeps me going until today.

Table of Contents

List of Figures

List of Tables

Acknowledgments

Foremost, I would like to express my sincere gratitude to my PhD thesis advisor Professor Stanley Osher for his wise guidance, constant encouragement and support over the past four years. Not only I learnt numerous advanced and useful mathematical techniques from him, but also realized the crucial aspects to be a good researcher that are reflected from him, namely passion, intelligence and hard working. I also want to thank for his generous financial supports that gave me several opportunities to travel to workshops and conferences.

My thanks also go to my other committee members, Professor Andrea Bertozzi, Professor Luminita Vese, and Professor Paul Thompson. I want to thank Professor Andrea Bertozzi for organizing VIGRE summer internships that supported my research for the past two summers; I want to thank Professor Luminita Vese for the discussions and her suggestions on part of the content in Section 3.2; I also want to thank Professor Paul Thompson for his valuable comments given during my presentations at the LONI-CCB.

Here I would also like to thank all my collaborators. I thank Professor Zuowei Shen (Department of Mathematics, National University of Singapore) for his insights and constructive suggestions for Section 3.2 and 3.3. I thank Dr. Aichi Chien (Division of Interventional Neuroradiology, David Geffen School of Medicine) for all her help and efforts that lead to the current form of Chapter 2 and Section 3.2. I thank Professor Ivo Dinov (Department of Statistics and Center for Computational Biology) for organizing the SIG-WAVE meeting at LONI-CCB for the past three years, from which I am benefited a lot. I also truly appreciate his important inputs in Chapter 1. I thank Doctor Eric Savitsky

(Department of Emergency Medicine) for his sound medical advice and the vast valuable medical data that he generously provided, which lead to Chapter 5. I thank Professor Wotao Yin (Department of Mathematics, Rice University) for his sound suggestions on the materials in Chapter 4.

I also want to thank David Y. Mao, James Y. Jian, Yifei Lou, Yongning Zhu, Rongjie Lai, Yingying Li, Wenhua Gao, Tom Goldstein, Dr. Xiaoqun Zhang and Dr. Yonggang Shi for all the helpful discussions that I had with them over the years. My gratitude also goes to all the staffs of IPAM and the Mathematics Department, especially Maggie Albert and Babette Dalton.

My final, but most heartfelt, acknowledgment must go to my girlfriend Ranran Wang, who helped me walk through countless difficulties during my PhD study.

This work is supported by: NIH P20 MH65166; Department of Defense; NSF DUE 0716055; NSF DMS-0714807; and NIH U54 RR021813 VIGRE of UCLA Mathematics Department; SN-30014, Center for Computational Biology NIH Toga; and the Telemedicine and Advanced Technology Research Center (TATRC) of the US Army Medical Research and Material Command (MRMC).

Vita

1981	Born, Beijing, China.
1999–2003	B.S., School of Mathematical Sciences, Peking University,Beijing, China.
2003–2005	M.S., Department of Mathematics, National University of Singapore, Singapore.
2005–2009	Teaching and Research Assistant, Department of Mathematics, University of California, Los Angeles, USA.

Publications

B. Dong, A. Chien, Z. Shen and S. Osher, A new multiscale representation for shapes and its application to blood vessel recovery, submitted, March 2009.

B. Dong, E. Savitsky and S. Osher, A novel method for enhanced needle localization using ultrasound-guidance, UCLA CAM-Report 08-65, Sep. 2008.

B. Dong, A. Chien, Y. Mao, J. Ye, S. Osher, Level set based brain aneurysm capturing in 3D, submitted, Nov 2008.

S. Osher, Y. Mao, B. Dong, W. Yin, Fast linearized Bregman iterations for compressive sensing and sparse denoising, accepted by Communications in Mathematical Sciences (CAM-Report 08-37), Dec 2008.

B. Dong, A. Chien, Y. Mao, J. Ye, S. Osher, Level set based surface capturing in 3D medical images, Proc MICCAI, 162–169, 2008.

B. Dong, J. Ye, S. Osher and I. D. Dinov, Level set based nonlocal surface restoration, Multiscale Modeling and Simulation (MMS), 7(2), 589–598 (CAM-Report 07-44), 2008.

B. Dong, Y. Mao, I. D. Dinov, Z. Tu, Y. Shi, Y. Wang and A. W. Toga, Wavelet-based representation of biological shapes, CAM-Report 07-36, Sep. 2007.

B. Dong and Z. Shen, Pseudo-splines, wavelets and framelets, Appl. Comput. Harmon. Anal., 22, 78–104, 2007.

B. Dong and Z. Shen, Linear independence of pseudo-splines, Proc. Amer. Math. Soc., 134 (9), 2685–2694, 2006.

B. Dong and Z. Shen, Construction of biorthogonal wavelets from pseudo-splines, J. Approx. Theory, Vol 138 (2), 211–231, 2006.

Abstract of the Dissertation

Applications of Variational Models and Partial Differential Equations in Medical Image and Surface Processing

by

Bin Dong

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2009

Professor Stanley J. Osher, Chair

Variational, level set and PDE based methods and their applications in digital image processing have been well developed and studied for the past twenty years. These methods were soon applied to some medical image processing problems. However, the study for biological shapes, e.g. surfaces of brains or other human organs, are still in its early stage. The bulk of this dissertation explores some applications of variational, level set and PDE based methods in biological shape processing and analysis. This dissertation also covers some aspects of compressive sensing, ℓ_1-minimizations and fast numerical solvers. Their applications in medical image analysis are also studied.

The first topic is on surface restoration using nonlocal means [1], where we extend nonlocal smoothing techniques for image regularization in [12] to surface regularization, with surfaces represented by level set functions. Numerical results show that our extension of nonlocal smoothing to surface regularization is very effective in removing spurious oscillations while preserving and even restoring sharp features. Furthermore, topology corrections are also made by our algorithms for

some of the surfaces.

The second topic is on 3D brain aneurysm capturing using level set based method. Inspired by the illusory contour techniques proposed by [36, 37], we present a level set based surface capturing algorithm to capture the aneurysms from the vascular tree. Numerical results are presented to show the accuracy, consistency and robustness of our method in capturing brain aneurysms and volume quantification.

The third topic is on multiscale representations (MSR) of 3D shapes. We introduce a new level set and PDE based MSR for shapes, which is intrinsic to the shape itself, does not need any parametrization, and the details of the MSR reveal important geometric information. Based on the MSR, we then design a surface inpainting algorithm to recover 3D geometry of blood vessels. Because of the nature of irregular morphology in vessels and organs, both phantom and real inpainting scenarios were tested using our new algorithm. Numerical results show that the inpainting regions are nicely filled in according to the neighboring geometry of the vessels and allow us to accurately estimate the volume loss of vessels.

The last, but definitely not the least, topic is on Bregman iteration as a fast solver for ℓ_1-minimizations in compressive sensing and medical image analysis. We analyzed the convergence properties of linearized Bregman and then improve its convergence speed. We further observe that a general TV-based model can be converted to an ℓ_1-minimization which can then be solved efficiently using Bregman iterations. Finally, an application of ℓ_1-minimization is considered for needle tracking in ultrasound images.

CHAPTER 1

Variational and PDE Models in Surface Processing I: Surface Restoration via Nonlocal Means

1.1 Introduction

Variational and partial differential equations (PDEs) based image denoising models have had great success in the past twenty years (see e.g. [3, 23, 146, 148]). The goal is to remove noise (in the form of random high frequency oscillations) from an image, while keeping features, e.g. sharp edges and textures.

Recently, some of the models used for image denoising have been extended to denoising surfaces (see [5–7, 14, 20, 21]). There are mainly two ways to represent surfaces. One is using triangular meshes, the other is implicitly, usually using level set functions. The well known advantages of handling implicitly represented surfaces over triangulated surfaces are numerical simplicity and flexibility of topological changes. Topological flexibility is important and it makes possible for our algorithm to do not only denoising, but also topology corrections. In this chapter we shall focus on implicitly represented surfaces. We note that Yoshizawa, Belyaev and Seidel [22] recently introduced a nonlocal averaging algorithm for denoising triangulated surfaces. Their algorithm is also related to some earlier works on semi-local similarity-based shape descriptors and applications in

shape matching, retrieval, and modelling [4, 10, 11, 17, 24]. Although both our and Yoshizawa-Belyaev-Seidel's methods are based on ideas of nonlocal means introduced by Buades, Coll and Morel [1] for image denoising, our approach is different in the sense that we are handling implicit surfaces with all their advantages. Elmoataz, Lezoray and Bougleux [9] recently introduced a nonlocal discrete regularization framework, which is the discrete analogue of the continuous Euclidean nonlocal regularization functionals in [12, 13]. This method is applied for image and manifold processing using weighted graphs of the arbitrary topologies. This approach is useful for various types of images, meshes, manifolds and data represented as graphs.

1.1.1 Nonlocal Means

The nonlocal means method was introduced by Buades, Coll and Morel in [1]. They suggested the following nonlocal averaging for image denoising:

$$NL(u)(x) = \frac{1}{c(x)} \int_{\Omega} e^{-d_a(u(x),u(y))/h^2} u(y) dy$$

where $c(x)$ is a normalization factor and

$$d_a(u(x), u(y)) = \int_{\Omega} G_a(t) |u(x+t) - u(x-t)|^2 dt,$$

with G_a a Gaussian with standard deviation a. This algorithm gives excellent results in image denoising (see [1, 12]). In a later work by Buades, Coll and Morel [2], they showed that the nonlocal means filter can be extended from the linear regression neighborhood filter. They also derived and analyzed the corresponding PDE to the linear regression neighborhood filter, as well as the connection between bilateral filters [19] and Perona-Malik equations [16].

1.1.2 Variational Viewpoints

In [12, 13], Gilboa and Osher put nonlocal averaging into a variational framework (an earlier variational formulation was done in [15]). We will extend their formulation to surface denoising.

In [12, 13], the following energy was used

$$J(u) := \frac{1}{4}\int_{\Omega} \left|\nabla_w u\right|^2 dx.$$

(This uses ideas introduced in [25].) The corresponding gradient flow of the energy $J(u)$ is the nonlocal heat equation

$$u_t = \nabla_w^2 u := \frac{1}{2}\mathrm{div}_w(\nabla_w u) = \int_{\Omega} (u(y) - u(x))w(x,y)dy, \tag{1.1}$$

for $x \in \Omega$. The discrete version of (1.1) is

$$u_j^{k+1} = u_j^k + dt \sum_{l \in \mathcal{N}_j} w_{jl}(u_l^k - u_j^k), \tag{1.2}$$

where u_j denotes the value of u at grid point j with j going over all grid points in the computational domain, and $\mathcal{N}_j$ is some neighborhood of j such that $w(l,j) > 0$ for $l \in \mathcal{N}_j$. The CFL restriction for the time step dt is

$$1 \geq dt \sum_{l \in \mathcal{N}_j} w_{jl}, \quad \forall j.$$

In [12], they showed that excellent denoising results can be obtained by using some properly chosen weight w, which is related to the kernel of nonlocal means.

The rest of this chapter is organized as follows. In Section 1.2, we will show how to choose the weight w and apply (1.2) to surface denoising. In Section 1.3

we will present some denoising results for both phantom and real surface data.

1.2 Numerical Strategies and Discussions

For the rest of this chapter, all surfaces are represented by signed distance functions. To be precise, the surface S is taken as the boundary of some domain Σ. The corresponding signed distance function ϕ satisfies: $\phi(x) < 0$, for $x \in \Sigma$; $\phi > 0$, for $x \notin \Sigma$; and $|\nabla\phi| = 1$ away from its singularities. Thus we have $S = \{x : \phi(x) = 0\}$. Our strategy for surface denoising is described as follows.

Strategy:

1. Surfaces are represented by signed distance functions ϕ.

2. Calculations are performed on a narrow band of zero level set of ϕ, denoted as Σ_δ with δ the width of narrow band.

3. Choice of weight $w(x, y)$ and similarity function $D(x, y)$:

$$\begin{aligned} w(x,y) &= e^{-|x-y|^2/c_1} e^{-D(x,y)/c_2}, \\ D(x,y) &= \|\phi[x] - \phi[y]\|_2^2, \quad x \in \Sigma_\delta,\ y \in N_x, \end{aligned}$$

   where N_x is a neighborhood of x within Σ_δ, and $\phi[x]$ is a 3D patch of ϕ centered at x.

4. Under discrete formulations, the iterative scheme (1.2) now reads as

$$\phi_j^{k+1} = \phi_j^k + dt \sum_{l \in \mathcal{N}_j} w_{jl}(\phi_l^k - \phi_j^k), \tag{1.3}$$

with w_{jl} calculated as given in (3), and dt chosen to be

$$dt = 1/\max_j\{\sum_{l\in\mathcal{N}_j} w_{jl}\},$$

where j goes over all grid points in Σ_δ.

5. Stopping time $k = K$ is chosen by the user (see the remark below for more details).

Remark 1.2.1.

1. *The reason we use a narrow band calculation is not only for numerical efficiency, but also because we want to (and should) focus more on the zero level set and its neighboring level sets. The nearby level sets contain more relevant information than distant level sets.*

2. *The way of choosing weight w_{jl} in (3) is, in fact, a semi-nonlocal version of the original nonlocal means in* (1.2) *(see [12]). The parameter c_1 controls how much one wishes to penalize distant of two grid points in the weight, while c_2 controls how much one wishes to penalize similarity of the two patches. Larger c_1 allows one make use of more remote information, while larger c_2 gives results with sharper features (but requires more iterations in general).*

3. *In the definition of similarity function $D(x, y)$ in (3), we simply measure the L_2 distance of two cubical patches without doing Gaussian smoothing first. This is because the signed distance function is not very noisy, even though its zero level set is quite noisy. Hence the direct L_2 distance gives a good measurement of similarity, which saves computation time.*

4. *Here we let users to determine the stopping iteration K. The reason is that for real noisy surfaces (like the cortical surfaces from MRI scans in Section 3.2), the type of noise can be arbitrary (not necessarily Gaussian noise). Hence there is no apparent way of giving a unified stopping criteria as people did for image denoising. In our experiments below, the number of iterations is around 300 for the twin-cubes and city terrain, and around 150 for the cortical surfaces. In future, we shall define a unified stopping criteria for our method.*

1.3 Numerical Results

In this section, we present numerical results for the algorithm given in the previous section. We will first show some denoising results for some shapes corrupted with Gaussian white noise. Then we test the algorithm on some biological data generated from high resolution MRI scans. For all experiments, the width δ of the narrow band Σ_δ is chosen to be 2, i.e. Σ_δ has grid points of 5 level sets including the approximate zero level set. For each given grid point $x \in \Sigma_\delta$, N_x is chosen to have 100 grid points which are closest to x within Σ_δ. The patch $\phi[x]$, centering at x, is taken to be of size $5 \times 5 \times 5$ (whose grid points could lie outside of Σ_δ).

All biological data, i.e. cortical surfaces and MRI images, are provided by the Laboratory of Neural Imaging, Center for Computational Biology, UCLA, http://www.ccb.ucla.edu.

1.3.1 Denoising: Synthesis Noise

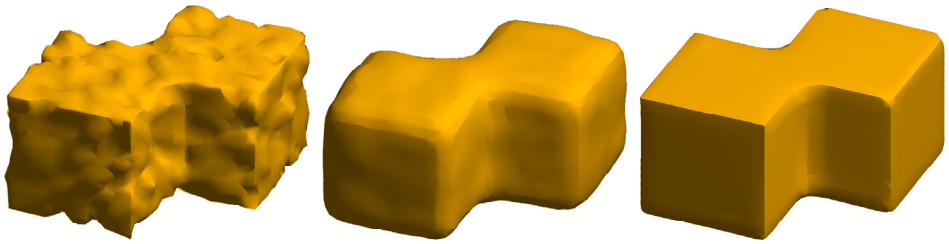
The noisy level set function $\tilde{\phi}$ is given by $\tilde{\phi} = \phi + \varepsilon$, with $\varepsilon \in \mathcal{N}(0, \sigma)$. We compared our approach to mean curvature based surface regularization:

$$\phi_t = |\nabla\phi|\Big(\nabla \cdot \Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - \lambda(H(\phi) - H(\tilde{\phi}))\Big),$$

with H the Heaviside function. The parameter λ is chosen manually for each example.

Figure 1.1 is a man made shape, where two boxes are attached together. Mean curvature smoothing gives a fair result which removes most of the noise and preserves some edge information. Our approach gives a much better result. All noise is removed and the edges are not only preserved perfectly, but also reconstructed for some regions. This is not surprising because one can regard the nonlocal smoothing as a "copy-paste" procedure. Since the noisy shape has some sharp edges uncontaminated, the algorithm then "copies" them to the regions where the edges are lost and reconstruct them almost perfectly (as one can see from (c) in Figure 1.1). One may notice that some corners are not recovered from the noisy data. This is because the neighborhood N_x is not global and we are not taking rotations into account, so that within N_x, no similar information (i.e. corners) can be found. We shall take rotation into account in our future work. Figure 1.2 shows denoising results for a cortical surface (gray matter), where our approach preserves the features (e.g. sulci) very well. Figure 1.3 shows denoising results for man made city terrain surface. This is actually a harder surface to denoise than the previous ones, because the "holes" on the base of the surface make it a high genus surface, i.e. the topology is changed by noise. Also, the thickness of the base is only 3 grid points, which is hard to preserve for some

algorithms. For example, the mean curvature algorithm will destroy the base as one can see from (c) of Figure 1.3. If one uses a triangular mesh based algorithm to denoise the surface, it will be very tricky to correct the topology and reconstruct back the base. In contrast, our algorithm here did a very good job in removing noise, and in topology corrections (see (d) of Figure 1.3).



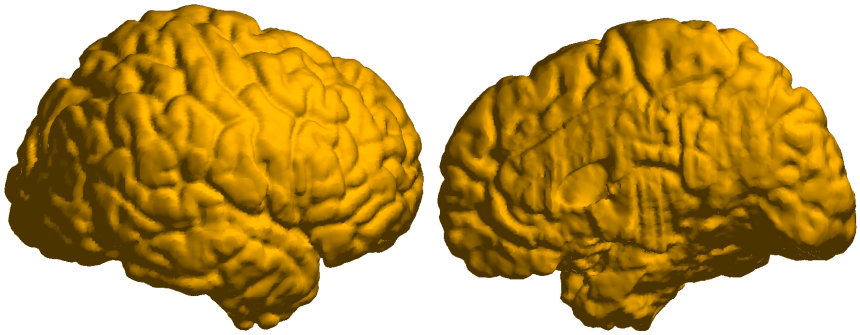
(a) (b) (c)

Figure 1.1: Figure (a) is the noisy shape; (b) is the result using mean curvature smoothing with $\lambda = 0.1$; (c) is our approach.

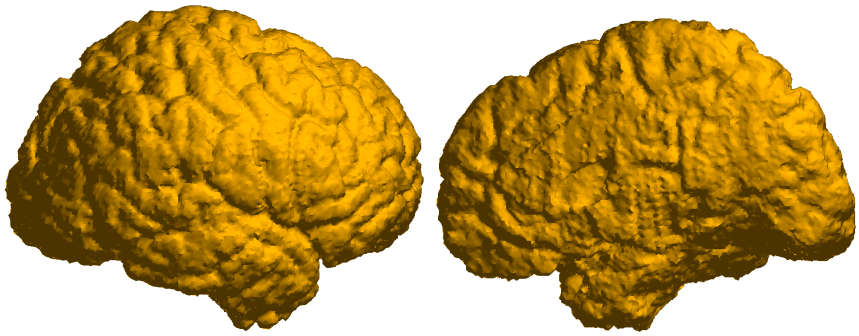
1.3.2 Denoising: Real Noisy Data

The cortical surface (white matter) in Figure 1.4 is obtained from high-resolution MRI scans. The raw data is a volumetric mask of size $181 \times 217 \times 181$ segmented from MRI scans manually, which means the segmentation is accurate; however it is very noisy. Then the mask is transformed to a signed distance function using a fast sweeping method introduced in [18].

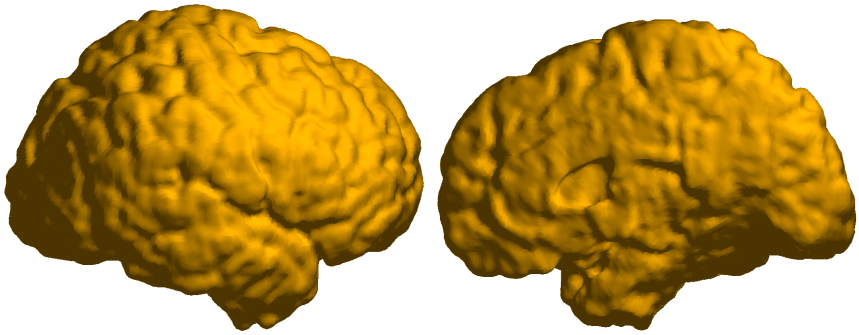
Cortical surfaces are much more complicated than the previous examples. First, they have very deep and narrow sulci and thin gyri. Since sulci and gyri are very important features for cortical surfaces, we want to preserve them as much as possible during regularization. In addition, the noisy cortical surfaces have lots of isolated small pieces that need to be removed. Our algorithm performed well here in removing noise and isolated pieces, while preserving sulci and gyri, as one can see from Figure 1.4 and Figure 1.5.



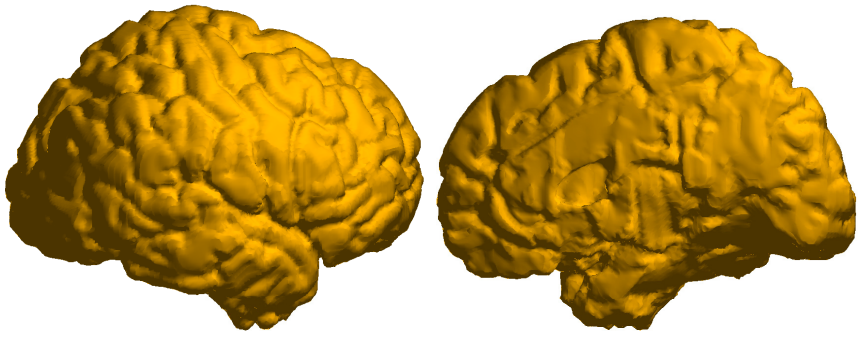
(a)



(b)

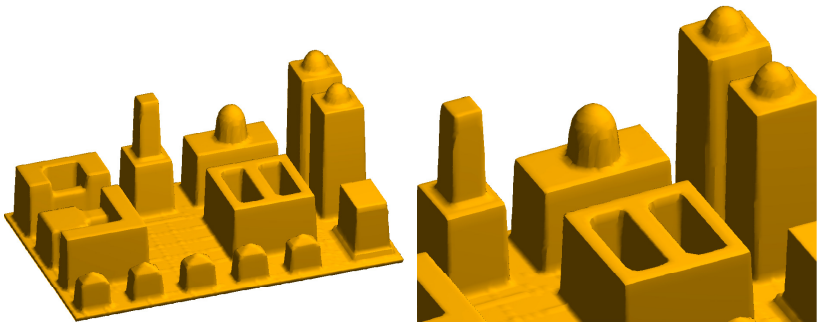


(c)

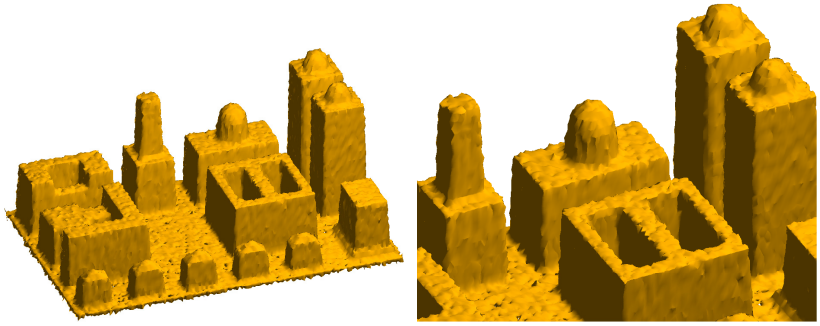


(d)

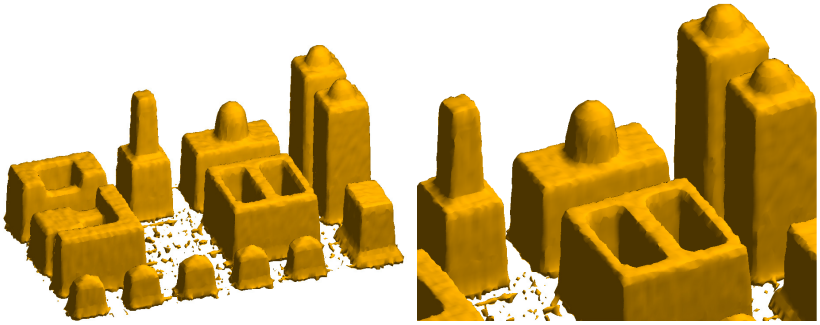
Figure 1.2: The two figures in (a), (b), (c) or (d) are the front and back views of the same cortical surface. Figure (a) is the clean gray matter; (b) is the noisy one; (c) is the denoising result from mean curvature smoothing with $\lambda = 0.001$; (d) is the result by our approach.



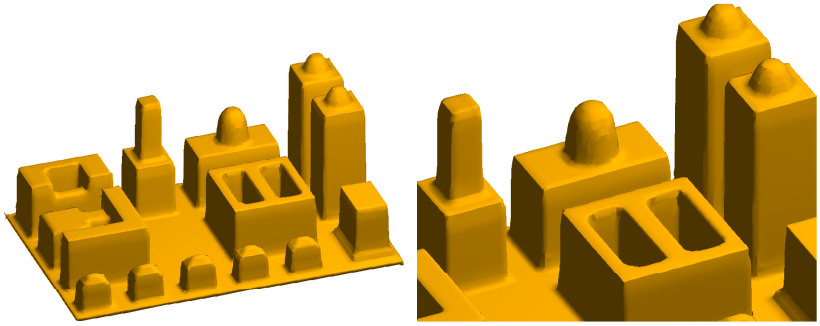
(a)



(b)

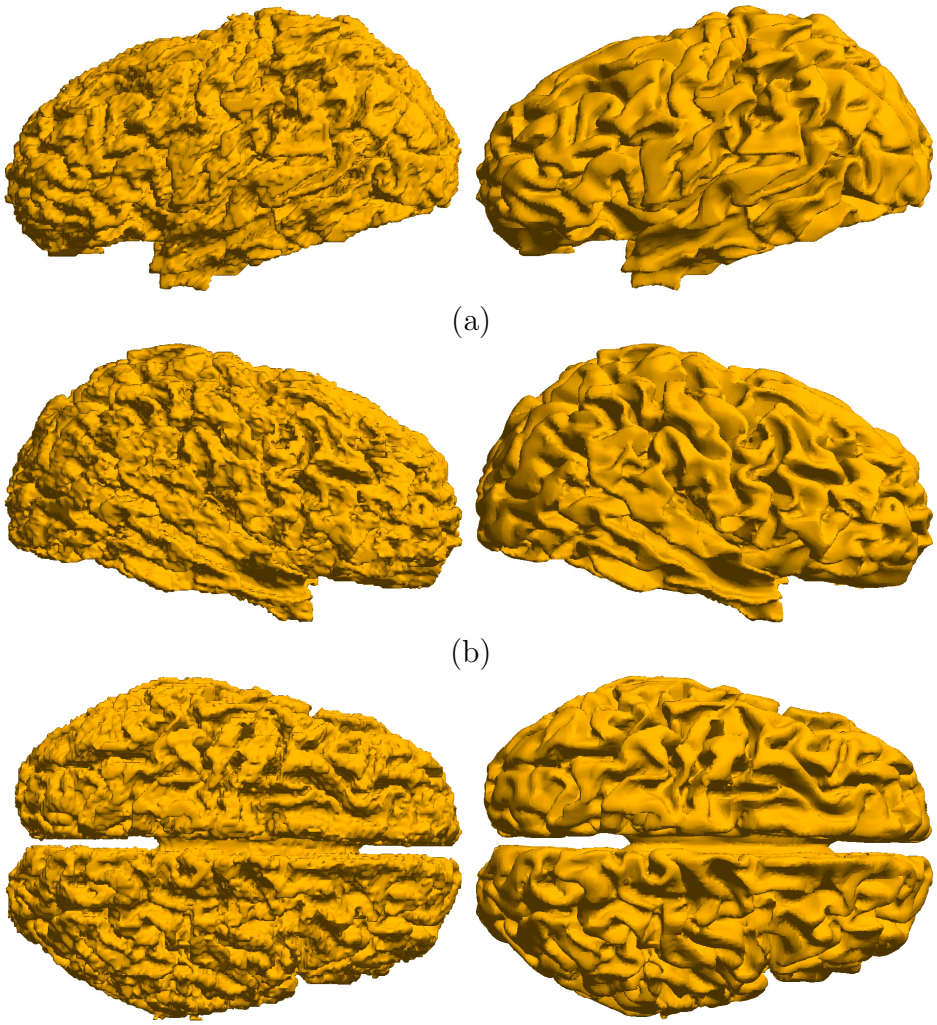


(c)



(d)

Figure 1.3: The two figures in (a), (b), (c) or (d) are the global view and a close-up of the same city terrain. Figure (a) is the clean terrain surface; (b) is the noisy one; (c) is the denoising result from mean curvature smoothing with $\lambda = 0.01$; (d) is the result by our approach.



