VESSEL SEGMENTATION IN MEDICAL IMAGING USING A TIGHT-FRAME BASED ALGORITHM

XIAOHAO CAI*, RAYMOND CHAN†, SERENA MORIGI‡, AND FIORELLA SGALLARI§

Abstract. Tight-frame, a generalization of orthogonal wavelets, has been used successfully in various problems in image processing, including inpainting, impulse noise removal, super-resolution image restoration, etc. Segmentation is the process of identifying object outlines within images. There are quite a few efficient algorithms for segmentation such as the model based approaches, pattern recognition techniques, tracking-based approaches, artificial intelligence-based approaches, etc. In this paper, we propose to apply the tight-frame approach to automatically identify tube-like structures in medical imaging, with the primary application of segmenting blood vessels in magnetic resonance angiography images. Our method iteratively refines a region that encloses the potential boundary of the vessels. At each iteration, we apply the tight-frame algorithm to denoise and smooth the potential boundary and sharpen the region. The cost per iteration is proportional to the number of pixels in the image. We prove that the iteration converges in a finite number of steps to a binary image whereby the segmentation of the vessels can be done straightforwardly. Numerical experiments on synthetic and real 2D/3D images demonstrate that our method is more accuracy when compared with some representative segmentation methods, and it usually converges within a few iterations.

Key words. Tight-frame, wavelet transform, automatic image segmentation, medical imaging.

1. Introduction. The segmentation problem of branching tubular objects in 2D and 3D images arises in many applications, for examples, extracting roads in aerial photography, and anatomical surfaces of blood vessels in medical images. In this paper, we are concerned with identifying tube-like structures in medical imaging, with the primary application of segmenting blood vessels in magnetic resonance angiography (MRA) images. Unlike classical segmentation problems, vessel segmentation is characterized by different aims such as (a) detect correctly branches and complex topologies, (b) detect vessels of very different thickness (from very thin to very thick), (c) repair small occlusions (false disconnections), (d) remove noise incorrectly segmented, and (e) control the minimum thickness of the vessels by a user given precision. Moreover, when used in a real-time medical environment, automatic, robust and efficient methods are essential. All these requirements make the vessel segmentation problem very challenging.

Many different approaches for image segmentation and, in particular, vessel segmentation have been proposed in literature, see for example [14, 15, 16, 21, 24, 29, 34, 35, 42, 44, 46] and the extended reviews [17, 33]. Below we give a brief account of some of these methods.

Deformable model approaches, initially introduced for general image segmentation, have been applied to vessel segmentation as well [16, 24, 35, 44, 46]. They start from an initial boundary estimate, and deform it iteratively so as to minimize a functional depending on structures and regularity of the surface. Explicit deformable models such as those discussed in [36] have some drawbacks like inaccurate calculation

* Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong. Email: xhcai@math.cuhk.edu.hk.

† Department of Mathematics, The Chinese University of Hong Kong, Shatin, Hong Kong. Email: rchan@math.cuhk.edu.hk. This work is partially supported by RGC 400510 and DAG 2060408.

‡ Department of Mathematics-CIRAM, University of Bologna, Bologna, Italy. Email: morigi@dm.unibo.it.

§ Department of Mathematics-CIRAM, University of Bologna, Bologna, Italy. Email: sgallari@dm.unibo.it.

of the normals and curvature because of the discrete nature of the contour parameterization; costly computation to prevent self-intersections in the resulting contours, etc. Therefore the level set technique which provides implicit representation of the deformable model was introduced. Interested readers can refer to recent literature on the level set segmentation strategy for tubular structures [29, 30, 38, 42]. However, the level set segmentation approach is also computationally expensive as it needs to cover the entire domain of interest, which is one dimension higher than the original one.