(a)

(b)

(c)

Figure 1.4: The two figures in (a), (b) and (c) are noisy (left) and regularized (right) white matter viewed from left, right and top.

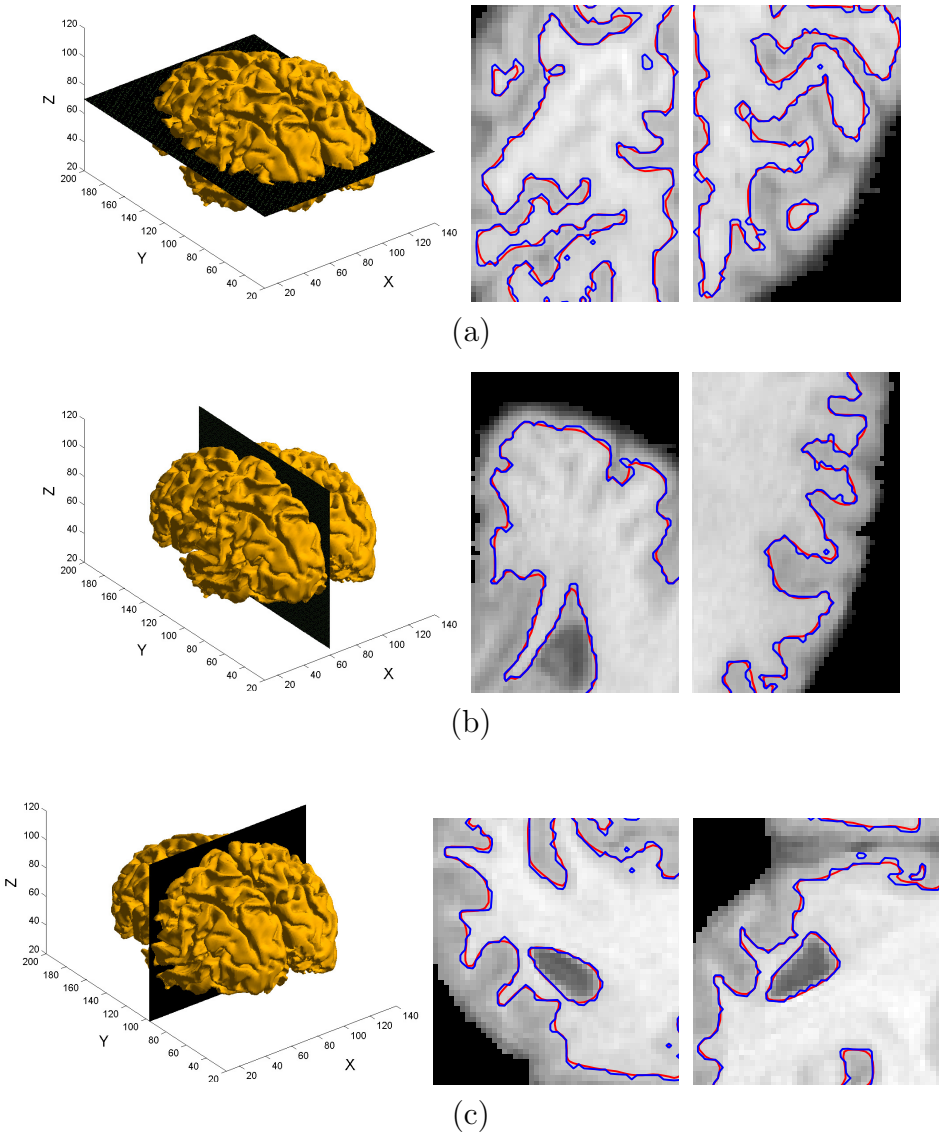


Figure 1.5: The first figure in (a), (b) and (c) illustrates which slice of the cortical surface is shown. The second and third figures in (a), (b) and (c) are the corresponding close-ups for the axial ((x, y)-slice), sagittal ((y, z)-slice) and coronal ((x, z)-slice) slices of the cortical surface and the MRI scan. Here blue curves are the original segmentation for white matter, and the red ones are regularized one.

CHAPTER 2

Variational and PDE Models in Surface Processing II: Capturing Brain Aneurysms from Vascular Trees

2.1 Introduction

Subarachnoid hemorrhage, primarily from brain aneurysm rupture, accounts for 5 to 10% of all stroke cases with a high fatality rate [26]. Advancements in neuroimaging technology have helped these aneurysms to be more frequently found prior to rupture. A method to determine if aneurysms are at higher risk of rupturing would be extremely valuable. Brain aneurysm rupture has been reported to be related to the size of aneurysms [27]. It is known that the risk of rupture greatly increases as the aneurysm becomes larger [28, 29]. Currently, methods to determine the aneurysm size are to simply manually measure the size of the neck and the dome of aneurysms. However, these methods may overlook important geometric information and are very hard to perform consistently across subjects.

Our goal is to first segment the aneurysm from the entire blood vessel with minimal human interaction, then compute its volume and other geometric quantities. This problem can actually be realized as an illusory surface capturing problem by observing Fig. 2.1, which is an extension from illusory contours in

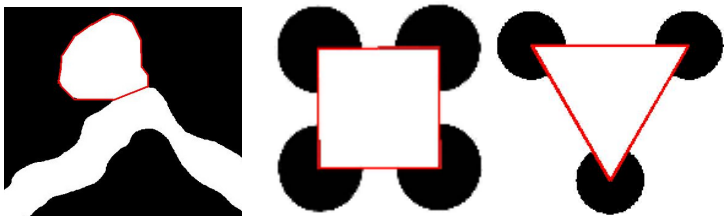


Figure 2.1: Left: phantom vessel in 2D; middle: Kanizsa square; right: Kanizsa triangle. The red curves are illusory contours.

2D. The boundaries of the aneurysm that are not part of the blood vessel surface can be completed naturally by illusory surfaces. Illusory contours have been intensively studied in cognitive neuroscience, where people find that the human vision system is capable of combining nonexistent edges and making meaningful visual organization of both the real and imaginary contour segments [30–32] (e.g. the Kanizsa square and triangle [30] in Fig. 2.1). Various researchers have introduced mathematical models and techniques to mimic the human vision system in detecting and capturing perceptual contours in images [33–38]. These mathematical models can be used to describe the process in medical evaluation when the location of an aneurysm needs to be identified. Given that our problem is to first capture and then calculate volumes and geometries of aneurysms, representing surfaces using level set functions and designing a proper surface evolution PDE is essential. Therefore, we introduce a level set and PDE based illusory surface model, inspired by the illusory contour models in [36], to capture aneurysms, and calculate their volumes and geometries.

The focus of this chapter is to introduce a novel method to capture a specific part of a given pre-segmented surface obtained from 3D images. Therefore, we will not place emphasis on the techniques of surface reconstruction from 3D images. However, we note that different segmentations from a 3D image may result in rather different surfaces in terms of geometry. In fact, surface segmentation from

3D images is highly nontrivial, and is a very active research area. Interested readers can consult [39–41] and their references for detailed techniques of blood vessel segmentations. As an example, we applied a simple thresholding method (with carefully chosen thresholds) followed by fast sweeping method [18] and Gaussian smoothing to reconstruct the surface represented by a level set function [174], which takes positive values inside the vessel region and negative values outside. We note that one can replace the Gaussian smoothing by some more sophisticated smoothing techniques, e.g. nonlocal means [1, 42], if there are some sharp and delicate features in the surfaces need to be well reserved. Fig. 2.2 illustrates the idea of reconstruction of vessel surfaces from 3D images and an example of brain aneurysm.

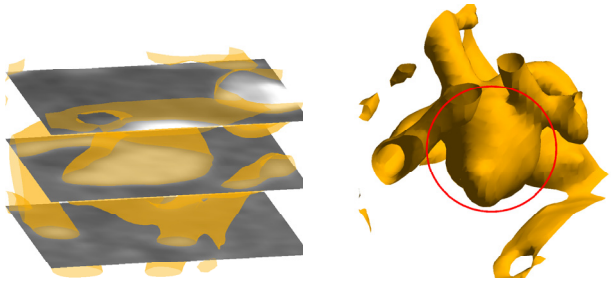


Figure 2.2: Reconstruction from 3D images. Left figure shows a few slices of images corresponding to the reconstructed surface; right figure shows the reconstructed surface with the red curve circling the aneurysm.

2.2 Review of Level Set Based Illusory Contour Capturing

Many PDE based methods have been proposed to identify illusory contours as well as explain the phenomena [33–38]. One of the most typical ones was introduced by Sarti et al [34, 35]. In their work, they first chose a fixation point inside the domain bounded by the ideal illusory contour, and constructed a surface on the

whole domain on the basis of the point, then evolve the entire surface based on the image gradient. In fact, our user interactive initialization strategy (in Section 2.3.3) is very similar to their fixation point idea. More details can be found in [35].

In this section, we shall focus on reviewing the level set formulations of the illusory contour problems introduced by Zhu and Chan [36], because our PDE model is motivated by theirs. As a convention, all level set functions in this chapter take negative values inside the domain of interest and positive outside. For example in Figure 2.1, denoting the regions inside the red curves as Ω, then the corresponding level set function ϕ satisfies

$$\phi(x) \begin{cases} < 0 & x \in \Omega \\ > 0 & x \in \Omega^c, \end{cases} \tag{2.1}$$

and $\partial\Omega$, which is the zero level set of ϕ, represents the illusory contours (red curves).

The first model considered in [36] is

$$E(\phi) = \int_\Omega \Big(d\delta(\phi)|\nabla\phi| + \alpha H(\psi)H(\phi) + \beta\delta(\phi)|\nabla\phi| \Big) dx, \tag{2.2}$$

where ψ is the signed distance function obtained from a given image (see e.g. Fig. 2.1) whose zero level set represents the boundaries of the objects in the image, and $d = |\psi|$ is the corresponding unsigned distance function. The symbol ∇ is the gradient operator, $\delta(\phi)$ is the Dirac delta functional, and $H(\phi)$ is the Heaviside function. The energy term $\alpha H(\psi)$ ensures that the model will capture only the inner contour, instead of the outer one (see e.g. the Kanizsa square and triangle in Fig. 2.1). We note that in [36], the authors also had an additional

term $\int_\Omega \kappa^2 dx$ in the energy (5.2) to ensure the continuity of the curvature of ϕ. However, this term will result in a fourth order evolution PDE which significantly slows down the computations. Since what is important for us is the consistency of segmentation of aneurysms and their volumes, it seems unnecessary to have the curvature term in the energy.

From equation (5.2), the corresponding gradient flow can be written as

$$\frac{\partial \phi}{\partial t} = \delta(\phi)\nabla d \cdot \frac{\nabla \phi}{|\nabla \phi|} + \delta(\phi) d \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} - \alpha \delta(\phi) H(\psi) + \beta \delta(\phi) \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}. \quad (2.3)$$

Since the function $\delta(\phi)$ is concentrated only on the zero level set of ϕ, the PDE (2.3) only describes a motion for the zero level set of ϕ. Similar to [43], to ensure all level sets of ϕ have similar motions and to be able to solve the PDE on the entire 3D rectangular domain, we replace $\delta(\phi)$ in (2.3) by $|\nabla \phi|$ and obtain the following PDE

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \Big(\nabla d \cdot \frac{\nabla \phi}{|\nabla \phi|} + d \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} - \alpha H(\psi) + \beta \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \Big). \quad (2.4)$$

Numerical results in [36] show that solving the PDE (2.4) gives fairly good results. However, the sharp corners, e.g. the corners in Kanizsa squares and triangles in Fig. 2.1, are not well captured. Therefore in that paper, the authors also considered the following improved model which enables the final curves to stick to sharp corners more closely

$$E(\phi) = \int_\Omega \Big((1 + \mu c_{a,b} \kappa^+(\psi)) d \delta(\phi) |\nabla \phi| + \alpha H(\psi) H(\phi) + \beta \delta(\phi) |\nabla \phi| \Big) dx, \quad (2.5)$$

where μ is some constant, $c_{a,b}$ is some restriction function defined in (2.13) and $\kappa^+(\psi)$ is the positive part of curvature. The corresponding evolution PDE to the

energy (2.5) is

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \bigg(\nabla[(1 + \mu c_{a,b} \kappa^+(\psi))d] \cdot \frac{\nabla \phi}{|\nabla \phi|} + [(1 + \mu c_{a,b} \kappa^+(\psi))d] \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} - \alpha H(\psi) + \beta \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \bigg). \tag{2.6}$$

Numerical experiments in [36] show that the model (2.6) does an excellent job in capturing illusory contours, especially at regions with sharp features.

2.3 Our Method and Model Comparisons

In this section, we will first discuss the possibility of extending (5.2) and (2.5) directly from 2D to 3D, and what difficulties and problems one may encounter. Then, motivated by these discussions, we shall introduce our model in Section 2.3.2.

2.3.1 Direction Extension from 2D to 3D?

Let us first consider the model (5.2) and call it E_1. Notice that it can be extended to 3D trivially. Since it gives a fairly good result in 2D, one may expect it to do so in 3D. However, the following special examples says otherwise. Take $\alpha = \beta = 0$ in (5.2), and take ψ represents a unit circle/sphere, and ϕ represents a circle/sphere with radius $r \in [0, 1]$ (see Fig. 2.3). Then we can write E_1 explicitly as a function of r,

$$E_1(r) = \int_S d(s) ds = \begin{cases} 2\pi(1 - r)r, & \text{2D} \\ 4\pi(1 - r)r^2, & \text{3D} \end{cases} \tag{2.7}$$

The plot of $E_1(r)$ in Fig. 2.3 shows that one should initialize ϕ by a circle with radius $r > \frac{1}{2}$ and $r > \frac{2}{3}$ respectively in 2D and 3D in order to converge to the right solution. Thus, the initialization constraint in 3D is more restrictive than that in 2D. This is because the shrinking force from minimizing the surface area in 3D is stronger than that from minimizing the curve length in 2D. On the other hand, model (5.2) does not perform well in terms of capturing sharp corners in both 2D and 3D and the problem is even more severe in 3D than in 2D for similar reasons.

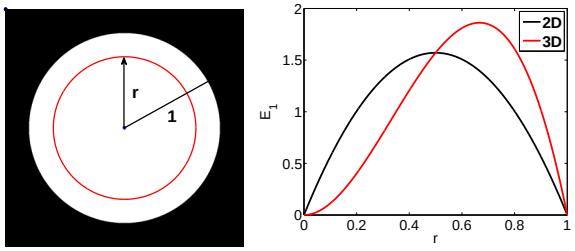


Figure 2.3: Let figure illustrates the special example consider above, while the right one shows the plot of $E_1(r)$.

Let us now consider (2.5), and denote it by E_2. Take the same example as in Fig. 2.3, and let $\alpha = \beta = 0$ and $c_{a,b} \equiv 1$. If we choose κ to be mean and Gaussian curvature, we shall have the following two formulas for $E_2(r)$ in 2 and 3 dimensions,

$$E_2(r) = \int_S (1 + \mu\kappa_m) d(s) ds = \begin{cases} 2\pi(1 + \frac{\mu}{r})(1 - r)r, & \text{2D} \\ 4\pi(1 + \frac{\mu}{r})(1 - r)r^2, & \text{3D;} \end{cases} \tag{2.8}$$

$$E_2(r) = \int_S (1 + \mu\kappa_g) d(s) ds = \begin{cases} 2\pi(1 + \frac{\mu}{r})(1 - r)r, & \text{2D} \\ 4\pi(1 + \frac{\mu}{r^2})(1 - r)r^2, & \text{3D.} \end{cases} \tag{2.9}$$

The plots of $E_2(r)$ in Fig. 2.4 show that if we choose mean curvature for κ, then in 2D, we have global convergence, while in 3D, we still need to initialize

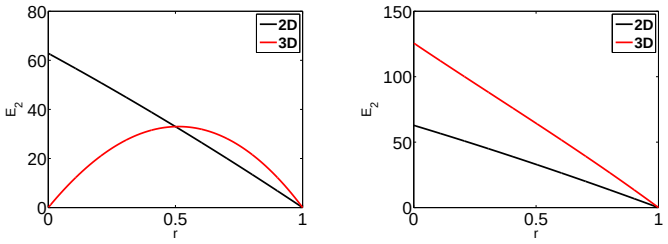


Figure 2.4: Left: plot of $E_2(r)$ in 2D/3D with κ = mean curvature; right: plot of $E_2(r)$ in 2D/3D with κ = Gaussian curvature. For both plots, the parameter $\mu = 10$.

properly in order to get the correct solution. However if we choose Gaussian curvature, we have global convergence in both 2D and 3D. This is one of the motivations for using Gaussian curvature in 3D. Another, yet more important, motivation for choosing Gaussian curvature instead of mean curvature for our particular application is illustrated in the following Fig. 2.5. As is shown below, Gaussian curvature can discriminate between aneurysm and blood vessels naturally, while mean curvature cannot. This is essentially because cylindrical objects have small Gaussian curvatures in general, while the Gaussian curvatures for bulb like objects are relatively large. In contrast, mean curvatures for both cylindric and bulb like objects generally have comparable magnitudes.

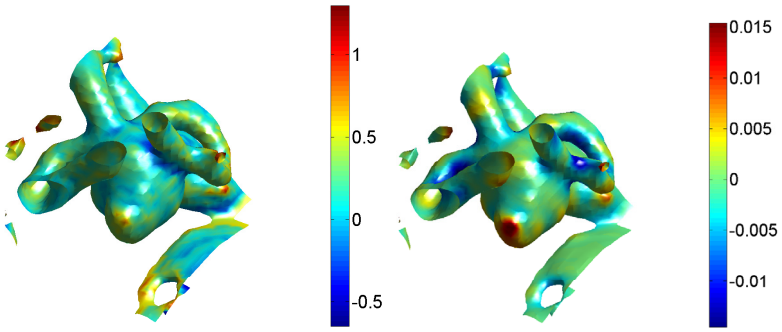


Figure 2.5: Left figure: mean curvature; right figure: Gaussian curvature.

However, directly applying (2.5) in 3D using Gaussian curvature for κ does

not give satisfactory results in general. We now consider another simple example in 2D to illustrate this. In Section 2.3.4 we shall give some comparisons in 3D. Let us now consider an example as given in Fig. 2.6, where the target object is no longer convex. We note that the aneurysms are usually not convex (see e.g. the right figure of Fig. 2.6). Therefore, the 2D example in Fig. 2.6 is a rather typical example for our applications. The left figure of Fig. 2.6 shows that if we initialize using the blue curve, we will converge to the red curve, which is not a satisfactory result because we lost the sharp feature (the small bump). To further explain the reason we obtain such a result, we recall the evolution PDE corresponding to the energy (2.5)

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \left(\nabla (A(\psi) d) \cdot \frac{\nabla \phi}{|\nabla \phi|} + A(\psi) d \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} - \alpha H(\psi) + \beta \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \right), \tag{2.10}$$

where $A(\psi) = 1 + \mu c_{a,b} \kappa^{+}(\psi)$. As we can see from (2.10), the force field $-\nabla(A(\psi)d)$ is indeed enhanced by $A(\psi)$, which means around regions with large curvature we have a stronger force pushing the zero level set of ϕ towards the sharp tip (see the middle figure of Fig. 2.6). However, because of the concavity change, the force vectors are almost tangential to the blue curve and hence most of the forces are wasted. Meanwhile, the shrinking force given by the second term of (2.10) is also enhanced by the factor $A(\psi)$. Therefore, the blue curve will eventually shrink to the red one, instead of moving forward and capturing the entire small bleb, which is a very important feature for aneurysms (see right figure in Figure 2.6). To overcome this, one may choose the initial curve containing the entire small bump. However for our particular application here, it is not reasonable to assume that one can always start with some surface that includes the entire aneurysm within or as a subsurface. In addition, it is always desirable to have a

method with less restrictive initialization constraints.

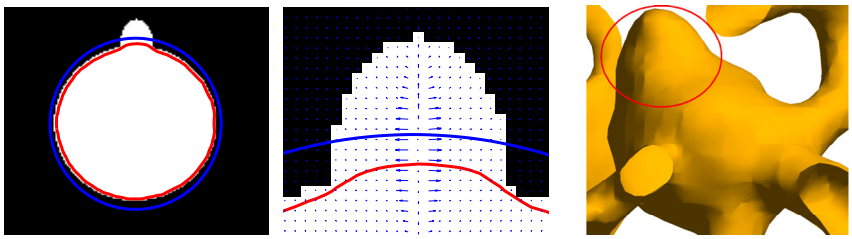


Figure 2.6: Left: special example with blue curve being the initial curve and the red one the result; middle: one zoom-in with the blue arrows specifying the vector field $-A(\psi)\nabla d$; right: the blood vessel (same object as in Fig. 2.2 with another view) with concavity change on the aneurysm region (within the red circle).

2.3.2 Our Model

Here we introduce our modified illusory surface model based on equation (2.4),

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \Big(A(\psi) \nabla d \cdot \frac{\nabla \phi}{|\nabla \phi|} + d \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} - \alpha H(\psi) + \beta \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|} \Big), \tag{2.11}$$

$$A(\psi) = 1 + \mu \kappa^{+}(\psi), \tag{2.12}$$

where μ is a constant parameter and $\kappa^{+}(\psi)$ is the positive part of the Gaussian curvature of ψ.

Remarks:

1. The major difference of our model (2.11) from (2.6) is that, instead of putting the factor $A(\psi)$ in the energy as in (2.5), we modify (2.4) directly and only enhance the force field $-\nabla d$. The numerical experiments in Section 2.3.4 show significant improvement of the results using our model (2.11) for the blood vessel shown in Fig. 2.2. More details are discussed in Section 2.3.4.

2. The choice of the positive component (instead of some other choices such as the absolute value) of the Gaussian curvature is to ensure that the resulting surface does not contain any part of the vessels. Indeed, assuming that the initial surface contains part of the blood vessels, and if the vessel locally looks like a cylinder, then its Gaussian curvatures are small, and the part of the surface on the vessel area will shrink and disappear eventually. More often than not, vessels are curved instead of straight as cylinders, as shown in the left figure of Figure 2.7. Since in (2.12) we do not enhance the force field at the region with negative Gaussian curvatures, then the part of the surface on those regions of the vessel will be peeled off quickly, and eventually all the surface parts within vessel regions will shrink and disappear. On the other hand, if the curved vessel is small in diameter, the mean curvature term $\beta\nabla\cdot\frac{\nabla\phi}{|\nabla\phi|}$ dominates $A(\psi)\nabla d\cdot\frac{\nabla\phi}{|\nabla\phi|}$, and the zero level set of ϕ in these regions also shrink. To illustrate the above observations visually, we show in Figure 2.7 the process of evolution when solving (2.11). The left figure of Figure 2.7 superimposes Gaussian curvature of the surface onto the surface itself, where the red curves specifies the regions where the blood vessels are curved, with one side having positive curvature and the other side negative curvatures. The figures from the second to the last show the evolutions of ϕ. As one can see, all the vessels that were included in the initial guess of ϕ disappeared at the final stage.

3. We also note that our model (2.11) can be used in other types of surface capturing problems. We just need to fashion the factor $A(\psi)$ according to the type of surfaces and applications we have.

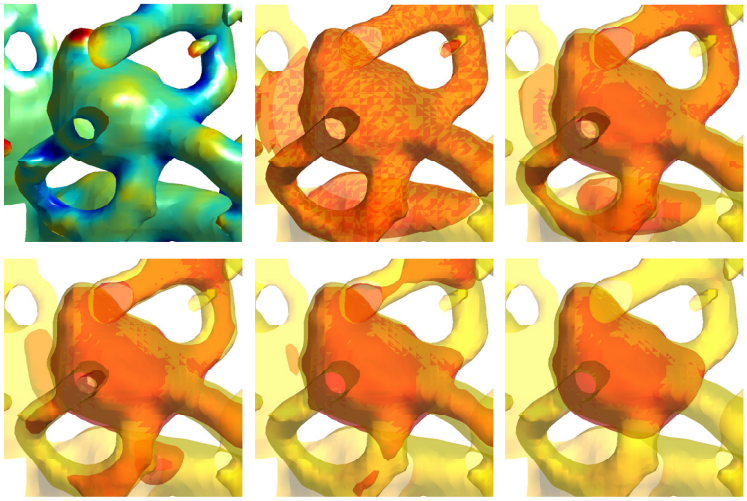


Figure 2.7: Figures from upper left to lower right are: Gaussian curvature on the surface; initial surface (red) superimposed with the reference surface; initial surface; iteration 200; iteration 500; iteration 1000; and final result.

2.3.3 Numerical Implementation and Computations of Geometries

Although our numerical experiments show that our model (2.5) is less restrictive in terms of initialization than the models (2.4) and (2.6), properly chosen initial guesses are still desirable. To obtain a reasonable initial surface, we adopt a user interactive strategy to initiate the computation. We let users to select points around the area of interest and use the selected points to determine a sphere/ellipsoid with level set function ϕ_{S}. Then $\phi(x,0)$ is defined as the intersection of ϕ_{S} with ψ, or mathematically $\phi(x,0) = \min\{\phi_{\mathrm{S}}(x), \psi(x)\}$. In our proposed method, the selection of the points for the region of interest is the only part that needs user interaction. Although automated computation is desirable, determining a pathologic region is a medical diagnosis which needs an expert's supervision. Therefore, it is reasonable to have experts' inputs and use them to initiate the computation. In the numerical section, we will show numerous clinical examples of allowing the user to select only a few (no more than six) points

to capture the surfaces of brain aneurysms.

With the initial condition described above, we employ the local level set method and finite difference discretizations [44] to solve equation (2.4) and (2.11), as well as to minimize (2.5), in order to alleviate the time step restrictions and lower the complexity of our numerical computations. Generally speaking, we solve

$$\phi_t + c_{a,b}(\phi)V_n(\phi)|\nabla\phi| = 0$$

instead of

$$\phi_t + V_n(\phi)|\nabla\phi| = 0$$

with $V_n(\phi)$ the normal velocity depending on ϕ (e.g. $V_n = -\kappa$ for mean curvature motion), and the restriction function $c_{a,b}$ introduced to confine all effective calculations within a narrow band of zero level set of ϕ. The restriction function $c_{a,b}$ is defined as

$$c_{a,b}(x) = \begin{cases} 1, & |x| \le a; \\ \frac{(|x|-b)^2(2|x|+b-3a)}{(b-a)^3}, & a < |x| \le b; \\ 0, & |x| > b. \end{cases} \tag{2.13}$$

There are three parameters in our model (2.11), i.e. μ, α and β. The parameter μ controls the amount of force one wishes to apply near the regions with sharp features. The term $\alpha H(\psi)$ prevents the zero level set of ϕ from passing through that of ψ. Since we initialize our ϕ within ψ, this term only acts as a barricade and we shall fix its value throughout our experiments. The parameter β controls the global smoothness of ϕ. The larger β is, the smoother our final results will be. In Section 2.4, we shall fix all the parameters μ, α and β, and the stopping tolerance. Our numerical results show that the set of parameters we have chosen

gives consistently good results for all of the ten different subjects we tested.

After we obtain the solution ϕ which represents the aneurysm, we calculate its volume using by $V(\phi) = \int H(\phi)dx$, the mean curvature by $\kappa_m(\phi) = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$, and the Gaussian curvature [45] by $\kappa_{\mathrm{g}}(\phi) = \frac{\nabla\phi^T H(\phi)\nabla\phi}{|\nabla\phi|^4}$, where

$$H(\phi) = \begin{pmatrix} \phi_{yy}\phi_{zz} - \phi_{yz}\phi_{zy} & \phi_{yz}\phi_{zx} - \phi_{yx}\phi_{zz} & \phi_{yx}\phi_{zy} - \phi_{yy}\phi_{zx} \\ \phi_{xz}\phi_{zy} - \phi_{xy}\phi_{zz} & \phi_{xx}\phi_{zz} - \phi_{xz}\phi_{zx} & \phi_{xy}\phi_{zx} - \phi_{xx}\phi_{zy} \\ \phi_{xy}\phi_{yz} - \phi_{xz}\phi_{yy} & \phi_{yx}\phi_{xz} - \phi_{xx}\phi_{yz} & \phi_{xx}\phi_{yy} - \phi_{xy}\phi_{yx} \end{pmatrix},$$

and subscripts denote the partial derivatives in Cartesian coordinates. Note that the mean and Gaussian curvatures in Fig. 2.5 are computed using the above formulas.

2.3.4 Models Comparison

The algorithms (2.4), (2.6) and our method (2.11) are applied to a set of brain images acquired by 3D CT angiography. The images have 512×512 in-plane spatial resolutions with each voxel size approximately 0.125mm^3. We then extract subimages of size 54×37 for the aneurysm from the entire brain images. The reconstruction of the surface is shown in Fig. 2.2. The initial surface, i.e. the zero level set of $\phi(x, 0)$, is visualized in Fig. 2.8 (right).

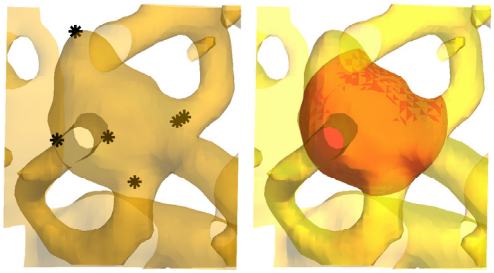


Figure 2.8: Left: selected points on the target vessel; right: initial surface (red).

The numerical results of solving (2.4) are shown in Fig. 2.9, top row. Although this model has been reported with fairly good results for 2D images [36], direct application to capturing 3D surface is not satisfactory, as discussed in Section 2.3.1. Here we also tested the model (2.6) which was developed to improve the illusory contours at corners [36]. The results are shown in the second row of Fig. 2.9. This model provides improvement at the tip of the aneurysm in comparison with the model (2.4); however, it still can not capture the entire tip which is a very important medical feature. The reason is the concavity change near the sharp tip, which is consistent with our earlier discussion in Section 2.3.1. The results of our surface capturing model (2.11) are shown in the third row of Fig. 2.9. Using our method (2.11), we are able to capture the entire aneurysm.

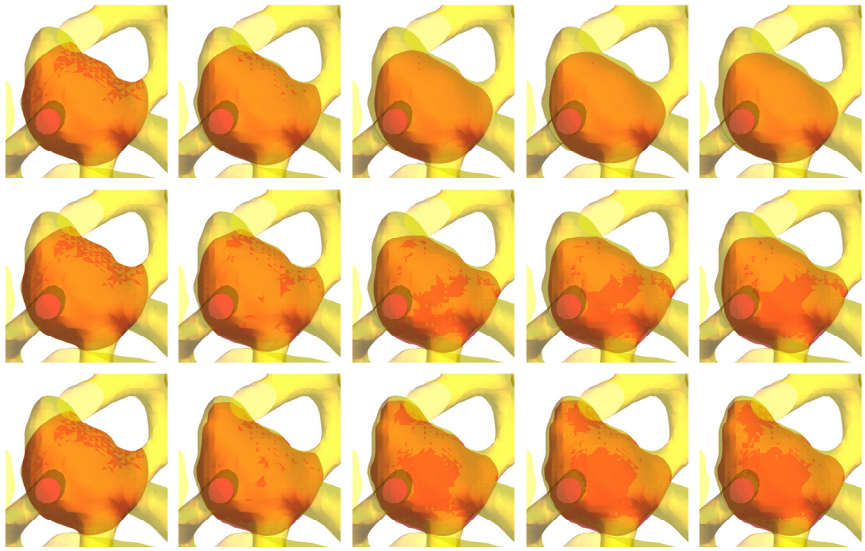


Figure 2.9: Row 1-3 shows results of (2.4), (2.5) and (2.11) respectively. For the visually best results, parameters are $\beta = 1$ for (2.4), (μ, β)=$(500, 0.05)$ for (2.5) and (μ, β)=$(2700, 0.05)$ for (2.11). In each row, the five figures are results at iteration=0, 100, 500, 1000 and 2000 respectively.

2.4 Validations of Our Method on Various Brain Aneurysm Data

In this section, we further validate our model (2.5) on ten different brain aneurysms. Brain aneurysms are classified as the narrow-neck aneurysms or wide-neck aneurysms by their dome/neck ratios. A narrow-neck aneurysm has a dome/neck ratio more than 1.5; otherwise, it is a wide-neck aneurysm [46]. We shall test the consistency and robustness of our method on both types(five subjects for each category). Throughout the numerical experiment, parameters μ, α and β are taken as $\alpha = 0.5$, $\beta = 0.001$, and $\mu = 2/\text{mean}(\kappa^+(\psi))$. Only the stopping criteria are different depending on whether the aneurysm is classified as narrow-neck or wide-neck. Numerical results presented in Fig. 2.10 to Fig. 2.13 show that this choice of parameters and stopping criteria gives consistently good results.

All the numerical experiments were performed using MATLAB on a Windows Laptop (Duo processor, 2.0GHz CPU and 2GB RAM). It took approximately one minute to capture an aneurysm with volume 100mm^3, and an additional one minute for every 100mm^3 increase in size.

2.4.1 Narrow-Necked Aneurysms

We test our model (2.11) on five narrow-neck aneurysms data. The reconstructed surfaces from 3D images are given in the top row of Fig. 2.10. We initialize our computation and perform the calculation by the algorithm in Section 2.3.2 and 2.3.3, and employ the following stopping criteria for narrow-necked aneurysms

$$\frac{\|\phi^{n+1} - \phi^n\|_2}{\|\phi^n\|_2} < \text{tolerance}, \tag{2.14}$$

where n is the iteration number which comes from the discretization of time variable t. Fig. 2.10 (bottom row) shows the numerical results of aneurysm capturing for the five subjects. The robustness of the numerical solutions is also tested by randomly choosing 6 different sets of initial points (Fig 2.11 top row) on one of five aneurysms in Fig. 2.10, which generates 6 different initial surfaces (Fig 2.11 middle row). The final results from the 6 different initializations are nearly identical to each other as shown in Fig. 2.11 bottom row. The volumes captured by different initial points are $312.273 \pm 6.245\text{mm}^3$ (mean $\pm$standard deviation). As a result, we expect the deviation of the volume computation which can be caused by different users is approximately 2% of the total aneurysm volume, which can be considered well acceptable in practice.

2.4.2 Wide-Necked Aneurysms

For wide-neck aneurysms, using (2.14) as stopping criteria may cause the zero level set of ϕ shrink to zero. This is because in general there does not exit a stable state for the zero level set of ϕ due to the fact that the necks of the aneurysms are usually wide open. Thus, we adopt the following stopping criteria based on the special geometry of wide-neck aneurysms,

$$\frac{\|\phi^{n+1} - \phi^n\|_2}{\|\phi^n - \phi^{n-1}\|_2} \approx 1. \tag{2.15}$$

The above equation means that the computation stops whenever the change of ϕ^n picks up some constant pace. We test our model (2.11) with the above stopping criteria (2.15) on five wide-neck aneurysms data. The reconstructed surfaces from 3D images are given in Fig. 2.12 top row. Fig. 2.12 bottom row shows the numerical results of aneurysm capturing. To test the robustness, we also randomly choose 6 different sets of initial points on one of five aneurysms in

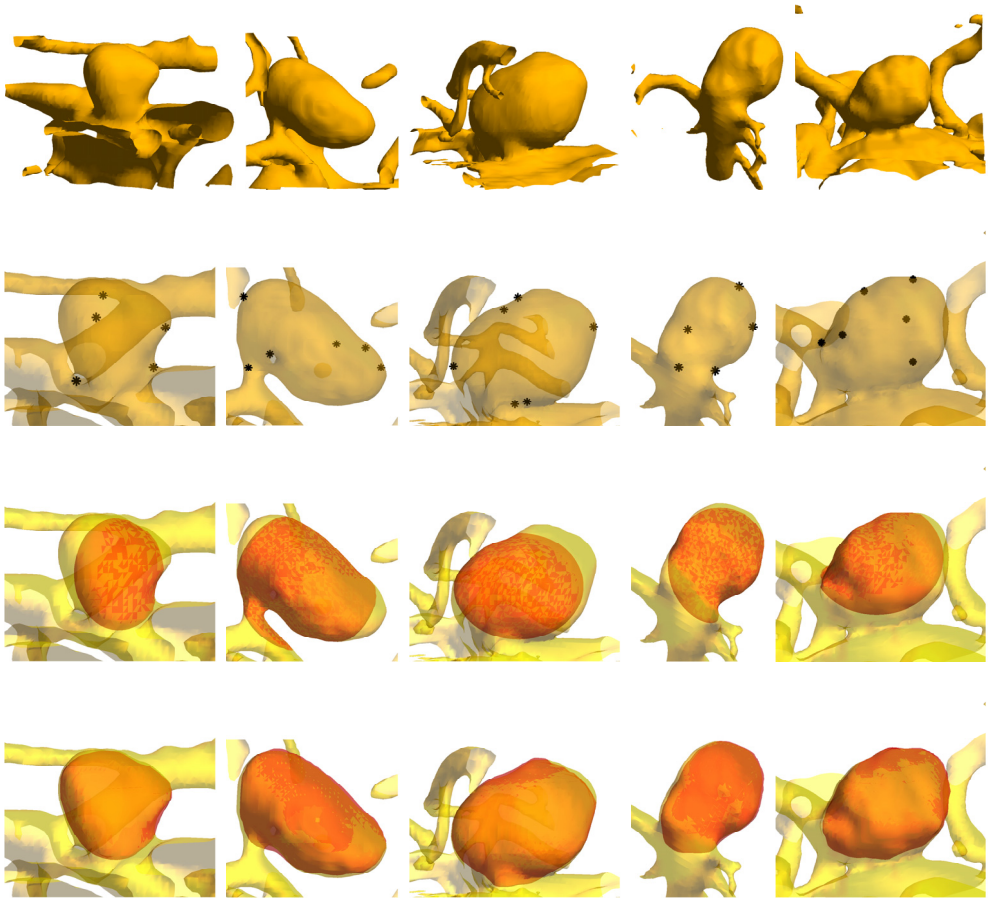


Figure 2.10: Top row shows the surfaces of narrow-necked aneurysms. Second row shows the sets of points given by users. Third row is the corresponding initial surfaces. Bottom row is the corresponding final captured surfaces. The surfaces in row 2-4 are shown with close-up views. The volumes of the aneurysms captured are 213.527mm^3, 520.196mm^3, 602.7mm^3, 319.296mm^3 and 516.399mm^3 respectively from left to right.

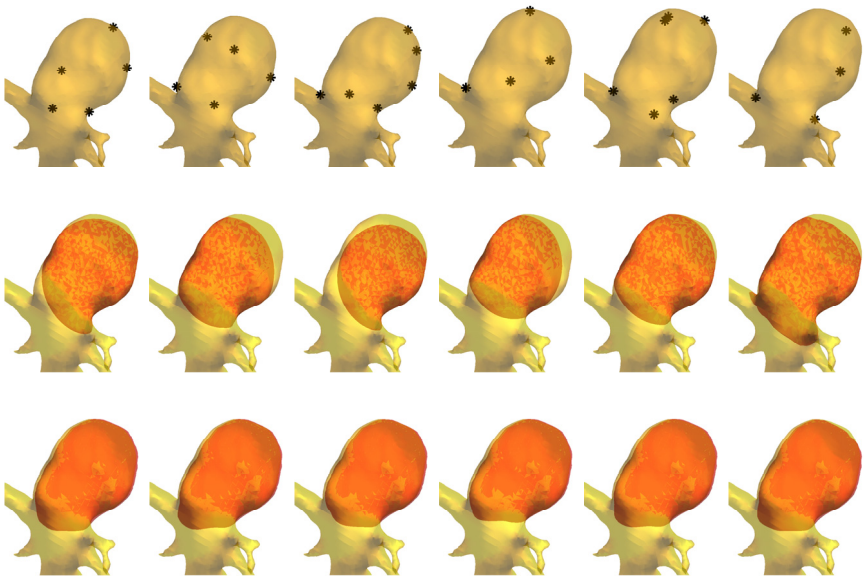


Figure 2.11: Top row is the set of points given by users. Middle row is the corresponding initial surfaces. Bottom is the corresponding final captured surfaces. The resulting volumes for the different points chosen by users from left to right are: 319.296mm^3, 317.275mm^3, 307.781mm^3, 302.881mm^3, 315.499mm^3 and 310.905mm^3.

Fig. 2.12, which generates 6 different initial surfaces. The final results from the 6 different initializations are also nearly identical as one can see in Fig. 2.13 bottom row. The volumes captured by different initial points are $122.42 \pm 5.37\text{mm}^3$ (mean $\pm$standard deviation). As a result, we expect the deviation of the volume computation which can be caused by different users is approximately 4.4% of the total aneurysm volume. Note that, although we do not have a theoretical guarantee that (2.15) works for all wide-neck aneurysms, we believe it should work for typical wide-necks and it certainly works well for the 5 subjects we tested on in Section 2.4.2.

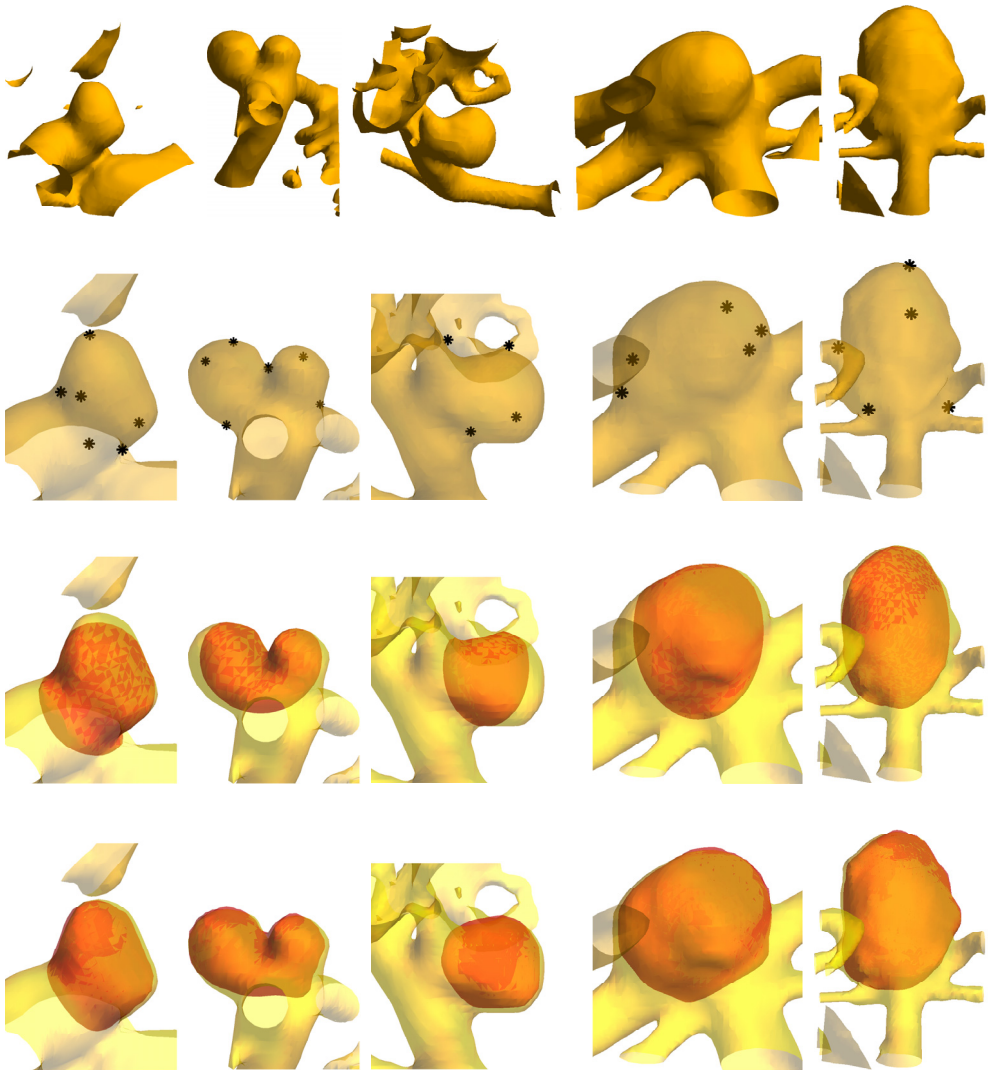


Figure 2.12: Top row shows the surfaces of narrow-necked. Second row shows the sets of points given by users. Third row is the corresponding initial surfaces. Bottom row is the corresponding final captured surfaces. The surfaces in row 2-4 are shown with close-up views. The volumes of the aneurysms captured are 78.767mm^3, 95.823mm^3, 117.355mm^3, 300.493mm^3 and 748.23mm^3 respectively from left to right.

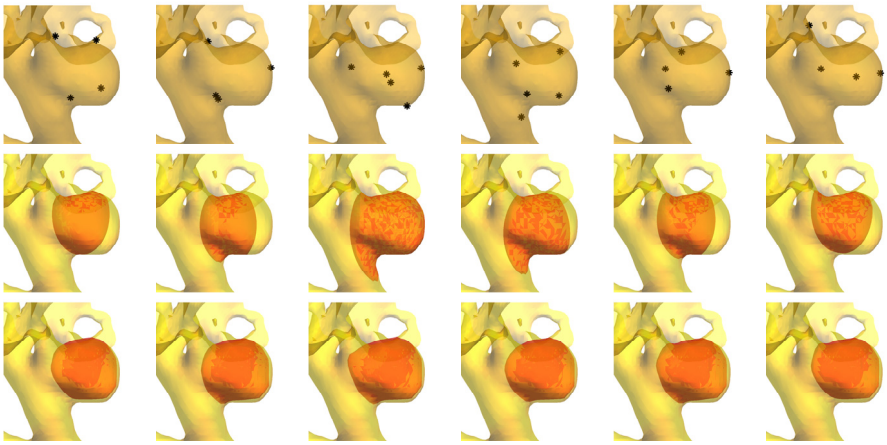


Figure 2.13: Top row is the set of points given by users. Middle row is the corresponding initial surfaces. Bottom is the corresponding final captured surfaces. The resulting volumes for the different points chosen by users from left to right are: 117.355mm^3, 122.133mm^3, 131.136mm^3, 122.5mm^3, 124.95mm^3 and 116.436mm^3.

CHAPTER 3

Multiscale Representation (MSR) for Shapes and Its Applications in Medical Image Analysis

3.1 Introduction

Multiscale representation (MSR in short) of functions, e.g. wavelets, has been well studied in the past twenty years [53, 54]. Theories and numerical algorithms have been developed extensively. However, when one deals with shapes, especially shapes in $\mathbb{R}^3$, most of the previous theories and algorithms cannot be extended straightforwardly. One fundamental problem and major difficulty is the fact that it is generally hard to associate each shape with a function such that the geometric properties of the shape can be naturally characterized by the function, and the MSR decomposition of such function reveals interesting geometric information that is intrinsic to the shape itself.

Many attempts have been made on designing wavelet-typed MSR for biological shapes, among which, the method proposed by Nain et. al. [66, 67] is rather effective. Their basic idea is to first map the shape (triangulated) onto the unit sphere so that one obtains a vector-valued function $f : S^2 \mapsto \mathbb{R}^3$; then apply spherical wavelet decomposition [72] to each component of f. However, the wavelet coefficients obtained from their approach are not intrinsic to the shape, but rather dependent on the mapping f (in other words, the spherical

parametrization). Furthermore, finding a good mapping from a shape to the unit sphere (or to some other canonical domains) is nontrivial and in fact a popular ongoing research area [58–65, 70].

More recently, Pauly et. al. [86] introduced an MSR for point-based surfaces. Their idea was to use Moving Least Square method [83] to define a series of smoother and smoother point-based surfaces, and then define wavelet coefficients as the displacements from two successive levels. Their method only requires a local parametrization of the point-based surface which is easy to calculate. However, the application of their method is rather limited in medical image analysis, because most of the biological shapes are not point-based.

Motivated by Pauly et. al.'s work, we shall introduce a new MSR framework in Section 3.2 based on level set motions via solving some properly chosen Hamilton-Jacobi (HJ) like equation (in contrast to the Moving Least Square method used in Pauly et. al.'s MSR [86]). We shall leave the details to Section 3.2. In section 3.3, we will apply our MSR framework to surface inpainting problems for both phantom and real biological shapes. A few remarks on applying the MSR in Section 3.2 to images and functions will be made in Section 3.4.

3.2 Level Set Based MSR for Shapes

In this section, we will propose a new MSR for shapes. The basic idea is using level set motions via solving some properly chosen HJ like equation to obtain a sequence of shapes that become smoother and smoother as time evolves (analogous to coarse level approximation in wavelet decomposition). Then we carefully define the so-called "details" (analogous to, e.g. wavelet coefficients) of the MSR which carry important geometric information and facilitate a perfect reconstruc-

tion. While the wavelet based multi scale decomposition and reconstruction use filters, which are linear processes, the proposed new MSR for shapes uses (non-linear) PDEs for both decomposition and reconstruction. However, the spirit is the same, i.e. separate features from smooth components of the surface. Due to the level set formulation, parametrization is no longer needed.

Throughout this section, shapes are defined to be smooth boundaries of domains $\Omega \in \mathbb{R}^3$ and are represented by level set functions, typically signed distance functions. A level set function ϕ that represents the shape $\partial\Omega$ is defined as follows

$$\phi(x) \begin{cases} < 0 & x \in \Omega; \\ > 0 & x \in \Omega^c. \end{cases}$$

We always assume that the function ϕ is at least Lipschitz continuous.

Level set motions can be achieved by solving the following HJ like equation [95],

$$\phi_t + v_n\left(\nabla\phi\right)|\nabla\phi| = 0, \quad \phi(x,0) = \phi_0(x), \tag{3.1}$$

where we take $(x,t) \in \mathcal{D} \times [0,T]$ with $\mathcal{D}$ some bounded domain in $\mathbb{R}^3$ and $T > 0$. Here $v_n\left(\nabla\phi\right)$ is the normal velocity, which essentially depends on $\nabla\phi$ while second order derivatives of ϕ may be involved (e.g. mean curvature). If it only depends on first order derivatives, then it is a HJ equation. We also assume that the PDE (3.1) is geometric [103, 104], which is satisfied for the velocity fields that we will focus in this section. Comprehensive theoretical analysis of PDE (3.1) and surface evolution equations can be found in [96, 97, 99–104, 106].

The velocity fields that we shall focus here are

$$v_n = c(x) - \lambda\kappa, \ \lambda > 0 \quad \text{and} \quad v_n = \kappa_a - \kappa, \tag{3.2}$$

where $c(x)$ is some smooth function, κ is the mean curvature defined as $\kappa := \nabla \cdot \frac{\nabla \phi}{|\nabla \phi|}$, and κ_a is the average mean curvature. Note that for $v_n = \kappa_a - \kappa$, the PDE (3.1) generates an volume preserving mean curvature motion [82, 89, 93, 105].

3.2.1 Continuous Transformations and Discrete Algorithms

Let $\Omega_t \in \mathbb{R}^3$ be some domain with scale t, and $S_t := \partial \Omega_t$ be the shape at scale t represented by some time-dependent level set function $\phi(x,t)$, i.e. $\phi(x,t) < 0$ for $x \in \Omega_t$, $\phi(x,t) > 0$ for $x \in \Omega_t^c$, and

$$S_t = \{x \in \mathbb{R}^3 \mid \phi(x,t) = 0\}_{t \geq 0}. \tag{3.3}$$

Here S_0 denotes the original shape with the corresponding level set function $\phi_0(x) = \phi(x,0)$. Throughout the rest of the section, the function $\phi(x,t)$ is always taken to be the solution of (3.1). For some properly chosen v_n in (3.1), e.g. with $v_n = -\kappa$ or $\kappa_a - \kappa$, we can obtain a continuous series of shapes $\{S_t\}_{t \in [0,T]}$, which tends to become smoother when t increases. Based on this, we define our continuous level set based MSR of S_0 as follows.

Definition 3.2.1. *Let $\phi(x,t)$ be the solution of the PDE (3.1) and $(x,t) \in \mathcal{D} \times [0,T]$. We now understand $x_l(t)$ as a path on the propagating l-th level set of ϕ, i.e. $\phi(x_l(t),t) = l$. For simplicity, we shall omit the subscript "l" unless a particular level set is considered.*

1. *We now define the **multiscale transformation (MST)** of $\phi_0(x)$ as*

$$\vec{W}(x,t) := W(\phi_0) := -v_n \frac{\nabla \phi}{|\nabla \phi|} = -x'(t). \tag{3.4}$$

*Vector $-x'(t)$ is the **displacement vector** and $w(x,t) := -v_n(x,t)$ is the*

detail *of the MST.*

2. *We shall call* $\vec{W}(x,t)$ *the* ***displacement vector field*** *at scale* t*, and denote* $\vec{W}_|(x,t)$ *(*$w_|(x,t)$*) as the restriction of* $\vec{W}(x,t)$ *(*$w(x,t)$*) on* S_t*.*

3. *The MSR for the original shape* S_0 *in terms of* $\phi_0(x)$ *is denoted as*

$$MSR(\phi_0) = \Big\{\{\vec{W}(x,t)\}_{t\in(0,T)}, \phi(x,T)\Big\}.$$

4. *We define the* ***inverse multiscale transformation (IMST)*** *via solving the following PDE*

$$\psi_\tau + \vec{W}(x, T-\tau)\cdot\nabla\psi = 0, \quad \psi(x,0) = \phi(x,T). \tag{3.5}$$

for given $T > 0$ *and* $0 \le \tau \le T$*.*

Remark 3.2.2.

1. *The technique of generating a sequence of the spaces* $\{S_t\}$ *via solving PDEs is known as scale space decomposition (see e.g. [47, 48]). However, a classical scale space analysis does not study the details as defined above.*

2. *The last identity in* (3.4) *can be shown by taking the directive of* $\phi(x(t),t) = l$ *w.r.t.* t *and using the PDE* (3.1) *and the assumption that* $x'(t)$ *is aligned with normal directions of level sets of* ϕ*. Indeed, taking derivative with respect to* t *at* $\phi(x(t),t) = l$*, one obtains*

$$\nabla\phi\cdot x'(t) + \phi_t = 0.$$

Using equation (3.1)*, we have*

$$\nabla\phi \cdot x'(t) = H(\nabla\phi).$$

Assuming that $x'(t)$ *is in the same direction as* $\nabla\phi$*, we let* $x'(t) = A\nabla\phi$ *and then we have*

$$A|\nabla\phi|^2 = H(\nabla\phi).$$

Solving A *from the above equation, we obtain the desired expression for* $x'(t)$*.*

3. *The detail* $w_|(x,t)$ *is a function on* S_t *that characterizes intrinsic geometric information of the shape at scale* t*. Here by intrinsic we mean that* $w_|(x,t)$*, as well as* $\{S_t\}_{t>0}$*, does not depend on the initial embedding* ϕ_0 *for a large class of functions [99, 103], but only depends on* S_0*. Therefore, we now have an intrinsic MSR for* S_0*:*

$$MSR(S_0) = \Big\{\{\vec{W}_|(x,t)\}_{t\in(0,T)}, S_T\Big\}. \tag{3.6}$$

4. *The MSR defined above can be easily adapted to a point-based or triangulated surface. One simply need to first associate the surface with a level set function and then perform the MST. For point-based surfaces, the IMST from its MSR* (3.6) *can be point-wise defined as* $S_0 = S_T + \int_0^T \vec{W}_|(x,t)dt$ *or equivalently* $x_0(0) = x_0(T) + \int_0^T -x_0'(t)dt$*, which is obviously true.*

5. *For different purpose of application, one can choose different* v_n*. For example, if one seeks a sparse MSR of shapes, it makes more sense to use the motion with* $v_n = \kappa_a - \kappa$ *than* $v_n = -\kappa$*. To see this intuitively, let us imagine the shape* S_0 *is the unit sphere. It is a perfectly smooth and simple*

shape and its MSR should not produce any details. If $v_n = \kappa_a - \kappa$, $w(x,t)$ is actually equal to zero [105], while for $v_n = -\kappa$, $w(x,t)$ will never be zero, because S_t will keep shrinking and vanishes in finite time [99, 103]. Generally speaking, different velocity field v_n means different type of MSR of the original shape S_0, which mimics different choices of wavelet representation of functions.

Now the question is that if we have perfect reconstructions via (3.5). The answer is given in the following proposition, which directly follows from theories of ODEs.

Proposition 3.2.3. *Assume that $\vec{W}(x,t)$ stays Lipschitz continuous for $(x,t) \in \mathcal{D} \times [0,T]$. Then the equation* (3.5) ***inverts*** *the MST defined by* (3.4) *in the sense that $\psi(x,\tau) := \phi(x, T-\tau)$ is the unique solution of* (3.5).

Remark 3.2.4. *The assumption in Proposition 3.2.3 is not always valid (e.g. $v_n = c < 0$ and $\phi_0(x)$ representing a cube). However, if we choose, for example, $v_n = -\kappa$, $v_n = \kappa_a - \kappa$ or $v_n = c(x) - \lambda\kappa$ (for some smooth $c(x)$ and $\lambda > 0$), and choose some appropriate ending time $T > 0$ (e.g. before any topological changes occur), the above assumption will be valid and we will have a perfect reconstruction using* (3.5) *[104, 105].*

Notice from Definition 3.2.1 and Proposition 3.2.3 that to perfectly reconstruct $\phi_0(x)$ from $\phi(x,T)$, we need to store the entire vector field $\vec{W}(x,t)$ for every $x \in \mathcal{D}$ and all scale t. However, in practice, we only want a perfect reconstruction of S_0, and thus we do not need that much information. In fact, we should only store the displacement vectors within a narrow band of the zero level set of $\phi(x,t)$ for each scale t.