A very successful deformable method for vessel segmentation in volume datasets was proposed in [35]. The method is an extension of geodesic active contour and minimal surfaces, with the distinction that its regularizing force is derived from an underlying one-dimensional curve in three dimensions. The curve can be considered intuitively as the center line of the tubular structures. This method has very good performance and the software is public available. However, from our experiments, we noticed that it still can not detect thin vessels in MRA images especially when their intensities are close to the intensity of the background, see numerical results in Section 4.

In [24], a geometric deformable model for the segmentation of tubular-like structures was proposed. It continuously deforms an initial distance function following the Partial Differential Equation (PDE)-based diffusion model derived from a minimal volume-like variational formulation. It uses a suitable diffusion tensor to control the directionality of the tubular structures, and hence it possesses the ability to segment twisted, convoluted and occluded structures. The authors in [24] have also applied a variant of their PDE model to the challenging problem of composed segmentation in [25]. However, their methods do not perform well for MRA images where there are weak edges, and they are also sensitive to noise and blur in the images.

Besides the methods above, new approaches based on wavelets and tight-frames have been proposed to classify texture and segmentation [1, 43]. The tight-frame approach is a versatile and effective tool for many different applications in image processing, see [2, 5, 12, 11, 40, 41, 45]. As proven in [5, 6], the approach is closely related to variational approach. There are many kinds of tight-frame systems, such as those from framelets [37], contourlets [20] and curvelets [9, 10], etc. Recently, the authors in [21] proposed a minimization model which replaces the total variation norm in model [13] by ℓ_1 norm of framelets coefficients. Notice that model [13] is the convex version of the Chan-Vese active contour model [14]. The Chan-Vese model has the minimal partition property, i.e. it assumes that the segmentation result is a piecewise constant image with two different constant values, and its partition effectiveness can be accomplished after it is minimized. Therefore the tight-frame model in [21] has the advantage of multi-resolution analysis of tight-frame systems, yet it also shares the minimal partition property of the Chan-Vese model. It is shown in [21] that it can segment complex structures in medical images very well. But from our experiments in Section 4, we noticed that it also can not detect thin vessels well especially for those with low intensities.

In this paper, we derive a segmentation algorithm that also uses tight-frames. However, instead of modeling the problem as a minimization problem with a data-fitting term and a fidelity term as in [21], we use an iterative procedure that gradually updates an interval that contains pixel values of potential boundary pixels. The major advantage of this technique is the ability to segment twisted, convoluted and occluded structures without user interactions; and it can follow the branching of different layers,

from thinner to larger structures. Moreover, we can prove the convergence of our method and it is robust in case of noisy and blurred images.

Comparison with methods in [14, 21, 24, 25, 35] on synthetic and real 2D and 3D MRA images show that our method gives more accurate vessel segmentation. On the synthetic images, see Figures 4.1 and 4.2, our method basically can recover all the branches in the images while other methods do not perform as well. Moreover, for all the images we tested, our methods converges within 10 iterations, where the cost for each iteration is proportional to the number of pixels in the image. (In principle, it can be further reduced to proportional to the number of pixels on the boundary of the vessels though we did not pursue this in this paper.) This is an advantage when doing real-time segmentation.

A preliminary version of our segmentation algorithm has been given in an SSVM proceedings [8]. The main contributions in this paper are: (1) a simpler strategy to initialize and refine the regions enclosing the potential boundary of the vessels, (2) the new strategy leads to a simple proof of convergence and easier choice of parameters, (3) a different tight-frame with better directionally selective property is used to obtain more details in the image, (4) new synthetic, 2D, and 3D tests are added, also noise in the data are considered.

The rest of the paper is organized as follows. In Section 2, we recall some basic facts about tight-frame and generic tight-frame algorithms. Our segmentation algorithm is given in Section 3. In Section 4 we test our algorithm on various synthetic and real 2D/3D images and we provide the comparison with some representative algorithms from different segmentation approaches: active contour models [14], tight-frame [21], CURVES vessel segmentation method [35], and anisotropic deformable approach [24, 25]. Conclusions are given in Section 5.

2. Tight-Frame Algorithm. In this section, we briefly introduce the tight-frame algorithms used in [2, 3, 4, 5, 11] which are based on tight-frame transforms. All tight-frame transforms $\mathcal{A}$ have a very important property, the "*perfect reconstruction property*": $\mathcal{A}^T\mathcal{A} = \mathcal{I}$, the identity transform, see [37]. Unlike the wavelets, in general, $\mathcal{A}\mathcal{A}^T \neq \mathcal{I}$. For theories of framelets and tight-frame transforms, we refer the readers to [18, 5, 6] for more details. In order to apply the tight-frame algorithm, one only needs to know the filters corresponding to the framelets in the tight-frame. In the followings, we give two examples of tight-frames: the piecewise linear B-spline tight-frame [19] and the dual-tree complex wavelet tight-frame [39].

The filters in the piecewise linear B-spline tight-frame are:

$$h_0 = \frac{1}{4}[1,\ 2,\ 1], \quad h_1 = \frac{\sqrt{2}}{4}[1,\ 0,\ -1], \quad h_2 = \frac{1}{4}[-1,\ 2,\ -1], \tag{2.1}$$

see [37]. The tight-frame coefficients of any given vector $\mathbf{v}$ corresponding to filter h_i can be obtained by convolving h_i with $\mathbf{v}$. In matrix terms, we can construct, for each filter, its corresponding filter matrix which is just the circulant matrix with diagonals given by the filter coefficients, e.g. $H_0 = \frac{1}{4}\text{tridiag}[1,\ 2,\ 1]$. Then the 1D tight-frame forward transform is given by

$$\mathcal{A} = \begin{bmatrix} H_0 \\ H_1 \\ H_2 \end{bmatrix}. \tag{2.2}$$

To apply the tight-frame transform onto $\mathbf{v}$ is equivalent to computing $\mathcal{A}\mathbf{v}$, and $H_i\mathbf{v}$ gives the tight-frame coefficients corresponding to the filter h_i, $i = 1, 2, 3$.

The d-dimensional piecewise linear B-spline tight-frame is constructed by tensor products from the 1D tight-frame above, see [22]. For example, in 2D, there are nine filters given by $h_{ij} \equiv h_i^T \otimes h_j$ for $i, j = 1, 2, 3$, where h_i is given in (2.1). For any 2D image f, the tight-frame coefficients with respect to h_{ij} are obtained by convolving h_{ij} with f. The corresponding forward transform $\mathcal{A}$ will be a stack of nine block-circulant-circulant-block matrices (cf. (2.2)). The tight-frame coefficients are given by the matrix-vector product $\mathcal{A}\mathbf{f}$, where $\mathbf{f} = \text{vec}(f)$ denotes the vector obtained by concatenating the columns of f.

Dual-tree complex wavelet transform (D$\mathbb{C}$WT) was firstly introduced by Kingsbury [31, 32]. Apart from having the usual perfect reconstruction property, shift-invariance property and linear complexity, it also has the nice directionally selective property at $\pm 15^\circ, \pm 45^\circ, \pm 75^\circ$. The idea is to use two different sets of filters: one gives the real part of the transform and the other gives the imaginary part. Let $\{g_0, g_1\}$ and $\{h_0, h_1\}$ denote the two different sets of orthonormal filters, where g_0 and h_0 are the low pass filters, and g_1 and h_1 are the high pass filters. Let the square matrix $A_{g_i h_j}$ denote the 2D separable wavelet transform implemented using g_i along the rows and h_j along the columns, and define $A_{h_i h_j}, A_{g_i g_j}, A_{h_i g_j}$ similarly, where $i, j = 1, 2$. Then the forward transform for the 2-dimensional D$\mathbb{C}$WT is represented by

$$\mathcal{A}\mathbf{f} := \frac{1}{\sqrt{8}} \begin{bmatrix} I & -I & 0 & 0 \\ I & I & 0 & 0 \\ 0 & 0 & I & I \\ 0 & 0 & I & -I \end{bmatrix} \begin{bmatrix} A_{g_i g_j} \\ A_{h_i h_j} \\ A_{h_i g_j} \\ A_{g_i h_j} \end{bmatrix} \mathbf{f}, \quad i, j = 1, 2.$$