We can be even more "greedy" here by only storing $\vec{W}_|(x,t)$ which lives on

the zero level set of $\phi(x,t)$. As for the inverse transform, we need to extend $\vec{W}_|(x,t)$ to at least a narrow band of the zero level set of $\phi(x,t)$. Note that no extension can guarantee an exact recovery of the vector field $\vec{W}(x,t)$, and hence the reconstruction will not be exact. However, if the extension is conducted accurately and when the mesh grid is dense enough, i.e. the resolution of the shape is high enough, the reconstruction should be more and more accurate. The extension we shall adopt here is such that the extended vectors are constant in the normal directions of each level set of $\phi(x,t)$ [110]. In our experiments below, we shall simply use a local search method to extend $\vec{W}_|(x,t)$.

We now propose the discrete version of MSR in the following Algorithm 1.

Algorithm 1 Level Set Based MST and IMST

Start from the given level set function $\phi_0(x)$ representing shape S_0. Choose time steps $0 = t_0 < t_1 < \ldots < t_N = T$, where $\max_i(t_{i+1} - t_i)$ is small.

Initialize: Sample a point set X_0 from S_0 (either uniformly or non-uniformly).

MST:

while $i \leq N$ **do**

1. Starting from $\phi(x, t_{i-1})$, solve (3.1) for $t \in [t_{i-1}, t_i]$ and obtain $\phi(x, t_i)$.

2. Orthogonally project X_{i-1} onto the zero level set of $\phi(x, t_i)$ and obtain X_i.

2. Compute the discrete displacement vector by $\vec{W}_{|i} = X_i - X_{i-1}$, and $i \leftarrow i + 1$.

end while

We then obtain the discrete MSR of S_0:

$$\mathrm{MSR}(S_0) := \{\vec{W}_{|1}, \vec{W}_{|2}, \ldots, \vec{W}_{|N}, \phi(x,T)\}.$$

IMST:

1. Extend the vector fields $\{\vec{W}_{|i}\}_{i=1}^N$ such that the values are constant along normal directions of the level sets of $\phi(x, t_i)$.

2. Solve (3.5) using $\vec{W}_{|i}$ within interval $[t_i, t_{i-1}]$ iteratively for each i.

3.2.2 Numerical Experiments on the MSR

One of the key steps of implementing Algorithm 1 is to solve the evolution PDE (3.1) efficiently. There are many ways of solving equation (3.1). The most straightforward way is to use monotone finite difference schemes [95, 174]. However, it is not very efficient computationally. To overcome this, Merriman, Bence and Osher introduced a diffusion-based level set motion in [87, 88], and it was further studied in [84, 90–92], where in [84] the correctness of the method is rigorously proven. In [89], Ruuth and Wetton introduced a fast algorithm to calculate volume preserving motion by mean curvatures. All these methods speeded up curvature driven motions drastically.

In this section, we will recall the fast algorithms of level set motion for the case $v_n = c$ and $v_n = \kappa_a - \kappa$ given by [87–89, 91]. These algorithms will be used in the later sections to generate fast multiscle decompositions of shapes.

Now we first recall the fast method of solving (3.1) with $v_n = c$ (see [87, 88, 91]) in Algorithm 2.

Algorithm 2 Level Set Motion with Constant Normal Velocity

Start from a given shape represented by ϕ.

while $t < T$ **do**

1. Define the corresponding characteristic function by $\chi = \mathbf{1}_{\{\phi<0\}}$. Set V_0 equal to the volume of $\{\phi < 0\}$.

2. Starting from χ, evolve $\bar{\chi}$ for a time Δt by $\bar{\chi}_t = \nabla^2 \bar{\chi}$.

4. Sharpen:

$$\chi = \begin{cases} 1 & \text{if } \bar{\chi} > 0 \\ 0 & \text{otherwise} \end{cases}$$

5. Let $t \leftarrow t + \Delta t$. Compute $\phi(x, t)$ from χ via fast sweeping method [18].

end while

We now recall the fast implementation of (3.1) with $v_n = \kappa_a - \kappa$ proposed by Ruuth and Wetton [89] in Algorithm 3. Their algorithm is based on the diffusion-

based mean curvature motion proposed by [87, 88]. Note that if we remove step 3 in Algorithm 3 and choose $\lambda = 0.5$ in step 4, it is exact the fast mean curvature motion proposed in [87, 88].

Algorithm 3 Volume Preserving Mean Curvature Motion: $v_n = \kappa_a - \kappa$.

Start from a given shape represented by ϕ.

while $t < T$ **do**

1. Define the corresponding characteristic function by $\chi = \mathbf{1}_{\{\phi<0\}}$. Set V_0 equal to the volume of $\{\phi < 0\}$.

2. Starting from χ, evolve $\bar{\chi}$ for a time Δt by $\bar{\chi}_t = \nabla^2 \bar{\chi}$.

3. Determine the threshold value that preserves the volume of the set: i.e. find a $0 < \lambda < 1$ s.t.

$$\Big| |\{x : \bar{\chi} < \lambda\}| - V_0 \Big| < \varepsilon.$$

4. Sharpen:

$$\chi = \begin{cases} 1 & \text{if } \bar{\chi} > \lambda \\ 0 & \text{otherwise} \end{cases}$$

5. Let $t \leftarrow t + \Delta t$. Compute $\phi(x, t)$ from χ via fast sweeping method [18].

end while

Some numerical results of the MST and IMST in Algorithm 1 are presented in Figure 3.1 using the tested shape (right hemisphere of a cortex). The velocity field is chosen to be $v_n = \kappa_a - \kappa$ and 5 levels of decomposition are conducted (first and second row of Figure 3.1). The IMST is also presented in Figure 3.1 where $\tilde{S}_i$ denotes the reconstruction of level i from level $i + 1$. As we can see, although the reconstructions are not exact for each level, they are quite accurate in the sense that most of the features are well reconstructed.

3.3 Application of Level Set Based MSR in Surface Inpainting

Inpainting problem, for both images and surfaces, has been extensively studied in the literature (see e.g. [107, 112–131]). It occurs when part of the data in an

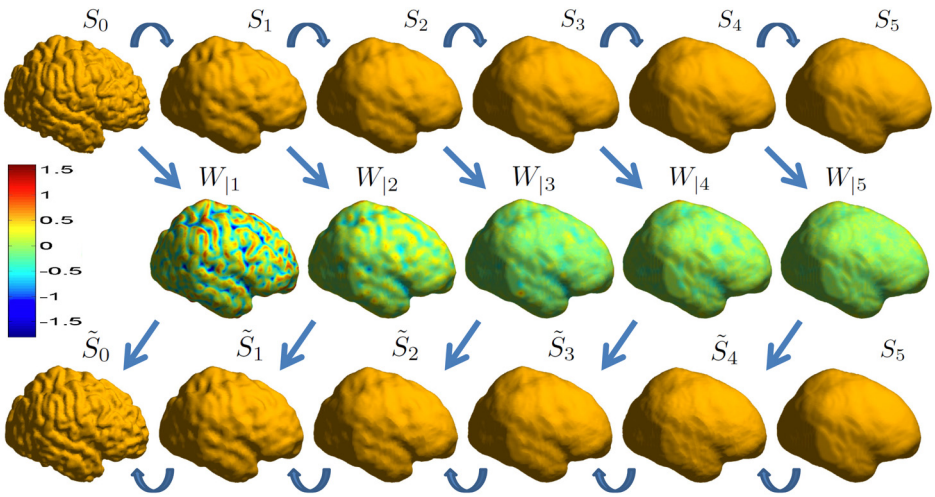


Figure 3.1: First row (left to right): MST $S_0, S_1, \ldots, S_5$. Second row shows the details of MSR on $S_1, \ldots, S_5$. Third row shows IMST $\tilde{S}_i$, $i = 0, 1, \ldots, 4$, where the Hausdorff distance between S_i and $\tilde{S}_i$ are: $1.12h$, $0.74h$, $0.74h$, $0.69h$, and $0.63h$ respectively (with h the mesh size).

image or regions of a surface is missing or corrupted. The major task of inpainting is to fill in the missing information based on the geometry of the image/surface.

In this section, we will propose a new surface inpainting algorithm based on the MSR in Section 3.2.1 for blood vessel reconstruction that arises in medical image analysis.

Our surface inpainting algorithm (Algorithm 4 below) inherits the following framelet-based image inpainting structure proposed by Cai et. al. [123]:

1. Take framelet transform of the given image;
2. Truncate the framelet coefficients via soft-thresholding and reconstruct;
3. Apply the exact data outside the inpainting domains, and repeat.

Since we already have an MSR for surfaces, the first step above can be replace by our MST. As for a mimicking thresholding scheme, we shall solve the following

PDE for IMST instead of the PDE (3.5) that was originally proposed in Definition 3.2.1:

$$\psi_\tau + \vec{W}(x, T-\tau) \cdot \nabla\psi = \varepsilon\nabla^2\psi, \quad \psi(x,0) = \phi(x,T). \tag{3.7}$$

The above PDE mimics thresholding in the sense that it penalizes the reconstruction from $\vec{W}$ by introducing a vanishing viscosity $\varepsilon\nabla^2\psi$, which forces some information outside the inpainting region to flow into the inpainting regions. Also, when $\varepsilon \to 0$, the solution of (3.7) converges uniformly to the viscosity solution of (3.5) [97, 106].

Since we generally expect volumes of surfaces to increase during inpainting, we choose the following PDE for the MST of Algorithm 1,

$$\phi_t + (c + \kappa_a - \kappa)|\nabla\phi| = 0, \quad \phi(x,0) = \phi_0(x), \quad c > 0. \tag{3.8}$$

Note that the PDE (3.8) generates a curvature-related motion with increasing volumes of the domains enclosed by level sets of $\phi(x,t)$. The constant c can be regarded as a parameter that need to be adjusted according to different surface inpainting scenarios. In our experiments, we solve PDE (3.8) via a combination of Algorithm 2 and Algorithm 3.

Algorithm 4 Surface Inpainting via MSR

Start from ϕ_0, with inpainting region D. Choose some $\varepsilon > 0$.
while "Not converge" **do**
1. Perform discrete MST by solving (3.8) and acquire $\vec{W}_{|i}$ by Algorithm 1.
2. Perform IMST by solving (3.7) and obtain ψ_ε.
3. Copy the known information to ψ_ε: $\psi_\varepsilon|_{D^c} \leftarrow \psi_0|_{D^c}$.
4. Decrease amount of smoothing: $\varepsilon \searrow$.
end while

We test Algorithm 4 on both phantom (first two vessels in Figure 3.2) and real (last two vessels in Figure 3.2) surface inpainting scenarios. First row of Figure 3.2

shows four blood vessels with inpainting regions specified by red circles. For the two phantom inpainting scenarios, the inpainting regions are created manually, and the surface within those regions were chopped off. For the two real inpainting scenarios, we do not know the exact inpainting regions. Therefore in practice, we adopt a user interactive strategy to determine the inpainting regions. After several points have been selected on the surface, the inpainting regions are then generated automatically. Inpainting results are given in second and third row of Figure 3.2. We want to point out that during the inpainting process, topological change may occur for some cases (e.g. second vessel in Figure 3.2). Although it violates the assumption in Proposition 3.2.3, topological change is still allowed for inpainting problems. The reason is that perfect reconstruction is only required at the very last stage of inpainting (i.e. when $\varepsilon \approx 0$) in order to ensure convergence, while topological changes will happen during the middle of the process if the parameters (e.g. c in (3.8)) are properly chosen.

3.4 Level Set Based MSR for Images: A Few Remarks

The ideas of the MSR in Section 3.2.1 can also be applied to images (in fact, all functions in any spatial dimension). Denote the original image by u_0. Then the MST, or the scale space decomposition [47, 48], can be obtained by solving

$$u_t + H(D^2u, Du, u) = 0, \quad u(x, 0) = u_0(x). \tag{3.9}$$

Since we now have a canonical domain, i.e. $\mathbb{R}^2$, for the function $u(x, t)$, it is much easier to define displacement vectors. The algorithm is summarized in Algorithm 5. Two numerical examples on the MSR given by Algorithm 5 are presented in Figure 3.3 and Figure 3.4, where 6 levels of decomposition were conducted.

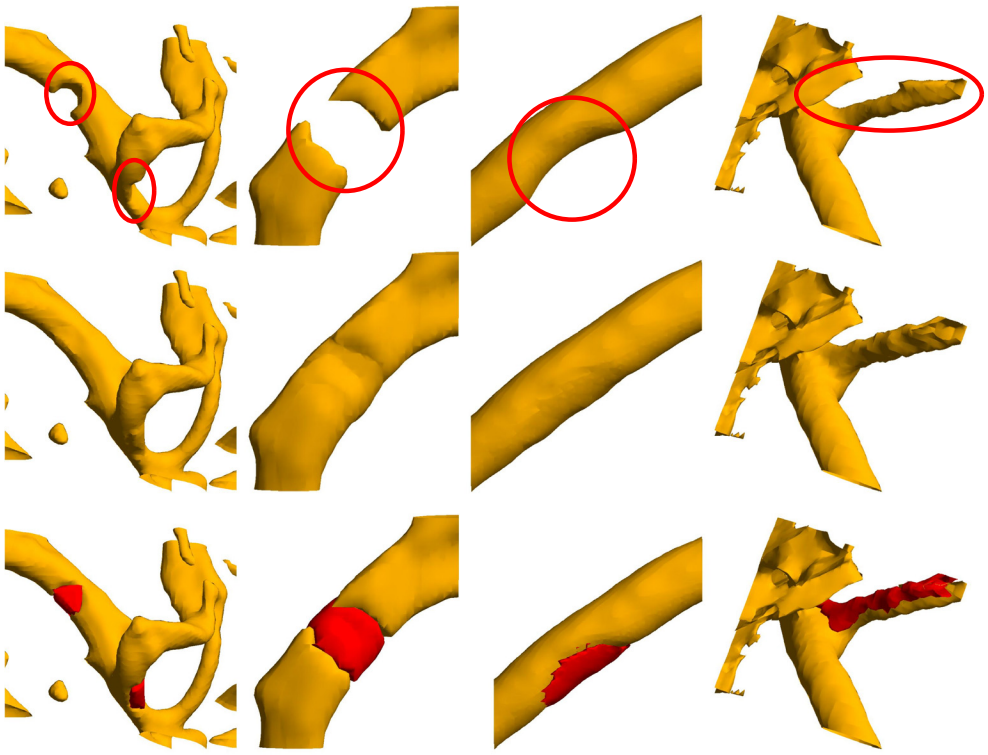


Figure 3.2: Experiments on blood vessel inpainting. Row 1: vessels before inpainting; row 2: vessels after inpainting; row 3: inpainted regions shown in red. The percentage of the volume of inpainted region over that of the entire shape are: 5.3%, 19.2%, 6.7% and 5.7%.

Algorithm 5 MST and IMST for Images

Start from a given image u_0. Let $i = 0$ and choose $0 = t_0 < t_1 < \ldots < t_N = T$.

MST:

while $i < N$ **do**

1. Solve PDE (3.9) for time $[t_i, t_{i+1}]$.

2. Compute the displacements by $v_i = u(x, t_i) - u(x, t_{i+1})$.

3. $i \leftarrow i + 1$.

end while

The above procedure describes the discrete MST of u_0:

$$\mathrm{MST}(u_0) := \{v_1, v_2, \ldots, v_N, u_T\}.$$

IMST: Perfect reconstruction from $\mathrm{MST}(u_0)$ to u_0 (the discrete IMST) is trivial:

$$u_0 = u_T + v_N + \cdots + v_1.$$

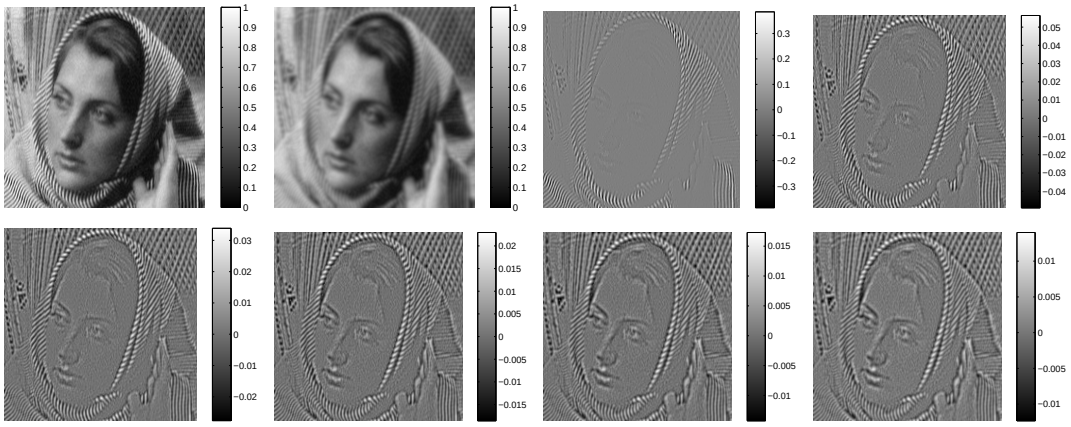


Figure 3.3: Images from upper left to lower right are: original image u_0; u_6, the low frequency approximation of u_0; and v_1, v_2, ..., v_6.

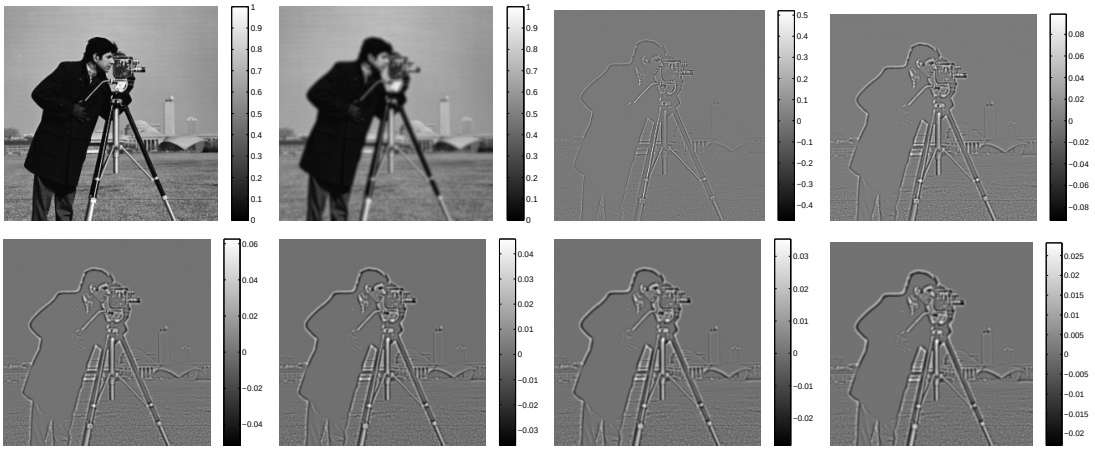


Figure 3.4: Images from upper left to lower right are: original image u_0; u_6, the low frequency approximation of u_0; and v_1, v_2, ..., v_6.

CHAPTER 4

Fast Numerical Methods for Compressive Sensing and ℓ_1-Minimizations via Bregman Iterations

4.1 Introduction

A compressive sensing (CS) problem is to recover a sparse signal $u \in \mathbb{R}^n$, with $\kappa := \|u\|_0 = |\text{supp}(u)| \ll n$, from some random measurements $f \in \mathbb{R}^m$ satisfying

$$Au = f,$$

where $A \in \mathbb{R}^{m\times n}$ is the measurement matrix with $n > m$. The rank of the matrix A could be $\leq m$ in general. However, for most measurement matrices used in practice, A is always full rank. Therefore, for simplicity, we shall assume that $\text{rank}(A) = m$ throughout this chapter.

The most straightforward way to solve the CS problem is to consider the following minimization problem:

$$\min_{u\in\mathbb{R}^n}\{\|u\|_0 | Au = f\}, \tag{4.1}$$

which can be solved approximately by matching pursuit [134–136]. However,

the matching pursuit method does not guarantee to obtain a global minimizer of (4.1) in general. After all, the problem (4.1) is not convex and is difficult to solve. Also, the matching pursuit method is not computationally efficient when the size of the measurement matrix A is huge, which is usually the case in practice.

An alternative and more popular way to solve the CS problem is to consider the following standard ℓ_1-minimization problem:

$$\min_{u\in\mathbb{R}^n}\{\|u\|_1 | Au = f\}. \tag{4.2}$$

There has been a recent burst of research in compressive sensing, which involves solving (4.2). This was led by Candès et.al. [141–143], Donoho, [144], and others (see [138–140] for extensive references). The correctness of the solution from (4.2) is guaranteed by the following theorem by Candès and Tao [141]:

Theorem 4.1.1. ***Restricted Isometry Property (RIP):*** *If a measurement matrix A is a restricted isometry, i.e. if there exists δ_κ and $0 < \delta_\kappa < 1$ s.t.*

$$(1-\delta_\kappa)\|u\| \le \|Au\| \le (1+\delta_\kappa)\|u\|,$$

for all 3κ-vector u, then we can recover the sparse signal u from (4.2)*.*

Judging from the definition of RIP, it is very difficult to find a matrix A satisfying RIP deterministically. However, there are many types of matrices A that satisfy RIP with high probability (i.e. less than but very close to 1, e.g. $1-e^{-\rho}$ or $1-O(n^{-\rho})$, see [143, 145] for more details). For example, it is shown in [145] that when the $m \times n$ matrix A is subgaussian (e.g. Gaussian, Bernoulli etc.), and if

$$m \sim \kappa \log(n/\kappa),$$

then the RIP is satisfied with high probability. Another example is partial Fourier matrices. It is shown in [143] that when A is Fourier submatrix obtained by randomly chosen m rows from the original $n \times n$ Fourier matrix, and if

$$m \sim \kappa \log n,$$

then the RIP is satisfied with high probability.

Now, the CS problem then becomes one of solving (4.2) fast. Conventional linear programming solvers are not tailored for the large scale dense matrices A and the sparse solutions u that arise here. To overcome this, Bregman distance based methods were developed recently (see [137–140]), which are all very efficient in solving (4.2).

Bregman iteration applied to (4.2) involves solving the constrained optimization problem through solving a small number of unconstrained optimization problems:

$$\min_u \left\{ \mu \|u\|_1 + \frac{1}{2} \|Au - f\|_2^2 \right\} \tag{4.3}$$

for $\mu > 0$.

In [138], the authors used a method called the fast fixed point continuation solver (FPC) [147] which appears to be efficient. Other solvers of (4.3) could be used in this Bregman iterative regularization procedure.

In this chapter we will improve and analyze a linearized Bregman iterative regularization procedure, which, in its original incarnation [137, 138], involved only a two line code and simple operations and was already extremely fast and accurate.

In addition, we are interested in the denoising properties of Bregman iterative regularization, for signals, not images. The results for images involved the BV

norm, which we may discretize for $n \times n$ pixel images as:

$$TV(u) = \sum_{i,j=1}^{n-1} ((u_{i+1,j} - u_{ij})^2 + (u_{i,j+1} - u_{ij})^2)^{\frac{1}{2}}. \tag{4.4}$$

We usually regard the success of the ROF TV based model [148]

$$\min_u \left\{ TV(u) + \frac{\lambda}{2}\|f - u\|^2 \right\} \tag{4.5}$$

(we now drop the subscript 2 for the L_2 norm throughout this section) as due to the fact that images have edges and in fact are almost piecewise constant (with texture added). Therefore, it is not surprising that sparse signals could be denoised using (4.3). The ROF denoising model was greatly improved in [146] and [149] with the help of Bregman iterative regularization. We will do the same thing here using Bregman iteration with (4.3) to denoise sparse signals, with the added touch of undersampling the noisy signals.

This chapter is organized as follows: In section 4.2 we describe Bregman iterative algorithms, as well as the linearized version. We motivate these methods and describe previously obtained theoretical results. In section 4.3 and 4.4 we introduce an improvement to the linearized version, call "kicking" which greatly speeds up the method, especially for solutions u with a large dynamic range. In section 4.5, we shall discuss some TV-based models which can be converted into a generalized ℓ_1-minimization problem, and hence fast implementations are available through Bregman iterations. Finally in section 4.6 we present some numerical results, including sparse recovery for u having large dynamic range, and the recovery of signals in large amounts of noise.

4.2 Bregman and Linearized Bregman Iterative Algorithms

Discussions in this section are based on the following more general minimization problem, with $J(u)$ some convex functional,

$$\min_{u\in\mathbb{R}^n}\{J(u)|Au=f\}. \tag{4.6}$$

The Bregman distance [151], based on the convex function J, between points u and v, is defined by

$$D_J^p(u,v)=J(u)-J(v)-\langle p,u-v\rangle \tag{4.7}$$

where $p\in\partial J(v)$ is an element in the subgradient of J at the point v. In general $D_J^p(u,v)\neq D_J^p(v,u)$ and the triangle inequality is not satisfied, so $D_J^p(u,v)$ is not a distance in the usual sense. However it does measure the closeness between u and v in the sense that $D_J^p(u,v)\geq 0$ and $D_J^p(u,v)\geq D_J^p(w,v)$ for all points w on the line segment connecting u and v. Moreover, if J is convex, $D_J^p(u,v)\geq 0$, if J is strictly convex $D_J^p(u,v)>0$ for $u\neq v$ and if J is strongly convex, then there exists a constant $a>0$ such that

$$D_J^p(u,v)\geq a\|u-v\|^2.$$

To solve (4.6) Bregman iteration was proposed in [138] . Given $u^0=p^0=0$, we define:

$$\begin{aligned} u^{k+1}&=\arg\min_{u\in\mathbb{R}^n}\left\{J(u)-J(u^k)-\langle u-u^k,p^k\rangle+\frac{1}{2}\|Au-f\|^2\right\} \\ p^{k+1}&=p^k-A^T(Au^{k+1}-f). \end{aligned} \tag{4.8}$$

This can be written as

$$u^{k+1} = \arg\min_{u \in R^2} \left\{ D_J^{p^k}(u, u^k) + \frac{1}{2}\|Au - f\|^2 \right\}.$$

It was proven in [138] that if $J(u) \in C^2(\Omega)$ and is strictly convex in Ω, then $\|Au^k - f\|$ decays exponentially whenever $u^k \in \Omega$ for all k. Furthermore, when u^k converges, its limit is a solution of (4.6). It was also proven in [138] that when $J(u) = \|u\|_1$, i.e. for problem (4.2), or when J is a convex function satisfying some additional conditions, the iteration (4.8) leads to a solution of (4.6) in finitely many steps.

As shown in [138], see also [146, 149], the Bregman iteration (4.8) can be written as:

$$\begin{aligned} f^{k+1} &= f^k + f - Au^k \\ u^{k+1} &= \arg\min_{u \in \mathbb{R}^n} \left\{ J(u) + \frac{1}{2}\|Au - f^{k+1}\|^2 \right\} \end{aligned} \tag{4.9}$$

This was referred to as "adding back the residual" in [146] . Here $f^0 = 0, u^0 = 0$. Thus the Bregman iteration uses solutions of the unconstrained problem

$$\min_{u \in R} \left\{ J(u) + \frac{1}{2}\|Au - f\|^2 \right\} \tag{4.10}$$

as a solver in which the Bregman iteration applies this process iteratively.

Since there is generally no explicit expression for the solver of (4.8) or (4.9), we turn to iterative methods. The linearized Bregman iteration which we will

analyze, improve and use here is generated by

$$u^{k+1} = \arg\min_{u\in\mathbb{R}^n} \left\{ J(u) - J(u^k) - \langle u - u^k, p^k\rangle + \frac{1}{2\delta}\|u - (u^k - \delta A^T(Au^k - f))\|^2 \right\}$$
$$p^{k+1} = p^k - \frac{1}{\delta}(u^{k+1} - u^k) - A^T(Au^k - f). \tag{4.11}$$

In the special case considered here, where $J(u) = \mu\|u\|_1$, then we have the two line algorithm

$$v^{k+1} = v^k - A^T(Au^k - f) \tag{4.12}$$
$$u^{k+1} = \delta \cdot \text{shrink}(v^{k+1}, \mu) \tag{4.13}$$

where v^k is an auxiliary variable

$$v^k = p^k + \frac{1}{\delta}u^k \tag{4.14}$$

and

$$\text{shrink}(x,\mu) := \begin{cases} x-\mu, & \text{if } x > \mu \\ 0, & \text{if } -\mu \le x \le \mu \\ x+\mu, & \text{if } x < -\mu \end{cases}$$

is the soft thresholding algorithm [152] .

This linearized Bregman iterative algorithm was invented in [137] and then used and analyzed in [138–140]. In fact it comes from the inner-outer iteration for (4.8). In [138] it was shown that the linearized Bregman iteration (4.11) is just one step of the inner iteration for each outer iteration. Here we repeat the arguments also in [138], which begin by summing the second equation in (4.11)

arriving at (using the fact that $u^0 = p^0 = 0$):

$$p^k + \frac{1}{\delta}u^k + \sum_{j=0}^{k-1} A^T(Au^j - f) = p^k + \frac{1}{\delta}u^k - v^k = 0, \text{ for } k = 1, 2, \ldots.$$

This gives us (4.13), and allows us to rewrite its first equation as:

$$u^{k+1} = \arg\min_{u \in \mathbb{R}^n} \left\{ J(u) + \frac{1}{2\delta}\|u - \delta v^{k+1}\|^2 \right\} \tag{4.15}$$

i.e. we are adding back the "linearized noise", where v^{k+1} is defined in (4.12).

In [138] and [139] some interesting analysis was done for (4.11), (and some for (4.15)). This was done first for $J(u)$ continuously differentiable in (4.11) and the gradient $\partial J(u)$ satisfying

$$\|\partial J(u) - \partial J(v)\|^2 \leq \beta\langle \partial J(u) - \partial J(v), u - v\rangle, \tag{4.16}$$

$\forall u, v \in R^n,\ \beta > 0.$

In [139] it was shown that, if (4.16) is true, then both of the sequences $(u^k)_{k \in N}$ and $(p^k)_{k \in N}$ defined by (4.11) converge for $0 < \delta < \frac{2}{\|AA^T\|}$.

In [140] the authors recently give a theoretical analysis, showing that the iteration in (4.12) and (4.13) converges to the unique solution of

$$\min_{u \in \mathbb{R}^n} \left\{ \mu\|u\|_1 + \frac{1}{2\delta}\|u\|^2 : Au = f \right\} \tag{4.17}$$

They also show the interesting result: let S be the set of all solutions of the Basis Pursuit problem (4.2) and let

$$u_1 = \arg\min_{u \in S} \|u\|^2 \tag{4.18}$$

which is unique. Denote the solution of (4.17) as u_μ^*. Then

$$\lim_{\mu\to\infty} \|u_\mu^* - u_1\| = 0. \tag{4.19}$$

In passing they show that

$$\|u_\mu^*\| \leq \|u_1\| \text{ for all } \mu > 0 \tag{4.20}$$

which we will use below.

Another theoretical analysis on Linearized Bregman algorithm is given by Yin in [153], where he shows that Linearized Bregman iteration is equivalent to gradient descent applied to the dual of the problem (4.17) and uses this fact to obtain an elegant convergence proof. He also proved that $u_\mu^* = u^*$ for $\mu > \mu_0$ and some $\mu_0 > 0$, where u^* is the optimizer of (4.2).

This summarizes the relevant convergence analysis for our Bregman and linearized Bregman models.

Next we recall some results from [146] regarding noise and Bregman iteration.

For any sequence $\{u^k\}, \{p^k\}$ satisfying (4.8) for J continuous and convex, we have, for any $\tilde{\mu}$

$$D_J^{p^k}(\tilde{u}, u^k) - D_J p^{k-1}(\tilde{u}, u^{k-1}) \leq \langle A\tilde{u} - f, Au^{k-1} - f\rangle - \|Au^{k-1} - f\|^2. \tag{4.21}$$

Besides implying that the Bregman distance between u^k and any element $\tilde{u}$ satisfying $A\tilde{u} = f$ is monotonically decreasing, it also implies that, if $\tilde{u}$ is the "noise free" approximation to the solution of (4.6), the Bregman distance between

u^k and $\tilde{u}$ diminishes as long as

$$\|Au^k - f\| > \|A\tilde{u} - f\| = \sigma, \text{ where } \sigma \text{ is some measure of the noise} \tag{4.22}$$

i.e., until we get too close to the noisy signal in the sense of (4.22). Note, in [146] we took A to be the identity, but these more general results are also proven there. This gives us a stopping criterion for our denoising algorithm.

In [146] we obtained a result for linearized Bregman iteration, following [154], which states that the Bregman distance between $\tilde{u}$ and u^k diminish as long as

$$\|A\tilde{u} - f\| < (1 - 2\delta\|AA^T\|)\ \|Au^k - f\| \tag{4.23}$$

so we need $0 < 2\delta\|AA^T\| < 1$.

In practice, we will use (4.22) as our stopping criterion.

4.3 Convergence Analysis of Linearized Bregman Iterations

We begin with the following simple results for the linearized Bregman iteration or the equivalent algorithm (4.11). Throughout the rest of this section, we shall focus on the case with $J(u) = \|u\|_1$, i.e. the problem (4.2).

Theorem 4.3.1. *If $u^k \to u^\infty$, then $Au^\infty = f$.*

Proof. Assume $Au^\infty \neq f$. Then $A^T(Au^\infty - f) \neq 0$ since A^T has full rank. This means that for some i, $(A^T(Au^k - f))_i$ converges to a nonzero value, which means that $v_i^{k+1} - v_i^k$ does as well. On the other hand $\{v^k\} = \{u^k/\delta + p^k\}$ is bounded since $\{u^k\}$ converges and $p^k \in [-\mu, \mu]$. Therefore $\{v_i^k\}$ is bounded, while $v_i^{k+1} - v_i^k$

converges to a nonzero limit, which is impossible. □

Theorem 4.3.2. *If $u^k \to u^\infty$ and $v^k \to v^\infty$, then u^∞ minimizes $\{J(u)+\frac{1}{2\delta}\|u\|^2 : Au = f\}$.*

Proof. Let $\tilde{J}(u) = J(u) + \frac{1}{2\delta}\|u\|^2$. then

$$\partial\tilde{J}(u) = \partial J(u) + \frac{1}{\delta}u.$$

Since $\partial J(u^k) = p^k = v^k - u^k/\delta$, we have $\partial\tilde{J}(u^k) = v^k$. Using the non-negativity of the Bregman distance we obtain

$$\begin{aligned}\tilde{J}(u^k) &\le \tilde{J}(u_{\text{opt}}) - \langle u_{\text{opt}} - u^k, \partial\tilde{J}(u^k)\rangle \\ &= \tilde{J}(u_{\text{opt}}) - \langle u_{\text{opt}} - u^k, v^k\rangle\end{aligned}$$

where u_{opt} minimizes (4.6) with J replaced by $\tilde{J}$, which is strictly convex.

Let $k \to \infty$, we have

$$\tilde{J}(u^\infty) \le \tilde{J}(u_{\text{opt}}) - \langle u_{\text{opt}} - u^\infty, v^\infty\rangle$$

Since $v^k = A^T\sum_{j=0}^{k-1} A^T(f - Au^j)$, we have $v^\infty \in \text{range}(A^T)$. Since $Au_{\text{opt}} = Au^\infty = f$, we have $\langle u_{\text{opt}} - u^\infty, v^\infty\rangle = 0$, which implies $\tilde{J}(u^\infty) \le \tilde{J}(u_{\text{opt}})$. □

Equation (4.17) (from a result in [139]) implies that u^∞ will approach a solution to (4.6), as μ approaches ∞.

The linearized Bregman iteration has the following monotonicity property:

Theorem 4.3.3. *If $u^{k+1} \neq u^k$ and $0 < \delta < 2/\|AA^T\|$, then*

$$\|Au^{k+1} - f\| < \|Au^k - f\|.$$

Proof. Let

$$u^{k+1} - u^k = \Delta u^k,\ v^{k+1} - v^k = \Delta v^k.$$

Then the shrinkage operation is such that

$$\Delta u_i^k = \delta q_i^k \Delta v_i^k \tag{4.24}$$

with

$$q_i^k \begin{cases} = 1 & \text{if } u_i^{k+1} u_i^k > 0 \\ = 0 & \text{if } u_i^{k+1} = u_i^k = 0 \\ \in (0,1] & \text{otherwise} \end{cases}$$

Let $Q^k = \text{Diag}\,(q_i^k)$. Then (4.24) can be written as

$$\Delta u^k = \delta Q^k \Delta v^k = \delta Q^k A^T (f - Au^k) \tag{4.25}$$

which implies

$$Au^{k+1} - f = (I - \delta A Q^k A^T)(Au^k - f). \tag{4.26}$$

From (4.24), Q^k is diagonal with $0 \preceq Q^k \preceq I$, so $0 \preceq AQ^kA^T \preceq AA^T$. If we choose $\delta > 0$ such that $\delta AA^T \prec 2I$, then $0 \preceq \delta AQ^kA^T \prec 2I$ or $-I \prec I - \delta AQ^kA^T \preceq I$ which implies that $\|Au^k - f\|$ is not increasing. To get strict decay, we need only show that $AQ^kA^T(Au^k - f) = 0$ is impossible if $u^{k+1} \neq u^k$. Suppose $AQ^kA^T(Au^k - f) = 0$ holds, then from (4.25) we have:

$$\langle \Delta u^k, \Delta v^k \rangle = \delta \langle A^T(f - Au^k), Q^k A^T (f - Au^k) \rangle = 0.$$

By (4.24), this only happens if $u_i^{k+1} = u_i^k$ for all i, which is a contradiction. □

We are still faced with estimating how fast the residual decays. It turns out that if consecutive elements of u do not change sign, then $\|Au - f\|$ decays exponentially. By 'exponential' we mean that the ratio of the residuals of two consecutive iteration converges to a constant, this type of convergence is sometimes called linear convergence. Here we define

$$S_u = \{x \in R^n : \text{sign}(x_i) = \text{sign}(u_i), \forall i\} \tag{4.27}$$

(where $\text{sign}(0) = 0$ and $\text{sign}(a) = a/|a|$ for $a \neq 0$). Then we have the following:

Theorem 4.3.4. *If $u^k \in S \equiv S_{u_k}$ for $k \in (T_1, T_2)$, then u^k converges to u^*, where $u^* \in \arg\min\{\|Au - f\|^2 : u \in S\}$ and $\|Au^k - f\|^2$ decays to $\|Au^* - f\|^2$ exponentially.*

Proof. . Since $u^k \in S$ for $k \in [T_1, T_2]$, we can define $Q \equiv Q^k$ for $T_1 \leq k \leq T_2 - 1$. From (4.24) we see that Q^k is a diagonal matrix consisting of zeros or ones, so $Q = Q^T Q$. Moreover, it is easy to see that $S = \{x | Qx = x\}$.

Following the argument in Theorem 4.3.3 we have:

$$u^{k+1} - u^k = \Delta u^k = \delta Q \Delta v^k = \delta Q A^T (f - Au^k) \tag{4.28}$$

$$Au^{k+1} - f = [I - \delta A Q A^T](Au^k - f) \tag{4.29}$$

and

$$-I \prec I - \delta A Q A^T \preceq I.$$

Let $R^n = V_0 \oplus V_1$, where V_0 is the null space of AQA^T and V_1 is spanned by the eigenvectors corresponding to the nonzero eigenvalues of AQA^T. Let

$Au^k - f = w^{k,0} + w^{k,1}$, where $w^{k,j} \in V_j$ for $j = 0, 1$. From (4.29) we have

$$\begin{aligned} w^{k+1,0} &= w^{k,0} \\ w^{k+1,1} &= [I - \delta AQA^T]w^{k,1} \end{aligned}$$

for $T_1 \leq k \leq T_2 - 1$. Since $w^{k,1}$ is not in the null space of AQA^T, then (4.28) and (4.29) imply that $\|w^{k,1}\|$ decays exponentially. Let $w^0 = w^{k,0}$, then $AQA^T w^0 = 0$ $AQQA^T w^0 \Rightarrow QA^T w^0 = 0$. Therefore, from (4.28) we have

$$\Delta u^k = \delta Q^T A^T (f - Au^k) = \delta QA^T (w^0 + w^{k,1}) = \delta QA^T w^{k,1}.$$

Thus $\|\Delta u^k\|$ decays exponentially. This means $\{u^k\}$ forms a Cauchy sequence in S, so it has a limit $u^* \in S$. Moreover

$$Au^* - f = \lim_k (Au^k - f) = \lim_k w^{k,0} + \lim_k w^{k,1} = w^0.$$

Since V_0 and V_1 are orthogonal:

$$\|Au^k - f\|^2 = \|w^{k,0}\|^2 + \|w^{k,1}\|^2 = \|Au^* - f\|^2 + \|w^{k,1}\|^2,$$

so $\|Au^k - f\|^2 - \|Au^* - f\|^2$ decays exponentially. The only thing left to show is that

$$u^* = \arg\min(\|Au - f\|^2 : u \in S) = \arg\min\{\|Au - f\|^2 : Qu = u\}.$$

This is equivalent to way that $A^T(Au^* - f)$ is orthogonal with the hyperspace $\{u : Qu = u\}$. It's easy to see that since Q is a projection operator, a vector v is orthogonal with $\{u : Qu = u\}$ if and only if $Qv = 0$, thus we need to show

$QA^T(Au^* - f) = 0$. This is obvious because we have shown that $Au^* - f = w^0$ and $QA^T w^0 = 0$. So we find that u^* is the desired minimizer. □

Therefore, instead of decaying exponentially with a global rate, the residual of the linearized Bregman iteration decays in a rather sophisticated manner. From the definition of the shrinkage function we can see that the sign of an element of u will change if and only if the corresponding element of v crosses the boundary of the interval $[-\mu, \mu]$. If μ is relatively large compared with the size of Δv (which is usually the case when applying the algorithm to a compressed sensing problem), then at most iterations the signs of the elements of u will stay unchanged, i.e. u will stay in the subspace S_u defined in (4.27) for a long while. This theorem tells us that under this scenario u will quickly converge to the point u^* that minimizes $\|Au - f\|$ inside S_u, and the difference between $\|Au - f\|$ and $\|Au^* - f\|$ decays exponentially. After u converges to u^*, u will stay there until the sign of some element of u changes. Usually this means that a new nonzero element of u comes up. After that, u will enter a different subspace S and a new converging procedure begins.

The phenomenon described above can be observed clearly in Fig 4.1. The final solution of u contains five non-zero spikes. Each time a new spike appears, it converges rapidly to the position that minimizes $\|Au - f\|$ in the subspace S_u. After that there is a long stagnation, which means u is just waiting there until the accumulating v brings out a new non-zero element of u. The larger μ is, the longer the stagnation takes. Although the convergence of the residual during each phase is fast, the total speed of the convergence suffers much from the stagnation. The solution of this problem will be described in the next section.

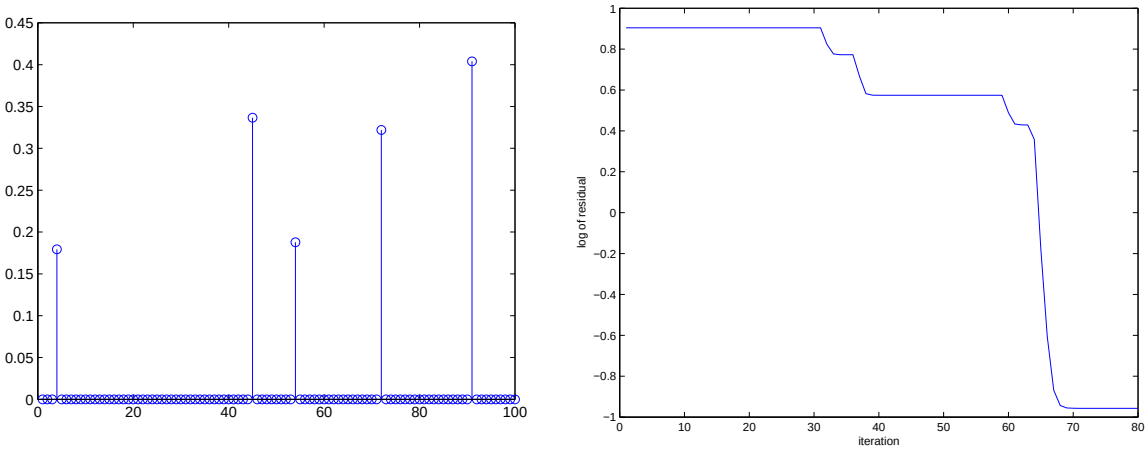


Figure 4.1: The left figure presents a simple signal with 5 non-zero spikes. The right figure shows how the linearized Bregman iteration converges.

4.4 Fast Implementation

The iterative formula in Algorithm 6 below gives us the basic linearized Bregman algorithm designed to solve (4.6), in particular (4.2).

Algorithm 6 Bregman Iterative Regularization

Initialize: $u = 0$, $v = 0$.
while "$\|f - Au\|$ not converge" **do**
$v^{k+1} = v^k + A^\top(f - Au^k)$
$u^{k+1} = \delta \cdot \text{shrink}(v^{k+1}, \mu)$
end while

This is an extremely concise algorithm, simple to program, involve only matrix multiplication and shrinkage. When A consists of rows of a matrix of a fast transform like FFT which is a common case for compressed sensing, it is even faster because matrix multiplication can be implemented efficiently using the existing fast code of the transform. Also, storage becomes a less serious issue.

We now consider how we can accelerate the algorithm under the problem of stagnation described in the previous section. From that discussion, during a

stagnation u converges to a limit u^* so we will have $u^{k+1} \approx u^{k+2} \approx \cdots \approx u^{k+m} \approx u^*$ for some m. Therefore the increment of v in each step, $A^\top(f - Au)$, is fixed. This implies that during the stagnation u and v can be calculated explicitly as following

$$\begin{cases} u^{k+j} \equiv u^{k+1} \\ v^{k+j} = v^k + j \cdot A^\top(f - Au^{k+1}) \end{cases} \quad j = 1, \cdots, m \tag{4.30}$$

If we denote the set of indices of the zero elements of u^* as I_0 and let $I_1 = \overline{I_0}$ be the support of u^*, then v_i^k will keep changing only for $i \in I_0$ and the iteration can be formulated entry-wise as:

$$\begin{cases} u_i^{k+j} \equiv u_i^{k+1} & \forall i \\ v_i^{k+j} = v_i^k + j \cdot (A^\top(f - Au^{k+1}))_i & i \in I_0 \\ v_i^{k+j} \equiv v_i^{k+1} & i \in I_1 \end{cases} \tag{4.31}$$

for $j = 1, \cdots, m$. The stagnation will end when u begins to change again. This happens if and only if some element of v in I_0 (which keeps changing during the stagnation) crosses the boundary of the interval $[-\mu, \mu]$. When $i \in I_0$, $v_i^k \in [-\mu, \mu]$, so we can estimate the number of the steps needed for v_i^k to cross the boundary $\forall i \in I_0$ from (4.31), which is

$$s_i = \left\lceil \frac{\mu \cdot \text{sign}((A^\top(f - Au^{k+1}))_i) - v_i^{k+1}}{(A^\top(f - Au^{k+1}))_i} \right\rceil \ \forall i \in I_0 \tag{4.32}$$

and

$$s = \min_{i \in I_0} \{s_i\} \tag{4.33}$$

is the number of steps needed. Therefore, s is nothing but the length of the

stagnation. Using (4.30), we can predict the end status of the stagnation by

$$\begin{cases} u^{k+s} \equiv u^{k+1} \\ v^{k+s} = v^k + s \cdot A^\top (f - Au^{k+1}) \end{cases} \quad j = 1, \cdots, m \tag{4.34}$$

Therefore, we can *kick* u to the critical point of the stagnation when we detect that u has been staying unchanged for a while. Specifically, we have the following algorithm: Algorithm 7.

Algorithm 7 Linearized Bregman Iteration with Kicking

```
Initialize: u = 0, v = 0.
while "‖f − Au‖ not converge" do
  if "u^{k−1} ≈ u^k" then
    calculate s from (4.32) and (4.33)
    v_i^{k+1} = v_i^k + s · (A^⊤(f − Au^k))_i, ∀i ∈ I_0
    v_i^{k+1} = v_i^k, ∀i ∈ I_1
  else
    v^{k+1} = v^k + A^⊤(f − Au^k)
  end if
  u^{k+1} = δ · shrink(v^{k+1}, μ)
end while
```

Indeed, this kicking procedure is similar to line search commonly used in optimization problems and modifies the initial algorithm in no way but just accelerates the speed. More precisely, note that the output sequence $\{u^k, v^k\}$ is a subsequence of the original one, so all the previous theoretical conclusions on convergence still hold here.

An example of the algorithm is shown in Fig 4.2. It is clear that all the stagnation in the original convergence collapses to single steps. The total amount of computation is reduced dramatically.

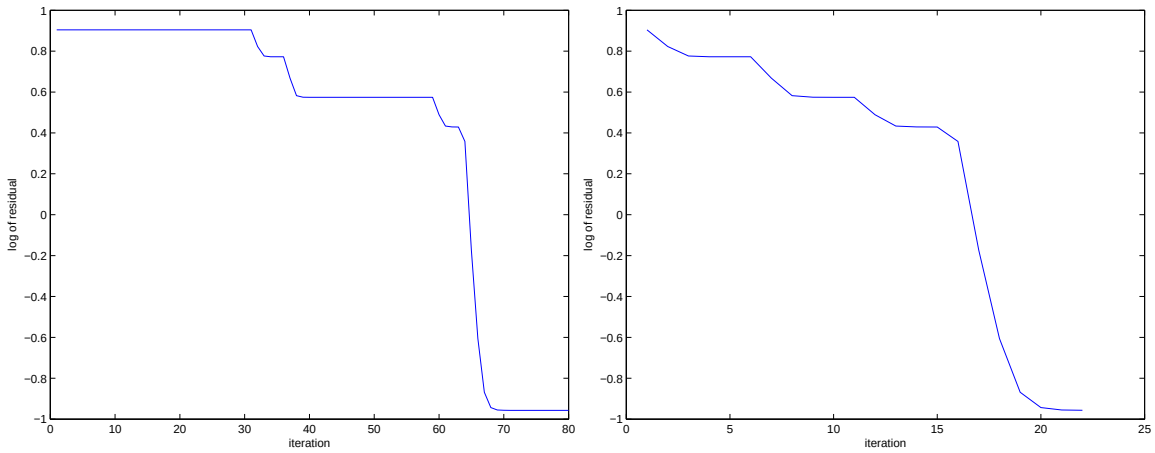


Figure 4.2: The left figure presents the convergence curve of the original linearized Bregman iteration using the same signal as Fig 4.1. The right figure shows the convergence curve of the linearized Bregman iteration with the kicking modification.

4.5 Fast Solver for Total Variation (TV) Based Models via Split Bregman Iterations

Total variation based model was first used in the context of image processing by Rudin, Osher and Fatemi [148]. After that, numerous TV based models were introduced for various image processing problem, e.g. denoising, deblurring, segmentation, inpainting, etc. Interested readers can consult the textbooks [173–175] and the references there for more details.

The classical and most straightforward way of solving a TV based model is by gradient descent method, which is very reliable in practice. However, gradient descent method is very slow computationally and it usually requires huge number of iterations before convergence. The major difficulty in designing a fast solver for TV based models is due to the non-differentiability of the TV term. In recent years, many numerical methods [157–167, 170] are developed to improve the speed of solving TV based models. One of the most efficient and general method is the

split Bregman iterations introduced by Goldstein and Osher [170]. We will recall their algorithm in this section and use it on medical image analysis in the next two chapters.

Consider the following general TV based variational model

$$\text{minimize} \quad E(u) := \int_\Omega g(x)|\nabla u(x)|dx + c_1\|u - f\|_1 + c_2\|Au - f\|^2, \qquad (4.35)$$

where Ω is the computation domain e.g. image domain, c_i, $i = 1, 2$ are constant parameters, $g(x)$ is some weight function, and f is the observed image. When $g(x) \equiv 1$, $c_1 = 0$ and $A = I$, (4.35) becomes the standard ROF model [148]; when $g(x) \equiv 1$, $c_1 = 0$ and A is some convolution operator, (4.35) becomes a model used for image deblurring (see e.g. [146]); when $g(x) \equiv 1$ and $c_2 = 0$, (4.35) becomes the TV-L_1 model [168, 177]; when $c_2 = 0$, then (4.35) becomes the snake model considered in [169].

Now let $d_1 = \nabla u$, $d_2 = u - f$ and $d := (d_1^T, d_2)^T$. Define

$$|d(x)|_* := g(x)\|d_1\| + c_1|d_2| \quad \text{and} \quad \Psi u := (\nabla u^T, u)^T.$$

Then the optimization (4.35) can be transformed to the following equivalent form

$$\begin{aligned} \text{Minimize}_{u,d} \quad & \int |d|_* + c_2\|Au - f\|^2 \\ \text{s.t.} \quad & d = \Psi u - (\mathbf{0}, f)^T =: Fu. \end{aligned} \qquad (4.36)$$

Since the above optimization problem is a special case of the general model (4.6),

we can use Bregman iterations, as described in section 1.2, to solve it:

$$\begin{aligned}(u^{k+1}, d^{k+1}) &= \arg\min_{u,d} \int |d|_* + c_2\|Au - f\|^2 + \frac{\mu}{2}\|d - Fu - b^k\|_2^2 \\ b^{k+1} &= b^k + \left(Fu^{k+1} - d^{k+1}\right).\end{aligned} \tag{4.37}$$

As one can see, the first step in the above scheme is the crucial step, where we need to solve the minimization problem for both variables u and d. It is suggested by [170] that one can minimize it alternatively in u and d, i.e. minimize the functional by fixing d and u alternatively. Here we shall skip the details and leave them to [170] (not exactly in the same format as (4.37), but the idea applies). In Chapter 5, we will derive the detailed scheme from (4.37) for the snake model, i.e. the case $c_2 = 0$.

4.6 Numerical Results

In this section, we demonstrate the effectiveness of the Algorithm 7 in solving the standard ℓ_1-minimization (4.2) and some related problems.

4.6.1 Efficiency

Consider the constrained minimization problem

$$\min \|u\|_1 \quad \text{s.t. } Au = f,$$

where the constraints $Au = f$ are under-determined linear equations with A an $m \times n$ matrix, and f generated from a sparse signal $\bar{u}$ that has a number of nonzeros $\kappa < m$.

Our numerical experiments use two types of A matrices: Gaussian matrices

whose elements were generated from i.i.d. normal distributions $\mathcal{N}(0,1)$ (**randn(m,n)** in MATLAB), and partial discrete cosine transform (DCT) matrices whose k rows were chosen randomly from the $n \times n$ DCT matrix. These matrices are known to be efficient for compressed sensing. The number of rows m is chosen as $m \sim \kappa \log(n/\kappa)$ for Gaussian matrices and $m \sim \kappa \log n$ for DCT matrices (following [143]).

The tested *original sparse signals* $\bar{u}$ had numbers of nonzeros equal to $0.05n$ and $0.02n$ rounded to the nearest integers in two sets of experiments, which were obtained by **round(0.05*n)** and **round(0.02*n)** in MATLAB, respectively. Given a sparsity $\|\bar{u}\|_0$, i.e., the number of nonzeros, an *original sparse signal* $\bar{u} \in \mathbb{R}^n$ was generated by randomly selecting the locations of $\|\bar{u}\|_0$ nonzeros, and sampling each of these nonzero elements from $\mathcal{U}(-1,1)$ (**2*(rand-0.5)** in MATLAB). Then, f was computed as $A\bar{u}$. When $\|\bar{u}\|_0$ is small enough, we expect the ℓ_1-minimization problem, which we solved using our fast algorithm, to yield a solution $u^* = \bar{u}$ from the inputs A and f.

Note that partial DCT matrices are implicitly stored fast transforms for which matrix-vector multiplications in the forms of Ax and $A^\top x$ were computed by the MATLAB commands **dct(x)** and **idct(x)**, respectively. Therefore, we were able to test on partial DCT matrices of much larger sizes than Gaussian matrices. The sizes m-by-n of these matrices are given in the first two columns of Table 4.1.

Our code was written in MATLAB and was run on a Windows PC with a Intel(R) Core(TM) 2 Duo 2.0GHz CPU and 2GB memory. The MATLAB version is 7.4.

The set of computational results given in Table 4.1 was obtained by using the

stopping criterion

$$\frac{\|Au^k - f\|}{\|f\|} < 10^{-5}, \tag{4.38}$$

which was sufficient to give a small error $\|u^k - \bar{u}\|/\|\bar{u}\|$. Throughout our experiments in Table 4.1, we used $\mu = 1$ to ensure the correctness of the results.

Table 4.1: Experiment results using 10 random instances for each configuration of $(m, n, \|\bar{u}\|_0)$, with nonzero elements of $\bar{u}$ come from $\mathcal{U}(-1, 1)$.

Results of linearized Bregman-L_1 with kicking										
Stopping tolerance.		$\|Au^k - f\|/\|f\| < 10^{-5}$								
		Gaussian matrices								
		stopping itr. k			relative error $\|u^k - \bar{u}\|/\|\bar{u}\|$			time (sec.)		
		mean	std.	max	mean	std.	max	mean	std.	max
n	m	$\|\bar{u}\|_0 = 0.05n$								
1000	300	422	67	546	2.0e-05	4.3e-06	2.7e-05	0.42	0.06	0.51
2000	600	525	57	612	1.8e-05	1.9e-06	2.1e-05	4.02	0.45	4.72
4000	1200	847	91	1058	1.7e-05	1.7e-06	1.9e-05	25.7	2.87	32.1
n	m	$\|\bar{u}\|_0 = 0.02n$								
1000	156	452	98	607	2.3e-05	2.6e-06	2.6e-05	0.24	0.06	0.33
2000	312	377	91	602	2.0e-05	4.0e-06	2.9e-05	1.45	0.38	2.37
4000	468	426	30	477	1.6e-05	2.1e-06	2.0e-05	6.96	0.51	7.94
		Partial DCT matrices								
n	m	$\|\bar{u}\|_0 = 0.05n$								
4000	2000	71	6.6	82	9.1e-06	2.5e-06	1.2e-05	0.43	0.06	0.56
20000	10000	158	14.5	186	6.2e-06	2.1e-06	1.1e-05	3.95	0.36	4.73
50000	25000	276	14	296	6.8e-06	2.6e-06	1.0e-05	17.6	0.99	19.2
n	m	$\|\bar{u}\|_0 = 0.02n$								
4000	1327	52	7.0	64	8.6e-06	1.3e-06	1.1e-05	0.27	0.04	0.35
20000	7923	91	10.3	115	7.2e-06	2.2e-06	1.1e-05	2.36	0.30	3.02
50000	21640	140	9.7	153	5.9e-06	2.4e-06	1.1e-05	8.53	0.66	9.42

4.6.2 Robustness to Noise

In real applications, the measurement f we obtain is usually contaminated by noise. The measurement we have is:

$$\tilde{f} = f + n = A\bar{u} + n, \quad n \in \mathcal{N}(0, \sigma).$$

To characterize the noise level, we shall use SNR (signal to noise ratio) instead of σ itself. The SNR is defined as follows

$$SNR(u) := 20 \log_{10}(\frac{\|\bar{u}\|}{\|n\|}).$$

In this section we test our algorithm on recovering the true signal $\bar{u}$ from A and the noisy measurement $\tilde{f}$. As in the last section, the nonzero entries of $\bar{u}$ are generated from $\mathcal{U}(-1, 1)$, and A is either a Gaussian random matrix or a partial DCT matrix. Our stopping criteria is given by

$$\text{std}\big(Au^k - \tilde{f}\big) < \sigma, \quad \text{and} \quad \text{Iter.} < 1000,$$

i.e. we stop whenever the standard deviation of residual $Au^k - \tilde{f}$ is less than σ or the number of iterations exceeds 1000. Table 4.2 shows numerical results for different noise level, size of A and sparsity. We also show one typical result for a partial DCT matrix with size $n = 4000$ and $\|\bar{u}\|_0 = 0.02n = 80$ in Figure 4.3.