We see that the 2-dimensional D$\mathbb{C}$WT requires four different wavelet transforms in parallel. We refer the readers to [7, 18, 27, 39] and the references therein for more details and for the implementation of 3-dimensional D$\mathbb{C}$WT. In practice, the D$\mathbb{C}$WT are implemented by using one set of filters for the first level and another set of filters for the remaining levels, see [39]. The filters and the Matlab code for D$\mathbb{C}$WT can be obtained from [7].

The tight-frame algorithms used in [2, 3, 4, 5, 11] can be presented in the following generic form:

$$\mathbf{f}^{(i+\frac{1}{2})} = \mathcal{U}(\mathbf{f}^{(i)}), \tag{2.3}$$

$$\mathbf{f}^{(i+1)} = \mathcal{A}^T \mathcal{T}_\lambda(\mathcal{A}\mathbf{f}^{(i+\frac{1}{2})}), \quad i = 1, 2, \ldots. \tag{2.4}$$

Here $\mathbf{f}^{(i)}$ is an approximate solution at the i-th iteration, $\mathcal{U}$ is a problem-dependent operator, and $\mathcal{T}_\lambda(\cdot)$ is the soft-thresholding operator defined as follows. Given vectors $\mathbf{v} = [v_1, \cdots, v_n]^T$ and $\lambda = [\lambda_1, \cdots, \lambda_n]^T$, $\mathcal{T}_\lambda(\mathbf{v}) \equiv [t_{\lambda_1}(v_1), \cdots, t_{\lambda_n}(v_n)]^T$, where

$$t_{\lambda_k}(v_k) \equiv \begin{cases} \text{sgn}(v_k)(|v_k| - \lambda_k), & \text{if } |v_k| > \lambda_k, \\ 0, & \text{if } |v_k| \le \lambda_k. \end{cases} \tag{2.5}$$

We refer to [23] for possible choices of λ_k.

We remark that (2.4) realizes a tight-frame denoising and smoothing on the image while (2.3) performs a data-fitting according to the specific problem at hand. For high-resolution image reconstruction problem [11], astronomical infra-red imaging [3], impulse noise removal [4], and inpainting problem [2], the tight-frame algorithm (2.3)–(2.4) has been shown to be convergent to a functional with the regularization term being the 1-norm of the tight-frame coefficients, see [5, 22].

3. Tight-Frame Based Algorithm for Segmentation. The technology of MRA imaging is based on detection of signals from flowing blood and suppression of signals from other static tissues, so that the blood vessels appear as high intensity regions in the image. The structures to be segmented in MRA images are vessels of variable diameters which can be very close to each other. Speckle noise and weak edges in the images make the detection of the structures an especially difficult task. One intuitive way to segment the vessels is to use a 2-phase segmentation, i.e. transform the image into a binary image of 0 and 1. Then the boundaries of the vessels are just the boundary between pixel values 0 and 1. Because of the presence of noise and weak edges, the naive way of simply thresholding the given image into 0 and 1 will not be able to recover very small vessels or to repair occlusions in vessels which may not appear connected in the volumetric image. Even sophisticated 2-phase segmentation methods for general images such as the popular Chan-Vese model [14] will not work for vessel segmentation, see the results in Section 4. Fortunately, these MRA images contain some properties that we can exploit to derive a good 2-phase segmentation algorithm. Namely, pixels near the vessel boundary have values in some gray-level range, whereas pixels in the background are completely outside of this range, see Figure 3.1(a). Thus the main idea behind our algorithm is to approximate this range accurately.