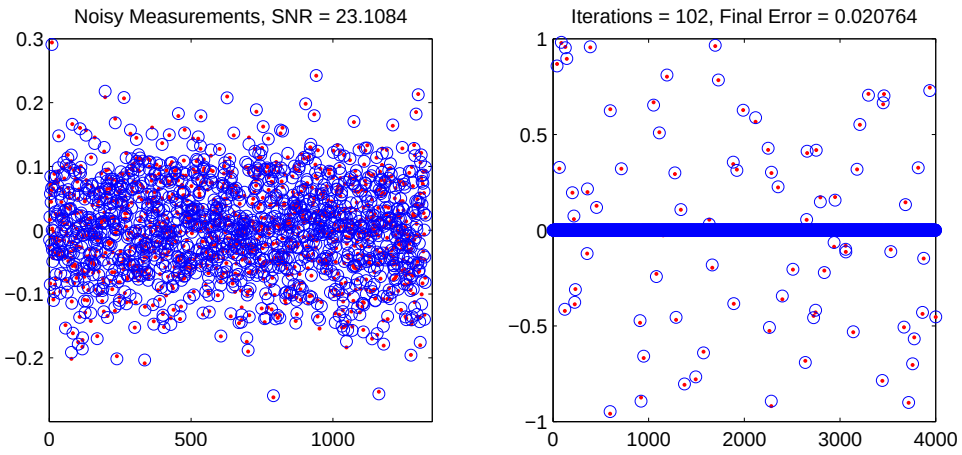


Figure 4.3: The left figure presents the clean (red dots) and noisy (blue circles) measurements, with SNR=23.1084; the right figure shows the reconstructed signal (blue circles) v.s. original signal (red dots), where the relative error=0.020764, and number of iterations is 102.

Table 4.2: Experiment results using 10 random instances for each configuration of $(m, n, \|\bar{u}\|_0)$.

Results of linearized Bregman-L_1 with kicking										
Stopping criteria.		std$(Au^k - f) < \sigma$.								
		Gaussian matrices								
		stopping itr. k			relative error $\|u^k - \bar{u}\|/\|\bar{u}\|$			time (sec.)		
		mean	std.	max	mean	std.	max	mean	std.	max
Avg. SNR	(n,m)	$\|\bar{u}\|_0 = 0.05n$								
26.12	(1000,300)	420	95	604	0.0608	0.0138	0.0912	0.33	0.09	0.53
25.44	(2000,600)	206	32	253	0.0636	0.0128	0.0896	1.49	0.22	1.79
26.02	(4000,1200)	114	11	132	0.0622	0.0079	0.0738	3.32	0.31	3.81
Avg. SNR	(n,m)	$\|\bar{u}\|_0 = 0.02n$								
27.48	(1000,156)	890	369	1612	0.0456	0.0085	0.0599	0.42	0.17	0.73
25.06	(2000,312)	404	64	510	0.0638	0.0133	0.0843	1.37	0.23	1.74
26.04	(4000,468)	216	35	267	0.0557	0.0068	0.0639	3.29	0.55	4.13
		Partial DCT matrices								
Avg. SNR	(n,m)	$\|\bar{u}\|_0 = 0.05n$								
23.97	(4000, 2000)	151	9.2	170	0.0300	0.0028	0.0332	0.94	0.07	1.03
24.00	(20000,10000)	250	14	270	0.0300	0.0010	0.0318	7.88	0.62	8.86
24.09	(50000,25000)	274	9.9	295	0.0304	0.0082	0.0315	20.4	0.74	20.1
Avg. SNR	(n,m)	$\|\bar{u}\|_0 = 0.02n$								
24.29	(4000,1327)	130	11	157	0.0223	0.0023	0.0253	0.79	0.08	1.00
24.37	(20000,7923)	223	14	257	0.0204	0.0025	0.0242	6.89	0.53	8.15
24.16	(50000,21640)	283	19	311	0.0193	0.0012	0.0207	21.5	1.68	24.1

4.6.3 Recovery of Signal with High Dynamical Range

In this section, we test our algorithm on signals with high dynamical ranges. Precisely speaking, let MAX $= \max\{|\bar{u}_i| : i = 1, \ldots, n\}$ and MIN $= \min\{|u_i| : u_i \neq 0, i = 1, \ldots, n\}$. The signals we shall consider here satisfy $\frac{\text{MAX}}{\text{MIN}} \approx 10^{10}$. Our $\bar{u}$ is generated by multiplying a random number in $[0, 1]$ with another one randomly picked from $\{1, 10, \ldots, 10^{10}\}$. Here we adopt the stopping criteria

$$\frac{\|Au^k - f\|}{\|f\|} < 10^{-11}$$

for the case without noise (Figure 4.4) and the same stopping criteria as in the previous section for the noisy cases (Figures 4.5-4.7). In the experiments, we take the dimension $n = 4000$, the number of nonzeros of $\bar{u}$ to be $0.02n$, and $\mu = 10^{10}$. Here μ is chosen to be much larger than before, because the dynamical range of $\bar{u}$ is large. Figure 4.4 shows results for the noise free case, where the algorithm converges to a 10^{-11} residual in less than 300 iterations. Figures 4.5-4.7 show the cases with noise (the noise is added the same way as in previous section). As one can see, if the measurements are contaminated with less noise, signals with smaller magnitudes will be recovered well. For example in Figure 4.5, the SNR$\approx$ 118, and the entries of magnitudes 10^4 are well recovered; in Figure 4.6, the SNR$\approx$ 97, and the entries of magnitudes 10^5 are well recovered; and in Figure 4.7, the SNR$\approx$ 49, and the entries of magnitudes 10^7 are well recovered.

4.6.4 Recovery of Sinusoidal Waves in Huge Noise

In this section we consider

$$\bar{u}(t) = a\sin(\alpha t) + b\cos(\beta t),$$

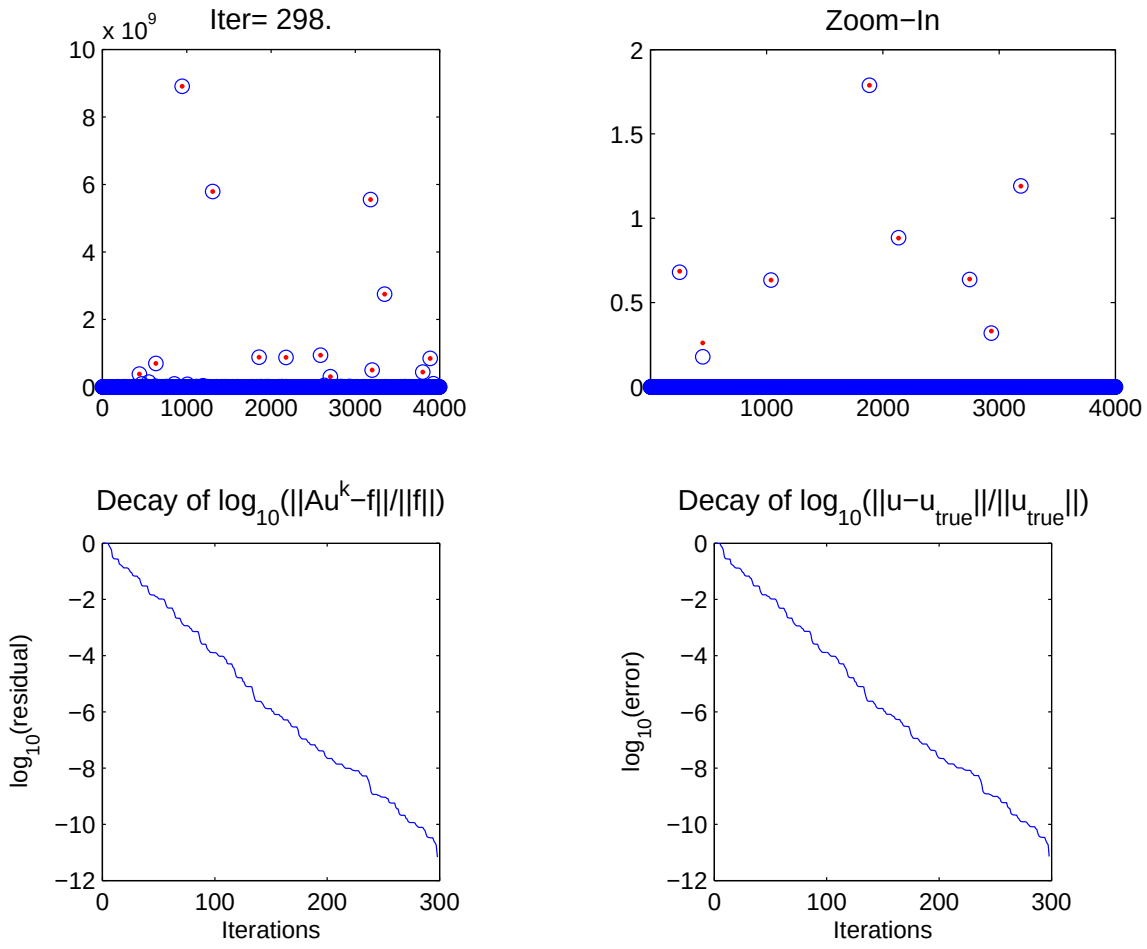


Figure 4.4: Upper left, true signal (red dots) v.s. recovered signal (blue circle); upper right, one zoom-in to the lower magnitudes; lower left, decay of residual $\log_{10} \frac{\|Au^k - f\|}{\|f\|}$; lower right, decay of error to true solution $\log_{10} \frac{\|u^k - \bar{u}\|}{\|\bar{u}\|}$.

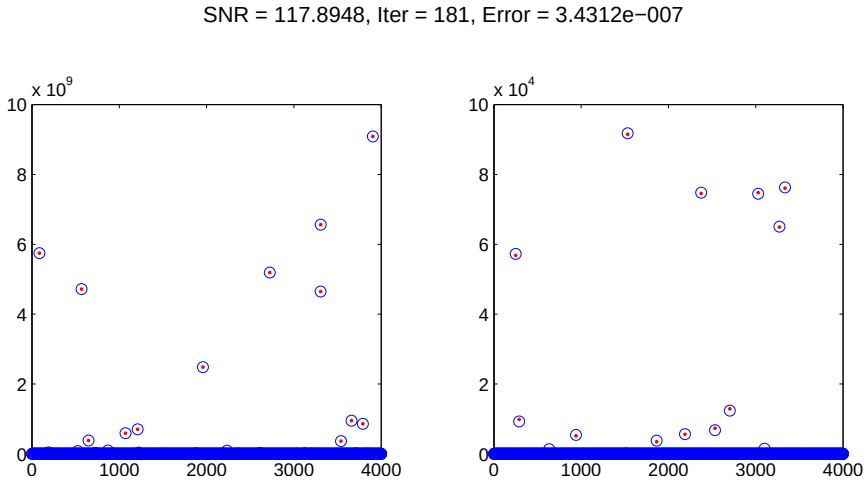


Figure 4.5: Noisy case. Left figure, true signal (red dots) v.s. recovered signal (blue circle); right figure, one zoom-in to the magnitude$\approx 10^5$. The error is measured by $\frac{\|u^k-\bar{u}\|}{\|\bar{u}\|}$.

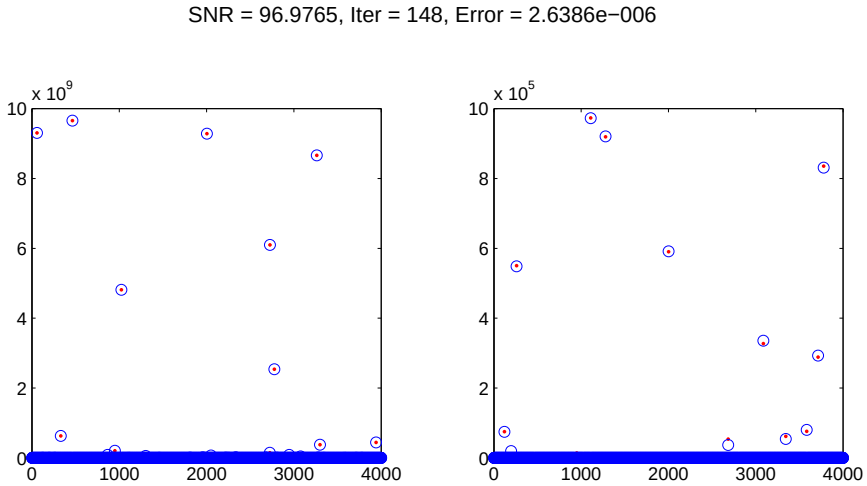


Figure 4.6: Noisy case. Left figure, true signal (red dots) v.s. recovered signal (blue circle); right figure, one zoom-in to the magnitude$\approx 10^6$. The error is measured by $\frac{\|u^k-\bar{u}\|}{\|\bar{u}\|}$.

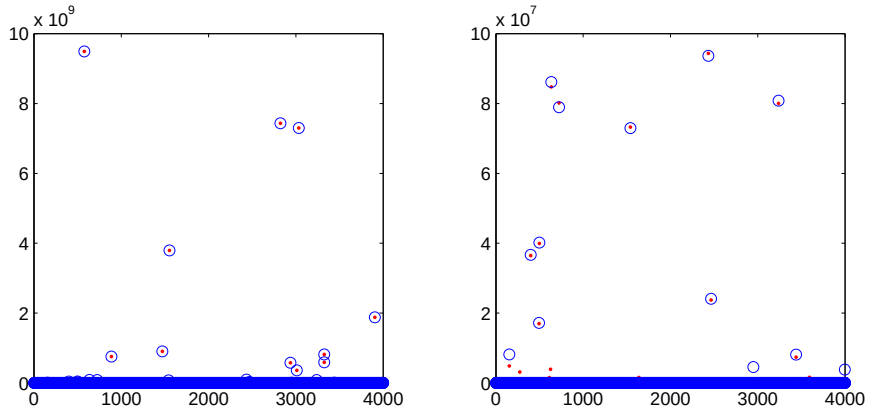


Figure 4.7: Noisy case. Left figure, true signal (red dots) v.s. recovered signal (blue circle); right figure, one zoom-in to the magnitude$\approx 10^8$. The error is measured by $\frac{\|u^k-\bar{u}\|}{\|\bar{u}\|}$.

where a, b, α and β are unknown. The observed signal $\tilde{u}$ is noisy and has the form $\tilde{u} = \bar{u} + n$ with $n \sim \mathcal{N}(0, \sigma)$. In practice, the noise in $\tilde{u}$ could be huge, i.e. possibly have a negative SNR, and we may only be able to observe partial information of $\tilde{u}$, i.e. only a subset of values of $\tilde{u}$ is known. Notice that the signal is sparse (only four spikes) in frequency domain. Therefore, this is essentially a compressed sensing problem and ℓ_1-minimization should work well here. Now the problem can be stated as reconstructing the original signal $\bar{u}$ from *random samples* of the observed signal $\tilde{u}$ using our fast ℓ_1-minimization algorithm. In our experiments, the magnitudes a and b are generated from $\mathcal{U}(-1, 1)$; frequencies α and β are random multiples of $\frac{2\pi}{n}$, i.e. $\alpha = k_1\frac{2\pi}{n}$ and $\alpha = k_2\frac{2\pi}{n}$, with k_i taken from $\{0, 1, \ldots, n-1\}$ randomly and n denotes the dimension. We let I be a random subset of $\{1, 2, \ldots, n\}$ and $f = \tilde{u}(I)$, and take A and $A^\top$ to be the partial matrix of inverse Fourier matrix and Fourier matrix respectively. Now we perform our algorithm adopting the same stopping criteria as in section 4.6.2, and obtain a reconstructed signal denoted as x. Notice that reconstructed signal x is in

Fourier the domain, not in the physical domain. Thus we take an inverse Fourier transform to get the reconstructed signal in physical domain, denoted as u^*. Since we know a priori that our solution should have four spikes in Fourier domain, before we take the inverse Fourier transform, we pick the four spikes with largest magnitudes and set the rest of the entries to be zero. Some numerical results are given in Figure 4.8-4.11. Our experiments show that the larger the noise level is, the more random samples we need for a reliable reconstruction, where reliable means that with high probability (>80%) of getting the frequency back exactly. As for the magnitudes a and b, our algorithm cannot guarantee to recover them exactly (as one can see in Figure 4.8-4.11). However, frequency information is much more important than magnitudes in the sense that the reconstructed signal is less sensitive to errors in magnitudes than errors in frequencies (see bottom figures in Figure 4.8-4.11). On the other hand, once we recover the right frequencies, one can use hardware to estimate magnitudes accurately.

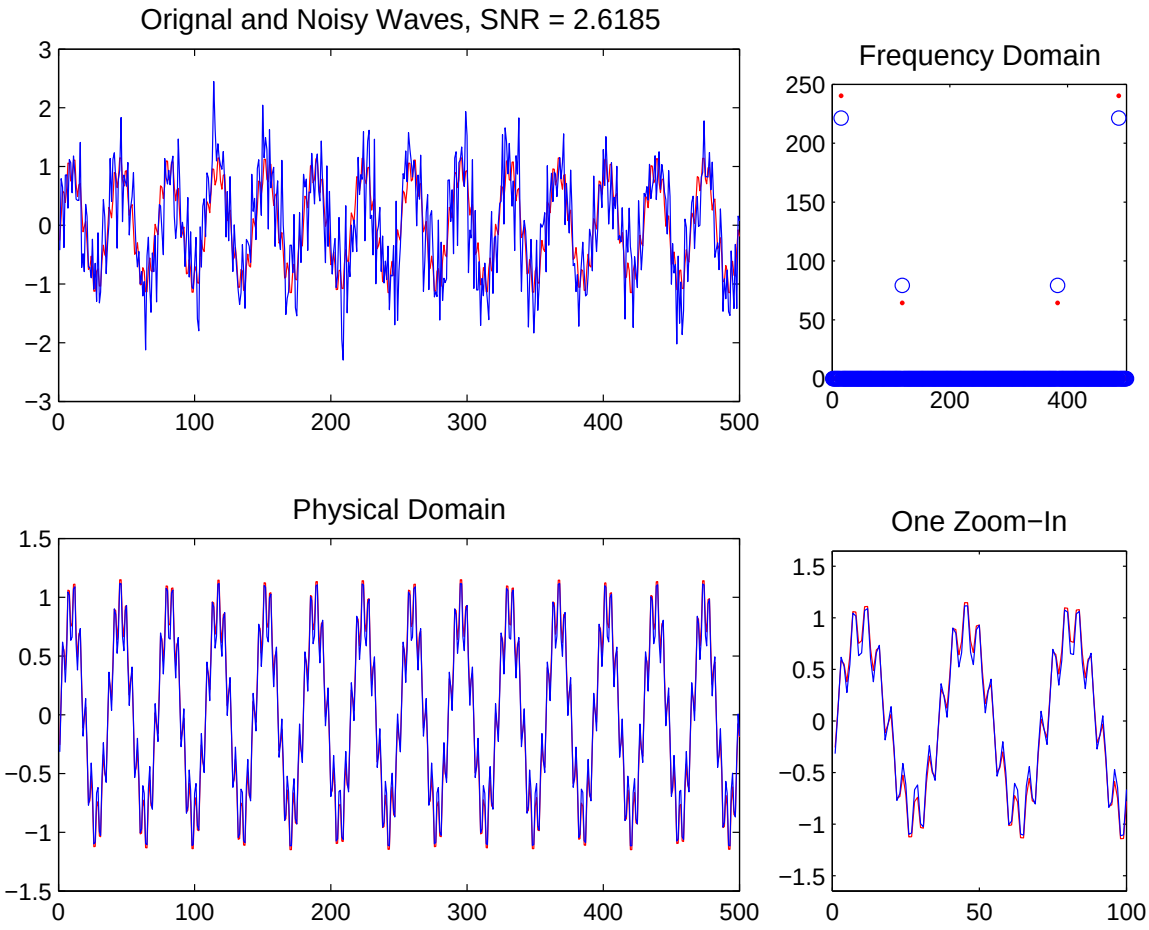


Figure 4.8: Reconstruction using 20% random samples of $\tilde{u}$ with SNR= 2.6185. The upper left figure shows the original (red) and noisy (blue) signals; the upper right shows the reconstruction (blue circle) v.s. original signal (red dots) in Fourier domain in terms of their magnitudes (i.e. $|\widehat{u}^*|$ v.s. $|\widehat{\tilde{u}}|$); bottom left shows the reconstructed (blue) v.s. original (red) signal in physical domain; and bottom right shows one close-up of the figure at bottom left.

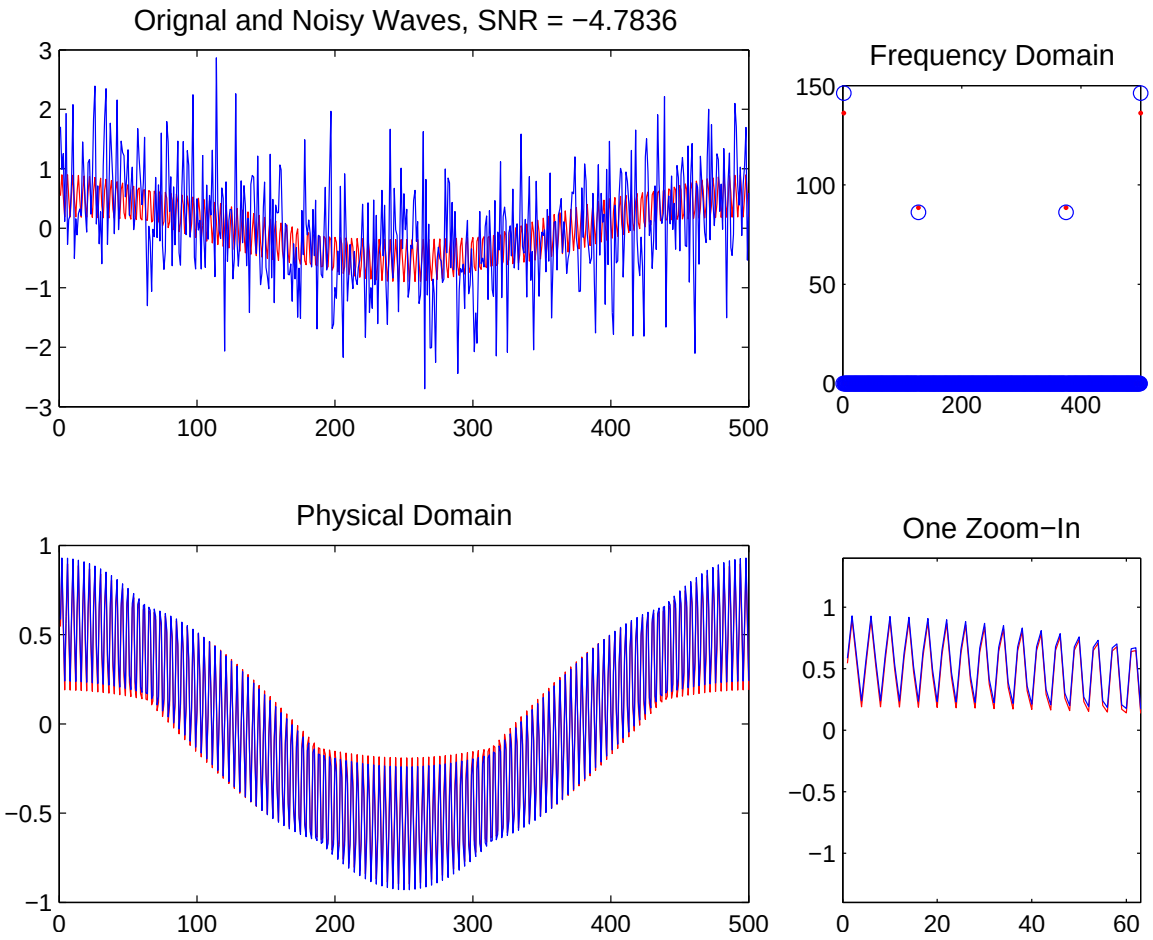


Figure 4.9: Reconstruction using 40% random samples of $\tilde{u}$ with SNR= -4.7836. The upper left figure shows the original (red) and noisy (blue) signals; the upper right shows the reconstruction (blue circle) v.s. original signal (red dots) in Fourier domain in terms of their magnitudes (i.e. $|\widehat{u}^*|$ v.s. $|\widehat{\tilde{u}}|$); bottom left shows the reconstructed (blue) v.s. original (red) signal in physical domain; and bottom right shows one close-up of the figure at bottom left.

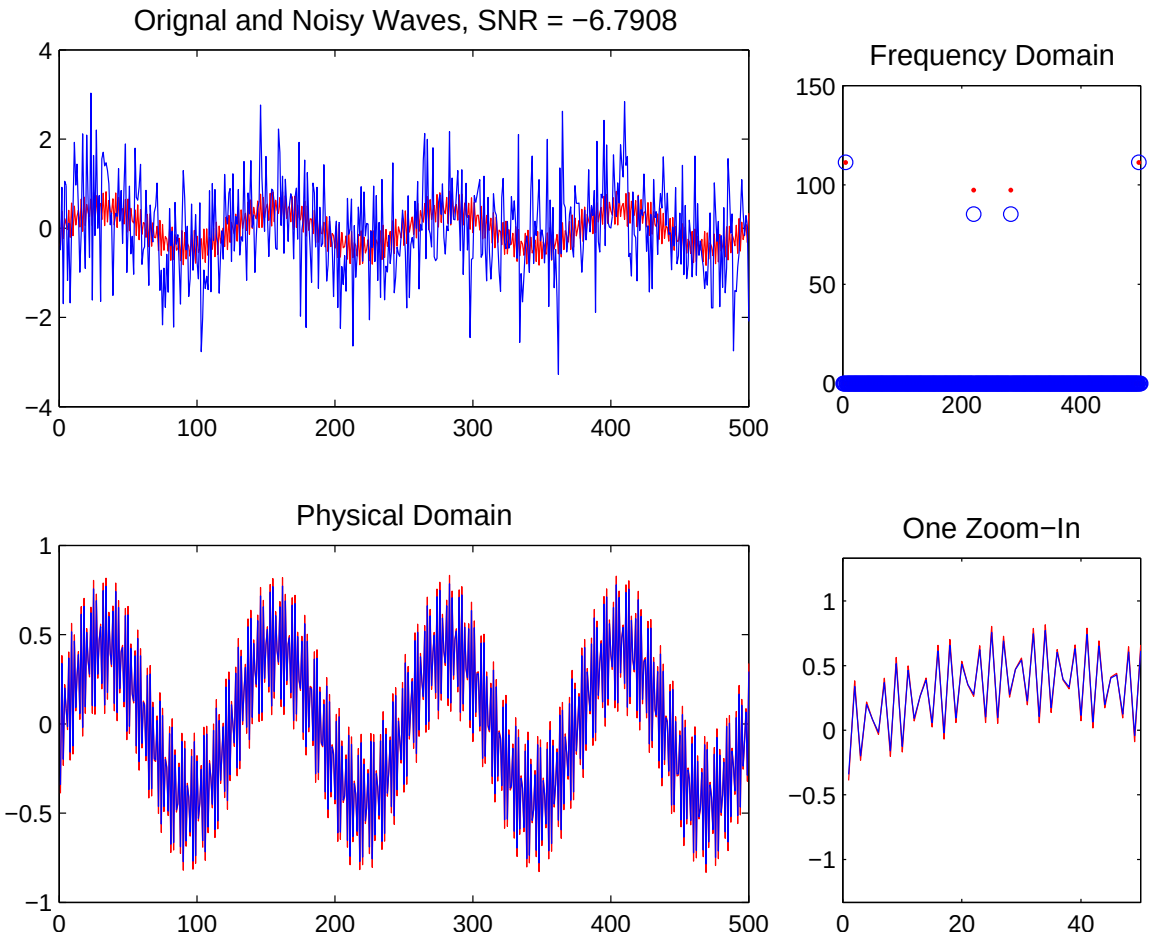


Figure 4.10: Reconstruction using 60% random samples of $\tilde{u}$ with SNR= -6.7908. The upper left figure shows the original (red) and noisy (blue) signals; the upper right shows the reconstruction (blue circle) v.s. original signal (red dots) in Fourier domain in terms of their magnitudes (i.e. $|\widehat{u}^*|$ v.s. $|\widehat{\tilde{u}}|$); bottom left shows the reconstructed (blue) v.s. original (red) signal in physical domain; and bottom right shows one close-up of the figure at bottom left.

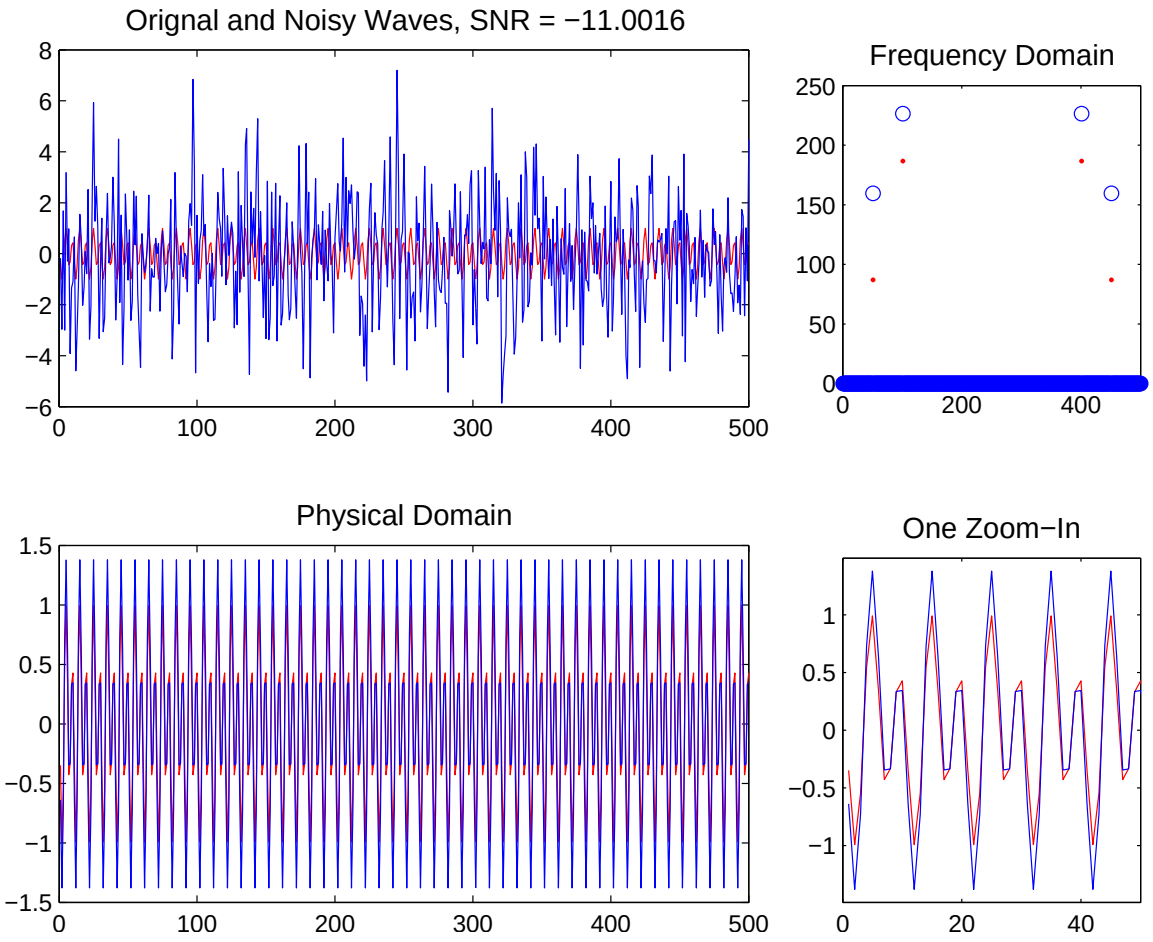


Figure 4.11: Reconstruction using 80% random samples of $\tilde{u}$ with SNR= −11.0016. The upper left figure shows the original (red) and noisy (blue) signals; the upper right shows the reconstruction (blue circle) v.s. original signal (red dots) in Fourier domain in terms of their magnitudes (i.e. $|\widehat{u}^*|$ v.s. $|\widehat{\tilde{u}}|$); bottom left shows the reconstructed (blue) v.s. original (red) signal in physical domain; and bottom right shows one close-up of the figure at bottom left.

CHAPTER 5

Application of ℓ_1-Minimizations to Needle Localization in Ultrasound Images

5.1 Medical Background

Image guided interventions have become the standard of care for many surgical procedures. Optimal visualization of the object of interest and biopsy needle in ultrasound images requires the use of specialized biopsy needles and high cost, cart-based ultrasound units. The success of image guided interventions is dependent on anatomic knowledge, visualization, and precise tracking and control of the biopsy needle. A majority of medical care-providers utilize low resolution ultrasound units. In addition, many office-based or emergency department procedures are performed using generic (non-specialized) needles. Unfortunately, the quality of the imagery obtained by most ultrasound units does not allow for clear and concise visualization of a regular needle during many needle-based procedures. The inability to clearly see the tip of a needle in relation to the object of interest (e.g., a vein, artery, or mass) makes such image guided interventions less accurate.

In view of the inadequacy of ultrasound technology identifying inserted needles with desired resolution, a new and improved system for tracking such needles needs to be developed. A more accurate method for localizing the distal tip of

inserted needles will greatly improve the efficacy and safety of ultrasound image-guided interventions. In this chapter, we shall employ the TV-based model (4.35) (or equivalently (4.36)), and the split Bregman iterations (4.37) to solve the needle tracking problem for ultrasound images.

5.2 Mathematical Model

We denote the video frames of ultrasound images as $I(x,t)$ with $0 \leq I(x,t) \leq 1$, and define the integrated difference of frames as

$$f(x,\tau) := \int_{\tau-\delta}^{\tau} \big|G_\sigma(x) * \partial_t I(x,t)\big| dt, \tag{5.1}$$

where G_σ is Gaussian with standard deviation σ. If the motions of the needle, e.g. jiggling or insertion, are different from the motions of the tissues and organs, which is usually the case, then in $f(x,\tau)$ we can see regions with such motions highlighted. However these regions in $f(x,\tau)$ are usually not very clear and have noisy boundaries. Therefore, a robust and efficient segmentation on $f(x,\tau)$ for each τ is needed. Since we will focus on the segmentation of $f(x,\tau)$ for each fixed τ, we will omit the variable τ and denote $f(x,\tau)$ as $f(x)$ for simplicity.

There are numerous image segmentation methods in the literature [108, 169, 171–177]. Since the image $f(x)$ defined in (5.1) is close to binary, we consider the following energy introduced in [169]

$$E(u) = \int g(x)|\nabla u(x)|dx + \lambda \int |u(x) - f(x)|dx. \tag{5.2}$$

It is shown in [169] that for any minimizer u of (5.2) and for almost all threshold

$\mu \in [0,1]$, the characteristic function

$$\mathbf{1}_{\Omega(\mu)=\{x:u(x)>\mu\}}(x)$$

is a *global minimizer* of the corresponding geometric active contour model (see [169] for more details). Therefore, a segmentation of $f(x)$ can be obtained by first computing a minimizer of (5.2) and then letting $\Omega := \{x : u(x) > 0.5\}$. Now the key issue here is to minimize (5.2) efficiently.

To minimize the energy (5.2) efficiently, we adopt the idea of the split Bregman method introduced in [170]. The derivation is essentially the same as that in Section 4.5 (which is a more general version). Here we shall repeat it for completeness. Define

$$|d|_* := g(x)\sqrt{d_1^2 + d_2^2} + \lambda|d_3| \quad \text{and} \quad Fu := (\nabla u^T, u - f)^T,$$

then minimizing energy (5.2) is equivalent to

$$\begin{aligned} \text{Minimize} \quad & \int |d|_* \\ \text{s.t.} \quad & d = Fu. \end{aligned} \tag{5.3}$$

After "Bregmanizing" the constrained optimization problem (5.3), we obtain the following algorithm which minimizes the original energy (5.2) rather efficiently (the derivation is similar to that in [170]),

$$\begin{aligned} (u^{k+1}, d^{k+1}) &= \arg\min_{u,d} \int |d|_* + \frac{\mu}{2}\|d - Fu - b^k\|_2^2 \\ b^{k+1} &= b^k + \left(Fu^{k+1} - d^{k+1}\right). \end{aligned} \tag{5.4}$$

For convenience, we denote $\bar{d} = (d_1, d_2)^T$ and hence $d = (\bar{d}, d_3)^T$. Similarly, we

can define $\bar{b}$ and b. Then we introduce the following algorithm to solve (5.4):

Algorithm 1. *We start with* $d^0 = \boldsymbol{0}$ *and* $b^0 = \boldsymbol{0}$.

1. *First update* u *by solving*

$$(-\nabla^2 + I)u^{k+1} = \nabla \cdot (\bar{b}^k - \bar{d}^k) + d_3^k + f - b_3^k;$$

2. *Then update* d *by*

$$\begin{aligned} d_1^{k+1} &= \max(s^k - \frac{g(x)}{\mu}, 0) \cdot \frac{u_x^k + b_1^k}{s^k}, \\ d_2^{k+1} &= \max(s^k - \frac{g(x)}{\mu}, 0) \cdot \frac{u_y^k + b_2^k}{s^k}, \\ d_3^{k+1} &= shrink(u^k - f + b_3^k, \frac{\lambda}{\mu}), \end{aligned}$$

   where $s^k = |\nabla u^k + \bar{b}^k|$.

3. *Finally update* b^{k+1} *by*

$$b^{k+1} = b^k + \big(F(u^{k+1}) - d^{k+1}\big);$$

4. *If* $\frac{\|u^{k+1}-u^k\|}{\|u^k\|} > tol$, *go back to step 1 and repeat.*

The Algorithm 1 is very efficient in terms of total number of iterations and the cost for each iteration. According to our experiments, it usually only takes about 30 iterations until $\frac{\|u^{k+1}-u^k\|}{\|u^k\|} \approx 10^{-3}$. For each iteration in Algorithm 1, the major calculation is in step 1, where the PDE can be solved rather efficiently by either FFT, for periodic boundary condition, or multigrid method, for Neumann and Dirichlet boundary conditions. An example is given in the following Figure 5.1

where noise was added to the original image. We note that the image is provided by Laboratory of Neural Imaging, Center for Computational Biology, UCLA.

For the special image $f(x)$ obtained from frames of ultrasound images by (5.1), the object of interest in $f(x)$ is either a needle or the tip of the needle, which are both simple geometric objects. Therefore, we can stop our iteration at an even earlier stage (e.g. in our experiments, we only perform two iterations) and the segmentation results would not change much if more iterations were carried out. The efficiency of Algorithm 1 ensures that the entire needle localization procedure can be finished in real-time. To be precise, by "real-time" we mean that the total time spent by the entire numerical procedure is no greater than that spent by the ultrasound machine in acquiring each image frame. A detailed description of the needle localization procedure will be given in next section.

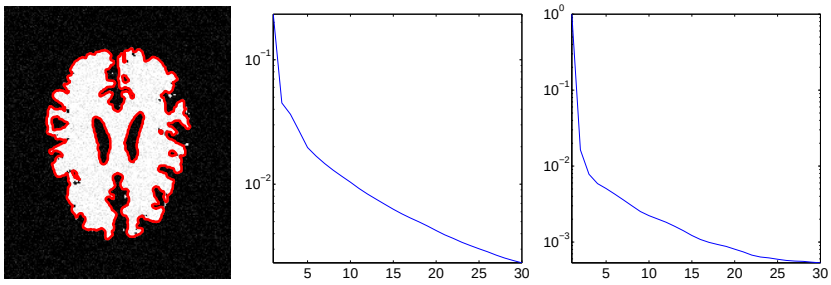


Figure 5.1: The left figure shows segmentation result using Algorithm 1; the middle one is the decay of $\frac{\|d-Fu^k\|}{\|d\|}$; and the right one is the decay of $\frac{\|u^{k+1}-u^k\|}{\|u^k\|}$.

5.3 Schematic Descriptions of Needle Detection and Tracking Procedure

The entire needle localization procedure can be decomposed into two phases. The first phase is to locate the needle in the images at the very beginning, based on a few seconds' image frames. During this phase, one can jiggle the needle or gently

poke the tissues to help our algorithm locate the needle fast and accurately. The second phase is to track the motion of the tip of the needle when it moves.

5.3.1 Phase I

To locate the needle when it is first inserted into the tissue, we perform the following operations:

1. Obtain $f(x)$ using (5.1) based on the previous 1-2 seconds' frames, denoted as $I(x,t)$;

2. Segment the regions using (5.4) via the Algorithm 1 (with 2 iterations);

3. Regularize the region obtained by step 2 via Algorithm 3, the fast area preserving mean curvature motion proposed by [87–89];

4. Obtain the skeleton of the regularized region which represents the needle, and then the tip of the needle can be located.

To help localize the needle based on $f(x)$, one could gently jiggle the needle, in order to differentiate its motion from that of the tissues or organs. The following Figure 5.2 illustrates the four steps described above. We first note that it is obviously crucial to consider $f(x)$ instead of any single frame in order to rule out other regions with comparable intensities as the needle (e.g. some tissues or organs). The left two figures in Figure 5.3 show that if we perform segmentation directly on a single frame, we will capture several regions besides the needle. We also note that the third step above is important because otherwise, we may not get a single line representing the needle, but several branches (see the right figure in Figure 5.3). In step 4, there is always an ambiguity of the tip (it could be the alternative end of the line). However the ambiguity can be easily removed

whenever the needle starts moving. Therefore, here and in the experiments below, we assume the tip is picked up correctly.

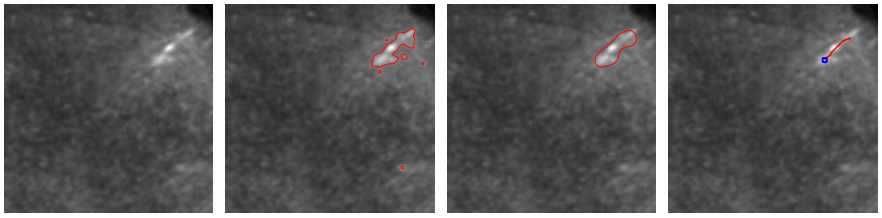


Figure 5.2: The four figures from left to right describes the four steps, and the four images are the same one $f(x)$ obtained by (5.1).

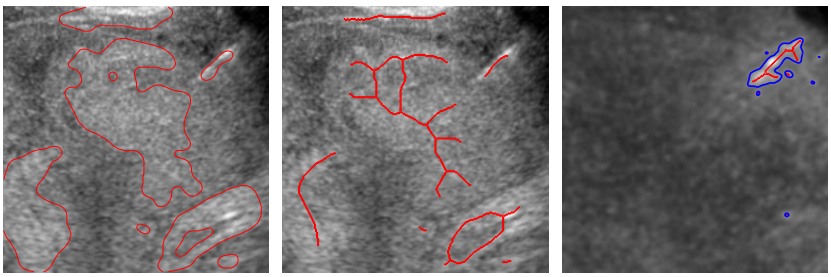


Figure 5.3: Left figure shows direct segmentation of one single frame; middle one shows the skeletons extracted from the segmented regions; right one shows the importance of step 3 in Phase I, where the blue curve is represented by the solution u obtained form step 2, and the red one is the skeleton by step 4.

5.3.2 Phase II

The second phase is to track the movements of the tip of the needle starting from the location we obtained from Phase I. We perform the following operations:

1. Obtain $f(x)$ using (5.1) based on the current and the previous 1-2 frames;
2. Segment the regions using (5.4) via the Algorithm 1 (with 2 iterations);
3. Regularize the region obtained by step 2 via Algorithm 3, the fast area preserving mean curvature motion proposed by [87–89];

4. Shrink the region, which possibly has disconnected components, to points and then choose one point from the set of points that is closest to the previous tracked location.

The following Figure 5.4 illustrates the four steps described above. We note that when the noise level is high or some irregular motions exist in tissues or organs, multiple locations may be captured in step 3, most of which are false detections. Therefore, step 4 affects the smoothness of the overall tracking. Evidently, we can use other ways to estimate the tip of needle based on the multiple locations captured in step 3. For example, if we know a priori that the needle moves in a regular fashion, then we can estimate the location of the tip based on the current multiple choices and the previously chosen locations such that the overall motion curve is smooth. For our experiments in Section 5.4, we only use the simpler operation described in step 4 because the needle moves in an irregular fashion. However, the result of the overall tracking is still quite satisfactory. We also note that in step 1, instead of considering the entire image $f(x)$, we can just consider a patch of $f(x)$ that centered at the previously located point (location of the tip in the previous frame). In this way, we can save some computations and also increase the smoothness of the overall tracking. Again, this only works when the motion of the needle is not too fast. For this reason, we will still use the entire image $f(x)$ in our experiments in Section 5.4.

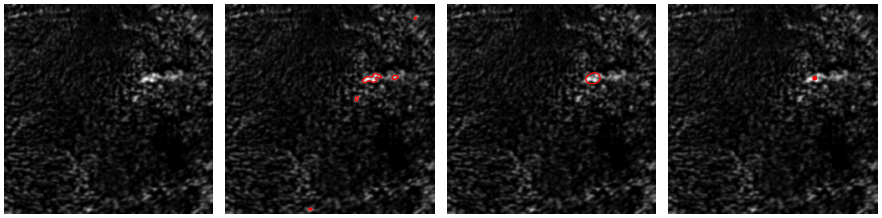


Figure 5.4: The four figures from left to right describes the four steps.

5.4 Numerical Results

All of the frames of ultrasound images are obtained by a Sonosite (Titan) ultrasound machine. The ultrasound machine captures 20 frames per second. In our following experiments, 120 frames are used, including 20 frames in Phase I and 100 frames in Phase II. Each image is of size 251×251. In Figure 5.5 we present 5 of the 20 frames in Phase I, and in Figure 5.7 we present 12 of the 100 frames in Phase II.

The numerical results for Phase I are given in Figure 5.6, and those for Phase II are given in Figure 5.8. We note that the PDE in (1) of Algorithm 1 is solved by FFT. Here we also provide a ground truth in Figure 5.9 as validation of our results, where we manually selected the positions of the needle based on neighboring frames. We note that for almost all of the frames during Phase II, the tracking is rather accurate. However for some of the frames, the localization is not very accurate, for example the fourth figure in the first row of Figure 5.8. The reason is because of acoustic shadows in some image frames, which appear in $f(x)$ with high intensities and conceal the movement of the tip of the needle (see the middle figure of Figure 5.10). However, an acoustic shadow only seems to appear in $f(x)$ occasionally when we extract the needle, instead of inserting the needle, and an accurate tracking of the needle is only required during insertion. Therefore in practice, this error is not an issue and will not affect the safety concerns during image guided surgical operations.

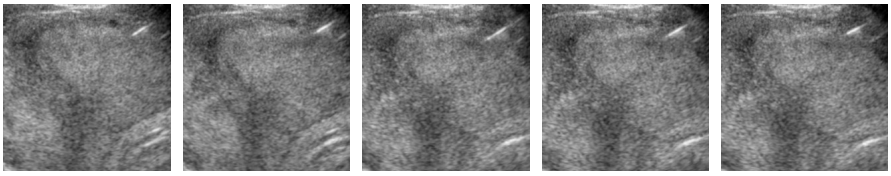


Figure 5.5: Images from left to right are 5 sample frames among total 20 frames of ultrasound images during Phase I.

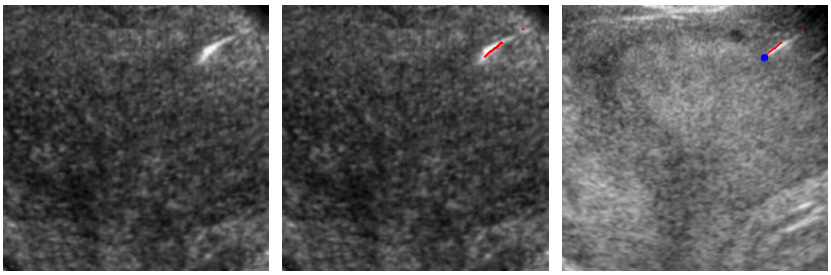


Figure 5.6: Left figure is $f(x)$ obtained from the 20 frames; middle one shows the result of localization of the body of the needle; right one shows the result of localization on the first image frame in Figure 5.5, where the blue dot indicates the tip of the needle.

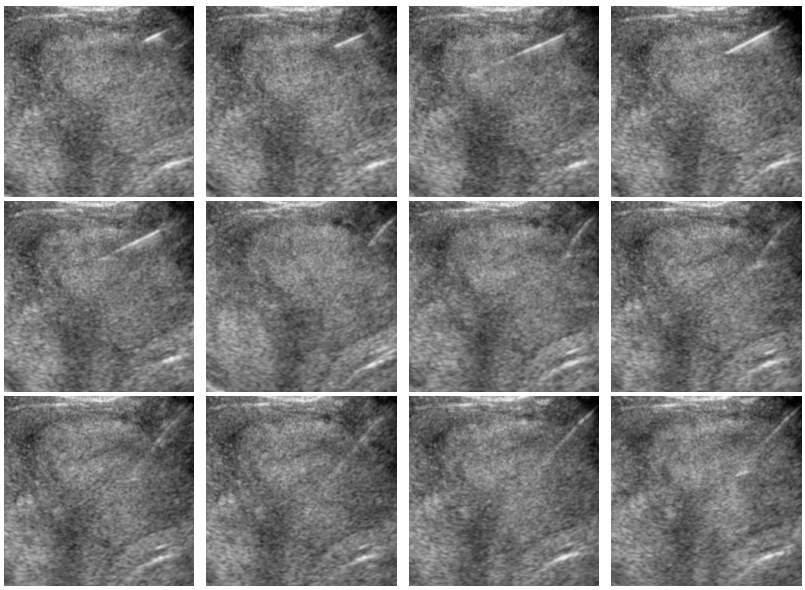


Figure 5.7: Images above are 12 sample frames among total 100 frames of ultrasound images during Phase II.

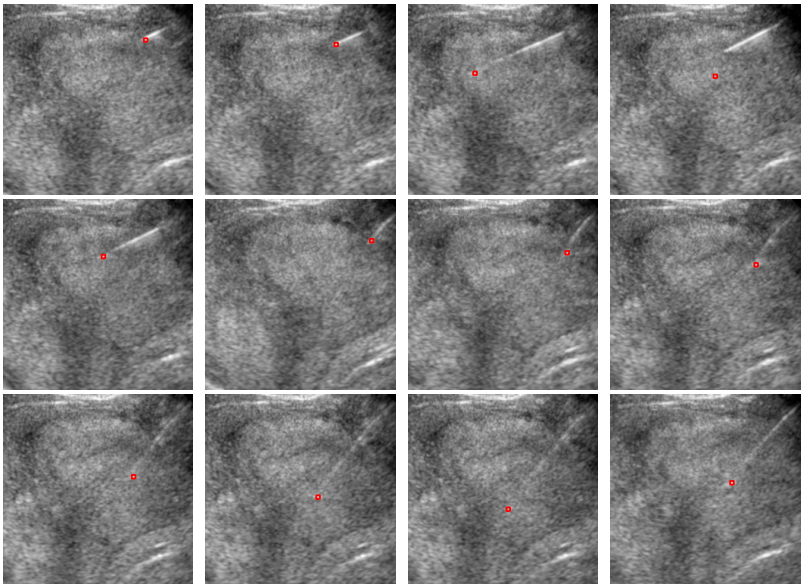


Figure 5.8: Tracking results of the 12 sample frames in Phase II shown in Figure 5.7.

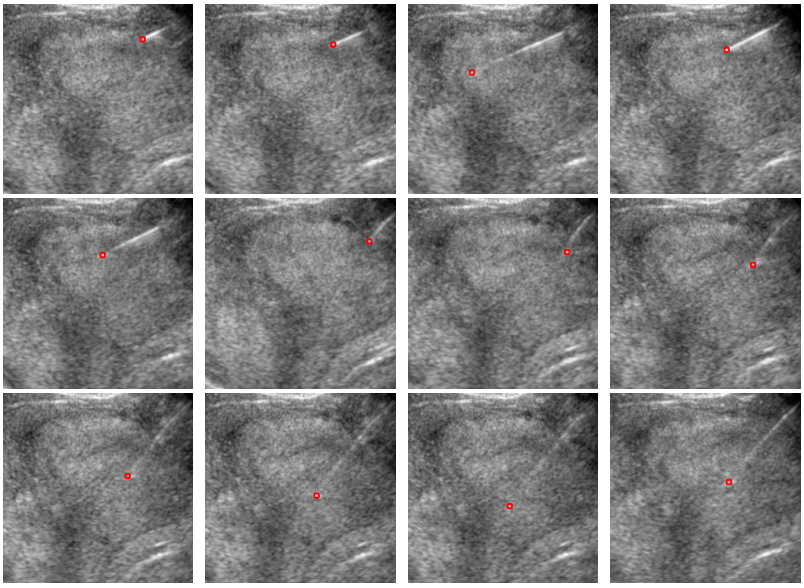


Figure 5.9: Manual segmentation results of the 12 sample frames in Phase II shown in Figure 5.7.

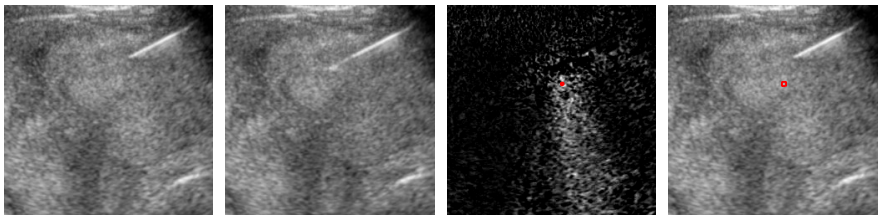


Figure 5.10: First figure is the current frame as shown in the fourth figure in first row of Figure 5.7; second figure is the previous frame of the first figure; third figure shows the corresponding $f(x)$ obtained from the first two figures and the red dot is the tracking result; the last one shows the tracking result on the current frame which is the same figure as in the upper fight figure of Figure 5.8.

Bibliography

[1] A. Buades, B. Coll, and J-M. Morel. On image denoising methods, Multiscale Model. Simul., 4(2), 490–530, 2005.

[2] A. Buades and B. Coll and J-M. Morel. Neighborhood filters and PDE's, Numer. Math. 105(1), 1–34, 2006.

[3] M. Burger, G. Gilboa, S. Osher and J. Xu. Nonlinear inverse scale space methods, Commun. Math. Sci., 4(1), 179–212, 2006.

[4] P. Bhat, S. Ingram and G. Turk. Geometric texture synthesis by example, Second Eurographics Symposium on Geometry Processing, 2004.

[5] U. Clarenz, U. Diewald, and M. Rumpf. Anisotropic diffusion in surface processing, T. Ertl, B. Hamann, and A. Varshney, editors, Proc. IEEE Vis., 397–405, 2000.

[6] M. Desbrun, M. Meyer, P. Schröder, and A. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow, ACM SIGGRAPH 1999.

[7] M. Desbrun, M. Meyer, P. Schröder, and A. Barr. Anisotropic feature-preserving denoising of height fields and bivariate data, Graphics Interface, 2000.

[8] M. Elsey and S. Esedoglu, Analogue of the Total Variation Denoising Model in the Context of Geometry Processing, CAM-Report 07–31, 2007.

[9] A. Elmoataz and O. Lezoray and S. Bougleux, Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing, preprint, 2007.

[10] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity, under revision for ACM TOG 2005.

[11] T. Gatzke, C. Grimm, M. Garland and S. Zelinka. Curvature maps for local shape comparison, Shape Modeling International (SMI), 244–256, 2005.

[12] G. Gilboa and S. Osher, Nonlocal Linear Image Regularization and Supervised Segmentation, Multiscale Model. Simul., 6(2), 595–630, 2007.

[13] G. Gilboa and S. Osher, Nonlocal Operators with Applications to Image Processing, CAM-Report 07–23, 2007.

[14] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle. Piecewise smooth surface reconstruction, ACM SIGGRAPH, 295–302, 1994.

[15] S. Kindermann, S. Osher and P. W. Jones, Deblurring and Denoising of Images by Nonlocal Functionals, Multiscale Model. Simul., 4(4), 1091–1115, 2005.

[16] P. Perona and J. Malik, Scale space and edge detection using anisotropic diffusion, IEEE Trans. Patt. Anal. Mach. Intell., 12, 629-639, 1990.

[17] A. Sharf, M. Alexa and D. Cohen-Or. Context-based surface completion, ACM Trans. Graph., 23(3), 878–887. Proceedings of ACM SIGGRAPH 2004.

[18] R. Tsai, L.-T. Cheng, S. J. Osher, and H. K. Zhao. Fast sweeping method for a class of Hamilton-Jacobi equations, SIAM J. Numer. Analy., 41, 673–694, 2003.

[19] C. Tomasi and R. Manduchi, Bilateral filtering for gray and color images, Sixth International Conference on Computer Vision, 839–46, 1998.

[20] T. Tasdizen, R. Whitaker, P. Burchard, and S. Osher. Geometric surface processing via normal maps, ACM Trans. Graph., 22, 1012–1033, 2003.

[21] T. Tasdizen, R. T. Whitaker, P. Burchard, and S. Osher. Geometric surface smoothing via anisotropic diffusion of normals, Proc. IEEE Vis., 125–132, 2002.

[22] S. Yoshizawa, A. Belyaev and H. P. Seidel, Smoothing by Example: Mesh Denoising by Averaging with Similarity-based Weights, IEEE International Conference on Shape Modeling and Applications, 38–44, 2006.

[23] J. Xu and S. Osher, Iterative regularizaiton and nonlinear inverse scale space applied to wavelet based denoising, IEEE Image Proc., 16(2), 534–544, 2007.

[24] S. Zelinka and M. Garland. Similarity-based surface modelling using geodesic fans, Second Eurographics Symposium on Geometry Processing, 209–218, 2004.

[25] D. Zhou and B. Scholkopf, A regularization framework for learning from graph data, ICML workshop on Statistical Relation Learning and Its Connections to Other Fields, 2004.

[26] N. F. Kassell and J. C. Torner, Aneurysmal rebleeding: a preliminary report from the Cooperative Aneurysm Study, Neurosurgery, 13(5), 479–81, 1983.

[27] Unruptured intracranial aneurysms–risk of rupture and risks of surgical intervention, by International Study of Unruptured Intracranial Aneurysms Investigators, N Engl J Med, 339(24), 1725–33, 1998.

[28] D. O. Wiebers, J. P. Whisnant, J. 3rd. Huston, I. Meissner, R.D. Jr. Brown, D. G. Piepgras, G. S. Forbes, K. Thielen, D. Nichols, W. M. O'Fallon, J.

Peacock, L. Jaeger, N. F. Kassell, G. L. Kongable-Beckman and J. C. Torner, Unruptured intracranial aneurysms: natural history, clinical outcome, and risks of surgical and endovascular treatment, Lancet 362(9378): 103-10, 2003.

[29] Marieke J.H. Wermer, Irene C. van der Schaaf, Ale Algra, and Gabriël J.E. Rinkel., Risk of rupture of unruptured intracranial aneurysms in relation to patient and aneurysm characteristics, Stroke, 38, 1404–1410, 2007.

[30] G. Kanizsa, Organization in Vision, Praeger, New York, 1979.

[31] D. Kersten, High-level vision and statistical inference, In M. S. Gazzaniga, editor, The New Cognitive Neurosciences, 353–363. The MIT Press, Cambridge, MA, USA.

[32] D. C. Knill and D. Kersten, Apparent surface curvature affects lightness perception, Nature, 351, 228–230, 1991.

[33] D. Geiger, H. K. Pao and N. Rubin, Salient and Multiple illusory surfaces, IEEE Computer Society Conference on Conputer Vision and Pattern Recognition, San Barbara, CA, June 1998.

[34] A. Sarti and G. Citti, Subjective surfaces and Riemann mean curvature flow of graphs, Acta Math. Univ. Comenianae, Vol. LXX, 1(2001), 85-103, Proceedings of Algoritmy 2000.

[35] A. Sarti, R. Malladi and J. A. Sethian, Subjective surfaces: A geometric model for boundary completion, Intl J. Comp. Vision, 46(3), 201–221, 2002.

[36] W. Zhu and T. F. Chan, Capture illusory contours: A level set based approach, UCLA CAM Report 03-65, 2003.

[37] W. Zhu and T. F. Chan, Illusory contours using shape information. UCLA CAM Tech. Report, 03-09, 2005.

[38] Y. M. Jung and J. Shen, First-order modeling and stability analysis of illusory contours, Journal of Visual Communication and Image Representation archive, 19(1), 42–55, 2008.

[39] P. J. Yim, J. J. Cebral, R. Mullick, H. B. Marcos and P. L. Choyke, Vessel surface reconstruction with a tubular deformable model, Medical Imaging, IEEE Transactions on, 20(12), 1411–1421, 2001.

[40] J. Chen and A. A. Amini, Quantifying 3-D vascular structures in MRA images using hybrid PDE and geometric deformable models, Medical Imaging, IEEE Transactions on, 23(10), 1251–1262, 2004.

[41] A. Buades, A. Chien, J. M. Morel and S. J. Osher, Topology preserving linear filtering applied to medical imaging, SIAM J. Imaging Sci., 1(1), 26–50, 2008.

[42] B. Dong, J. Ye, S. J. Osher and I. Dinov, Level set based nonlocal surface restoration, Multiscale Model. Simul., 7(2), 589–598, 2008.

[43] H. K. Zhao, T. F. Chan, B. Merriman and S. J. Osher, A variational level set approach to multiphase motion, J. Comput. Phys., 127, 179–195, 1996.

[44] D. Peng, B. Merriman, S. J. Osher, H. K. Zhao and M. Kang, A PDE-based fast local level set method, J. Comput. Phys., 155, 410–438, 1999.

[45] R. Goldman, Curvature formulas for implicit curves and surfaces, Computer Aided Geometric Design, 22, 632–658, 2005.

[46] L. Regli, A. Uske and N. de Tribolet, Endovascular coil placement compared with surgical clipping for the treatment of unruptured middle cerebral artery aneurysms: a consecutive series, J. Neurosurg, 90, 1025–1030, 1999.

[47] L. Alvarez, F. Guichard, P-L Lions and J.M. Morel, Axioms and fundamental equations of image processing. Archive for Rational Mechanics and Analysis., 16(9), 200–257, 1993.

[48] Frédéric Guichard and Jean-Michel Morel, Image Analysis and P.D.E.'s, IPAM GBM Tutorials, March 27 - April 6, 2001.

`http://www.ipam.ucla.edu/publications/gbm2001/gbmtut_jmorel.pdf`

[49] Martin Reuter, Franz-Erich Wolter and Niklas Peinecke, Laplace-Beltrami spectra as Shape-DNA of surfaces and solids, Computer-Aided Design, vol. 38 (4), 342–366, April 2006.

[50] F. Arandiga, R. Donat and A. Harten. Multiresolution based on weighted averages of the hat function I: Linear Reconstruction Techniques. UCLA CAM Reports 96-25, Aug. 1996.

[51] S. Buss and J. Fillmore. Spherical averages and applications to spherical splines and interpolation. ACM Transactions on Graphics, 20(2), 95–126, 2001.

[52] D. L. Collins, P. Neelin, T. Peters, A. C. Evans. Automatic 3D intersubject registration of MR volumetric data in standardized Talairach space. J. Comput. Assist. Tomogr. 18 (2), 192–205, 1994.

[53] I. Daubechies. Ten Lectures on Wavelets. CBMS-NSF Lecture Notes nr. 61, SIAM, 1992.

[54] S. Mallat. A Wavelet Tour of Signal Processing. 2nd ed. New York: Academic, 1999.

[55] I. D. Dinov, J. W. Boscardin, M. S. Mega, E. L. Sowell, A. W. Toga. A Wavelet-Based Statistical Analysis of fMRI data: I. Motivation and Data Distribution Modeling, NeuroInformatics, Humana Press, 3(4), 319–343, 2005.

[56] A. C. Evans, D. L. Collins, P. Neelin, D. MacDonald, M. Kamber, T. S. Marrett. Three-dimensional correlative imaging: applications in human brain mapping. In: Thatcher, R.W., Hallett, M., Zeffiro, T., John, E.R., Huerta, M. (Eds.), Functional Neuroimaging: Technical Foundations. Academic Press, San Diego, 145–162, 1994.

[57] J. D. Gibbons. Nonparametric Statistical Inference, 2nd Ed., M. Dekker, 1985.

[58] X. Gu, Y. Wang, T. F. Chan, P.M. Thompson and S.-T. Yau. Genus Zero Surface Conformal Mapping and Its Application to Brain Surface Mapping. IEEE Transaction on Medical Imaging, 23(8), 949–958, Aug. 2004.

[59] Y. Wang, X. Gu, T. F. Chan, P. M. Thompson and S.-T. Yau, " Intrinsic Brain Surface Conformal Mapping using a Variational Method", Medical Imaging 2004: Image Processing, J. M. Fitzpatrick and M. Sonka (Eds.) Proceedings of SPIE, Vol. 5370, 241–252, 2004.

[60] X. Gu, Y. Wang, T. F. Chan, P. M. Thompson and S.-T. Yau. Brain Surface Conformal Mapping. Human Brain Mapping, 2003.

[61] M. Jin, F. Luo and X. Gu. Computing General Geometric Structures on

Surfaces Using Ricci Flow. Computer-Aided Design, Vol. 39 (8), 663–675, August 2007.

[62] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro and M. Halle. Conformal surface parameterization for texture mapping. IEEE Trans Vis Comput Graph, 6(2):181-9, 2000.

[63] C. Gotsman, X. Gu and A. Sheffer. Fundamentals of spherical parameterization for 3D meshes. ACM Trans Graph, 22(3):358-63, 2003.

[64] X. Gu and S.-T. Yau. Global conformal parameterization. Symposium on geometry processing, 127-37, 2003.

[65] M. Jin, Y. Wang, S.-T. Yau and X. Gu. Optimal global conformal surface parameterization. IEEE Visualization, 267C74, 2004.

[66] D. Nain, S. Hacker, A. Bobick and A. Tannenbaum. A multiscale 3D shape analysis using spherical wavelets. Proc MICCAI, 459–467, Oct 26-29 2005.

[67] D. Nain, S. Hacker, A. Bobick and A. Tannenbaum. A shape-driven 3D segmentation using spherical wavelets. Proc MICCAI, Oct 2-5, 2006.

[68] D. MacDonald, A method for identifying geometrically simple surfaces from three dimensional images. PhD thesis, McGill University, 1998.

[69] Mazziotta JC, Toga AW, Evans AC, Fox PT, Lancaster J, Zilles K, Woods RP, Paus T, Simpson G, Pike B, Holmes CJ, Collins DL, Thompson PM, MacDonald D, Schormann T, Amunts K, Palomero-Gallagher N, Parsons L, Narr KL, Kabani N, A probabilistic atlas and reference system for the human brain: International Consortium for Brain Mapping. Philos Trans R Soc Lond B Biol Sci 356: 1293-1322, 2001.

[70] E. Praun and H. Hoppe, Spherical parametrization and remeshing. ACM Trans. Gr. Vol. 22 (3), July 2003.

[71] W. Sweldens. The lifting scheme: A construction of second generation wavelets. SIAM J. Math. Anal., Vol 29 (2), 511–546, 1997.

[72] P. Schröder and W. Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. Computer Graphics Proceedings, (SIGGRAPH 95), 161–172, 1995.

[73] D. W. Shattuck, S. R. Sandor-Leahy, K. A. Schaper, D. A. Rottenberg, R. M. Leahy. Magnetic resonance image tissue classification using a partial volume model. NeuroImage 13: 856–876, 2001.

[74] Y. Shi, P. M. Thompson , G. I. de Zubicaray, S. Rose, Z. Tu, I. Dinov, A. W. Toga, Direct mapping of hippocampal surfaces with intrinsic shape context, NeuroImage, 37(3), 792-807, 2007.

[75] Y. Shi, P. M. Thompson, I. Dinov, S. Osher, A. W. Toga, Direct cortical mapping via solving partial differential equations on implicit surfaces, Medical Image Analysis, 11(3), 207–223, 2007

[76] J. G. Sled, A. P. Zijdenbos, A. C. Evans. A non-parametric method for automatic correction of intensity non-uniformity in MRI data. IEEE Trans. Med. Imag. 17, 87–97, 1998.

[77] Paul M. Thompson, Arthur W. Toga. A Framework for Computational Anatomy. Springer Berlin / Heidelberg, vol 5, no. 1, 2002.

[78] Paul M. Thompson, Agatha D. Lee, Rebecca A. Dutton, Jennifer A. Geaga, Kiralee M. Hayashi, Mark A. Eckert, Ursula Bellugi, Albert M. Galaburda,

Julie R. Korenberg, Debra L. Mills, Arthur W. Toga, and Allan L. Reiss. Abnormal Cortical Complexity and Thickness Profiles Mapped in Williams Syndrome. The Journal of Neuroscience, vol. 25 no. 16, 2005.

[79] Thompson PM, Hayashi KM, de Zubicaray G, Janke AL, Rose SE, Semple J, Herman D, Hong MS, Dittmer S, Doddrell DM, Toga AW. Dynamics of gray matter loss in Alzheimer's disease. J Neurosci 23: 994-1005, 2003

[80] Paul M. Thompson, Kiralee M. Hayashi, Greig I. de Zubicaray, Andrew L. Janke, Stephen E. Rose, James Semple, Michael S. Hong, David H. Herman, David Gravano, David M. Doddrell and Arthur W. Toga, Mapping hippocampal and ventricular change in Alzheimer disease, Neuroimage, vol 22, no. 4, 1754-1766, 2004.

[81] Z. Tu, S. Zheng, A. Yuille, A. Reiss, R. Dutton, A.Lee, A.Galaburda, I. Dinov, P. Thompson, A. Toga, Automated Extraction of the Cortical Sulci Based on a Supervised Learning Approach, IEEE Tran. on Medical Imaging, vol. 26, no. 4, April, 2007

[82] L. Bronsard and B. Stoth, Volume preserving mean curvature flow as a limit of nonlocal Ginsburg-Landau equation. Technical Report 94-NA-008, Center for Nonlinear Analysis, Charnigie Mellon University, 1994.

[83] D. Levin, Mesh-independent surface interpolation. Geometric Modeling for Scientific visualization, Springer-Verlag, 37–50, 2003.

[84] H. Ishii. A generalization of the Bence, Merriman and Osher algorithm for motion by mean curvature. In A. Damlamian, J. Spruck, and A. Visintin, editors, Curvature Flows and Related Topics, 111–127, Gakkôtosho, Tokyo, 1995.

[85] J.-P. Pons, G. Hermosillo, R. Keriven and O. Faugeras, Maintaining the point correspondence in the level set framework. J. Comput. Phy. Vol 220 (1), Dec. 2006.

[86] M. Pauly, L. Kobbelt and M. Gross, Point-based multiscale surface representation. ACM Trans. on Graph., Vol. 25 (2), 177–193, Apr. 2006.

[87] B. Merriman, J. Bence and S. Osher, Motion of multiple junctions, a level set approach, J. Comput Phys, Vol. 112, 334–363, 1994.

[88] B. Merriman, J. Bence and S. Osher, Diffusion generated motion by mean curvature. In J.E. Taylor, editor, Computational Crystal Growers Workshop, AMS, Rhode Island, 73–83, 1992.

[89] S. Ruuth and B. Wetton, A simple scheme for volume preserving motion by mean curvature, J. Sci Comput, vol. 19, 373–384, 2003.

[90] S. Ruuth, Efficient Algorithms for diffusion-generated motion by mean curvature. Ph.D. thesis, University of British Columbia, Vancouver, Canada, 1996.

[91] S. Ruuth and B. Merriman, Convolution generated motion and generalized Huygens' principles for interface motion. SIAM J. Appl. Math., 60 (3), 868–890, 2000.

[92] S. Ruuth, B. Merriman and S. Osher, Convolution generated motion as a link between cellular automata and continuum pattern dynamics. J. Coput. Phys., 151, 836–861, 1999.

[93] J. Rubinstein and P. Sternberg, Nonlocal reaction-diffusion equations and nuleation, IMA J. Appl. Math. 48, 248–264, 1992.

[94] W. Sweldens, The lifting scheme: A construction of second generation wavelets. SIAM J. Math. Anal., Vol 29 (2), 511–546, 1997.

[95] S. Osher and J. Sethian, Fronts propagting with curvature-dependent speed: Algorithm based on Hamilton-Jacobi formulation. J. Computat. Physics, 79, 12–49, 1988.

[96] M. G. Crandall and P. L. Lions, Viscosity solutions of Hamilton-Jacobi equations, Trans. Amer. Math. Soc., 277, 1–42, 1983.

[97] M. G. Crandall and P. L. Lions, Two approximations of solutions of Hamilton-Jacobi equations, Math. Comput., 43(167), 1–19, 1984.

[98] M. G. Crandall, L. C. Evans and P. L. Lions, Some properties of viscosity solutions of Hamilton-Jacobi equations, Trans. Amer. Math. Soc., 282, 487–502, 1984.

[99] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. I. J. Differential Geom., 33(3), 635–681, 1991.

[100] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. II. Trans. Amer. Math. Soc., 330(1), 321–332, 1992.

[101] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. III. J. Geom. Anal., 2(2):121C150, 1992.

[102] L. C. Evans and J. Spruck. Motion of level sets by mean curvature. IV. J. Geom. Anal., 5(1), 77–114, 1995.

[103] Y. G. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. J. Differential Geom., 33(3), 749–786, 1991.

[104] Y. Giga. Surface evolution equationsCa level set method. Vorlesungsreihe, 44. Rheinische Friedrich-Wilhelms-Universität, Bonn, 2002.

[105] J. Escher and G. Simonett, The volume preserving mean curvature flow near spheres. Proc. Amer. Math. Soc., 126(9), 2789–2796, 1998.

[106] M. G. Crandall, H. Ishii, and P.-L. Lions. User's guide to viscosity solutions of second order partial differential equations. Bull. Amer. Math. Soc. (N.S.), 27(1), 1–67, 1992.

[107] H.K. Zhao, S. Osher, B. Merriman, and M. Kang, Implicit and non-parametric shape reconstruction from unorganized points using variational level set method, Computer Vision and Image Understanding, 80 (3), 295–319 2000.

[108] T. Goldstein, X. Bresson and S. Osher. Geometric Applications of the Split Bregman Method: Segmentation and Surface Reconstruction. CAM-Report 09-06, 2009.

[109] S. Chen, B. Merriman, S. Osher and P. Smereka. A Simple Level Set Method for Solving Stefan Problems, J. Comp. Phys., 135 (1), 8–29, 1997.

[110] M. Bertalmoio, L.T. Cheng, G. Sapiro and S. Osher. Variational problems and partial differential equations on implicit surfaces, J. Comp. Phys., 174 (2), 759–780, December 2001.

[111] I. Ur-Raman, I. Drori, V. Stodden, D. Donoho and P. Schroeder, Multiscale representations of manifold-valued data. Multiscale Modeling & Simulation, Vol. 4 (4), 2005.

[112] K. Ni, D. Roble and T. Chan, A Texture Synthesis Approach to Elastica Inpainting, ACM SIGGRAPH 2007.

[113] T. F. Chan, S. H. Kang and J. Shen. Eulers elastica and curvature based inpainting. SIAM journal on Applied Mathematics 63(2), 564–592, 2002.

[114] M. Bertalmio, G. Sapiro, V. Caselles and C. Ballester. Image inpainting. SIGGRAPH, Computer Graphics Proceedings, 417–424, 2000.

[115] M. Bertalmio, A. L. Bertozzi and G. Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. Proceedings of the International Conference on Computer Vision and Pattern Recognition, 355–362, 2001.

[116] T. Chan and J. Shen, Variational image inpainting, Communications on Pure and Applied Mathematics, 58, 579–619, 2005.

[117] A. Bertozzi, S. Esedoglu and A. Gillette. Inpainting of binary images using the cahn-hilliard equation. IEEE Trans. Image Proc. 60(2), 285–291, 2007.

[118] A. Bertozzi, S. Esedoglu and A. Gillette. Analysis of a two-scale cahn-hilliard model for image inpainting. Multiscale Modeling and Simulation 6(3), 913–936, 2007.

[119] S. Esedoglu and J. Shen. Digital inpainting based on the mumford-shah-euler image model. European J. Appl. Math. 13, 353–370, 2002.

[120] J. A. Dobrosotskaya and A. Bertozzi. A wavelet-laplace variational technique for image deconvolution and inpainting. IEEE Trans. Imag. Proc. 17(5), 657–663, 2008.

[121] M. Bertalmio, L. Vese, G. Sapiro and S. Osher. Simultaneous texture and structure image inpainting. Proceedings of the International Conference on Computer Vision and Pattern Recognition, 707–712, 2003.

[122] J. Cai, R. Chan, L. Shen and Z. Shen, Convergence analysis of tight framelet approach for missing data recovery, Advances in Computational Mathematics, preprint.

[123] J. Cai, R. Chan and Z. Shen, A framelet-based image inpainting algorithm, Applied and Computational Harmonic Analysis, 24, 131–149, 2008.

[124] M. Elad, J.-L. Starck, P. Querre, and D. Donoho, Simultaneous cartoon and texture image inpainting using morphplogical component analysis (MCA), Applied and Computational Harmonic Analysis, 19, 340–358, 2005.

[125] T. F. Chan, J. Shen and H.-M. Zhou, Total variation wavelet inpainting, J. Math. Imaging Vision, 25, 107–125, 2006.

[126] J. Davis, S. Marschner, M. Garr and M. Levoy. Filling holes in complex surfaces using volumetric diffusion. First International Symposium on 3D Data Processing, Visualization, and Transmission, June 2002.

[127] J. Verdera, V. Caselles, M. Bertalmio and G. Sapiro. Inpainting surface holes. Proc. Intl Conf. Image Processing, 2003.

[128] N. Amenta, M. Bern, M. Kamvysselis. A New Voronoi-Based Surface Reconstruction Algorithm. Proc. SIGGRAPH'98, ACM, 1998.

[129] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva and G. Taubin. The Ball-Pivoting Algorithm for Surface Reconstruction. IEEE Transactions on Visualization and Computer Graphics, October-December, 1999.

[130] R. Whitaker. A Level-set Approach to 3D Reconstruction from range data. International Journal of Computer Vision, Vol. 29, No. 3, October, 1998.

[131] M.D. Wheeler, Y. Sato, K. Ikeuchi. Consensus Surfaces for Modeling 3D Objects from Multiple Range Images. Proc. ICCV'98, 1998.

[132] E. Tadmor, S. Nezzar and L. Vese. A Multiscale Image Representation Using Hierarchical (BV,L2) Decomposition. Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal, Volume 2, Number 4, 554-579, 2004.

[133] E. Tadmor, S. Nezzar and L. Vese, Multiscale Hierarchical Decomposition of Images with Applications to Deblurring, Denoising and Segmentation, CAM-Report 08-05, February 2008.

[134] S. Mallat and Z. Zhang, Matching pursuit in a time-frequency dictionary, IEEE Trans. Signal Process., vol. 41, no. 12, 3397–3415, Dec. 1993.

[135] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, presented at the 27th Annu. Asilomar Conf. Signals, Systems, and Computers, 1993.

[136] S. S. Chen, D. L. Donoho, and M. A. Saunders, Atomic decomposition by basis pursuit, SIAM Rev., vol. 43, no. 1, 129–59, 2001.

[137] J. Darbon and S. Osher. Fast discrete optimizations for sparse approximations and deconvolutions. preprint 2007.

[138] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for compressed sensing and related problems. SIAM J. Imaging Sciences 1(1)., pages 143–168, 2008.

[139] J. Cai, S. Osher, and Z. Shen. Linearized Bregman iterations for compressed sensing. Math. Comp., 2008. to appear, see also UCLA CAM Report 08-06.

[140] J. Cai, S. Osher, and Z. Shen. Convergence of the linearized Bregman iteration for ℓ_1-norm minimization. UCLA CAM Report 08-52, 2008.

[141] E. Candes, T. Tao, Decoding by Linear Programming, IEEE Inf. Theory 51, 4203–4215, 2005.

[142] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies. IEEE Trans. Inform. Theory, 52, 5406–5425, 2006.

[143] E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. 52(2):489–509, 2006.

[144] D.L. Donoho. Compressed sensing. IEEE Trans. Inform. Theory, 52:1289–1306, 2006.

[145] S. Mendelson, A. Pajor, N. Tomczak-Jaegermann, Reconstruction and subgaussian operators in Asymptotic Geometric Analysis, Geometric and Functional Analysis, 17(4), 1248–1282, 2007.

[146] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation based image restoration. Multiscale Model. Simul, 4(2):460–489, 2005.

[147] E. Hale, W. Yin, and Y. Zhang. A fixed-point continuation method for ℓ_1-regularization with application to compressed sensing. CAAM Technical Report TR07-07, Rice University, Houston, TX, 2007.

[148] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. Phys. D, 60:259–268, 1992.

[149] T-C. Chang, L. He, and T. Fang. Mr image reconstruction from sparse radial samples using bregman iteration. Proceedings of the 13th Annual Meeting of ISMRM, 2006.

[150] Y. Li, S. Osher, and Y.-H. Tsai. Recovery of sparse noisy date from solutions to the heat equation. in preparation.

[151] L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. USSR Computational Mathematics and Mathematical Physics, 7(3):200–217, 1967.

[152] D.L. Donoho. De-noising by soft-thresholding. IEEE Trans. Inform. Theory.

[153] W. Yin. On the linearized bregman algorithm. private communication.

[154] M. Bachmayr. Iterative total variation methods for nonlinear inverse problems. Master's thesis, Johannes Kepler Universität, Linz, Austria, 2007.

[155] I. Ekeland and R. Témam, Convex Analysis and Variational Problems, Classics in Applied Mathematics, SIAM, Philadelphia, 1999.

[156] J. Cai, E. Candès and Z. Shen, A singular value thresholding algorithm for matrix completion.

[157] P. Blomgren, T. F. Chan, P. Mulet, L. Vese and W. L. Wan. Variational PDE models and methods for image processing, in: Research Notes in Mathematics, 420 (2000), 43–67, Chapman and hall/CRC.

[158] A. Chambolle, Total Variation minimization and a class of binary MRF models, In Springer-Verlag, (ed.), 5th International Workshop on En-

ergy Minimization Methods in Computer Vision and Pattern Recognition(EMMCVPR), Vol. LNCS 3757, 136–152, 2005.

[159] T. F. Chan, G. H. Golub and P. Mulet. A nonlinear primal dual method for total variation based image restoration, SIAM J. Sci. Comput. 20, 1964–1977, 1999.

[160] T. F. Chan, H. M. Zhou and R. H. Chan, Continuation Method for Total Variation Denoising Problems, UCLA CAM Report 95-28, 1995.

[161] J. Darbon and M. Sigelle. Exact optimization of discrete constrained total variation minimization problems. In: R. Klette and J. Zunic, editors, Tenth International Workshop on Combinatorial Image Analysis, volume 3322 of LNCS, 548–557, 2004.

[162] J. Darbon and M. Sigelle. A fast and exact algorithm for total variation minimization. In: J. S. Marques, N. Prez de la Blanca, and P. Pina, editors, 2nd Iberian Conference on Pattern Recognition and Image Analysis, volume 3522 of LNCS 351–359, 2005.

[163] D. Goldfarb and W. Yin. Second-order cone programming methods for total variation-based image restoration, SIAM J. Sci. Comput. 27, 622–645, 2005.

[164] D. Goldfarb and W. Yin. Parametric Maximum Flow Algorithmsfor Fast Total Variation Minimization. Rice CAAM Report TR 07-09.

[165] B. Zalesky, Network Flow Optimization for Restoration of Images. Journal of Applied Mathematics, Vol. 2, No. 4, 199–218, 2002.

[166] M. Zhu and T. Chan, An Efficient Primal-Dual Hybrid Gradient Algorithm for Total Variation Image Restoration, CAM-Report 08-34, May 2008.

[167] M. Zhu, S. J. Wright and T. Chan, Duality-Based Algorithms for Total Variation Image Restoration, CAM-Report 08-33, May 2008.

[168] T. Chan and S. Esedoglu. Aspects of total variation regularized L_1 function approximation. SIAM Journal on Applied Mathematics, 65:5, 1817–1837, 2005.

[169] X. Bresson, S. Esedoglu, P. Vandergheynst, J. Thiran and S. Osher, Fast Global Minimization of the Active Contour/Snake Model, Journal of Mathematical Imaging and Vision, 2007.

[170] Tom Goldstein and Stanley Osher, The Split Bregman Algorithm for L1 Regularized Problems, CAM-Report 08-29, April 2008.

[171] M. Kass, A. Witkin and D. Terzopoulos, Snakes: Active Contour Models, International Journal of Computer Vision, 321–331, 1987.

[172] V. Caselles, R. Kimmel and G. Sapiro, Geodesic Active Contours, International Journal of Computer Vision, Vol. 22(1), 61–79, 1997.

[173] G. Sapiro, Geometric Partial Differential Equations and Image Analysis, Chambridge U. Press, 2001.

[174] S. Osher and R. Fedkiw, Level Set Methods and Dynamic Implicit Surfaces. Springer-Verlag, New York, 2003.

[175] T. Chan and J. Shen, Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods. Society for Industrial and Applied Mathematics (SIAM), 2005.

[176] T. Chan and L. Vese, Active Contours Without Edges, IEEE Transactions on Image Processing, Vol. 10(2), 266–277, 2001.

[177] T. F. Chan, S. Esedoglu and M. Nikolova, Algorithms for finding global minimizers of denoising and segmentation models, SIAM J. Appl. Math. Vol. 66, 1632–1648, 2006.