We obtain the range iteratively by a tight-frame algorithm which has the same generic form as in (2.3) and (2.4). For our algorithm, the operator $\mathcal{U}$ in (2.3) signifies the process of making $\mathbf{f}^{(i)}$ more binary in each iteration. The main steps for each iteration are as follows. Suppose in the beginning of the ith iteration, we are given an approximate image $f^{(i)}$ and a set $\Lambda^{(i)}$ that contains all potential boundary pixels, i.e. pixels that are likely to be on the boundary. Then (1) we use $\Lambda^{(i)}$ to estimate an appropriate range $[\alpha_i, \beta_i]$ that contains the pixel values of potential boundary pixels; (2) we use the range to separate the image into three parts—those below the range are classified as background pixels, those inside the range will form the new set $\Lambda^{(i+1)}$ of potential boundary pixels, and those above the range are classified as pixels in the vessels; (3) we denoise and smooth the image on $\Lambda^{(i+1)}$ by the tight-frame algorithm to get a new image $f^{(i+1)}$. The algorithm is stopped when the image becomes binary.

We will prove that our algorithm converges in a finite number of steps. From our experiments, this happens within 10 iterations for the synthetic as well as real 2D/3D MRA images we tested, see Tables 4.1 and 4.2 in Section 4. In the following, we discuss in more details each of the steps. Without loss of generality, we assume that the given image f assumes values in [0, 1].

Initialization. To start the algorithm at $i = 0$, we need to define $f^{(0)}$, the initial guess, and $\Lambda^{(0)}$, the initial set of potential boundary pixels. We set $f^{(0)} = f$, the given image. For $\Lambda^{(0)}$, since we do not have any knowledge of where the boundary pixels are located when the process starts, we identify them by using the gradient of f, i.e. we locate them as pixels where the gradient is larger than a given threshold ϵ. More precisely, let Ω be the index set of all the pixels in the image, then we define

$$\Lambda^{(0)} \equiv \{j \in \Omega \mid \|[\nabla f]_j\|_1 \geq \epsilon\}. \tag{3.1}$$

Here $[\nabla f]_j$ is the discrete gradient of f at the jth pixel. Once $f^{(0)}$ and $\Lambda^{(0)}$ are defined, we start the iterative process. Let us describe the ith iteration in details below.

Step (1): computing the range $[\alpha_i, \beta_i]$. Given $\Lambda^{(i)}$, we first compute the mean

pixel value on $\Lambda^{(i)}$:

$$\mu^{(i)} = \frac{1}{|\Lambda^{(i)}|} \sum_{j \in \Lambda^{(i)}} f_j^{(i)}, \tag{3.2}$$

where $|\cdot|$ denotes the cardinality of the set and $f_j^{(i)}$ is the pixel value of pixel j in image $f^{(i)}$. Then we compute the mean pixel values of the two sets separated by $\mu^{(i)}$:

$$\mu_-^{(i)} = \frac{1}{|\{j \in \Lambda^{(i)} : f_j^{(i)} \leq \mu^{(i)}\}|} \sum_{\{j \in \Lambda^{(i)} : f_j^{(i)} \leq \mu^{(i)}\}} f_j^{(i)}, \tag{3.3}$$

and

$$\mu_+^{(i)} = \frac{1}{|\{j \in \Lambda^{(i)} : f_j^{(i)} \geq \mu^{(i)}\}|} \sum_{\{j \in \Lambda^{(i)} : f_j^{(i)} \geq \mu^{(i)}\}} f_j^{(i)}. \tag{3.4}$$

While $\mu^{(i)}$ reflects the mean energy of the set of potential boundary pixels, $\mu_-^{(i)}$ and $\mu_+^{(i)}$ reflect the mean energies of the pixels on the boundary closer to the background and closer to the vessels respectively. We define

$$\alpha_i \equiv \max\left\{\frac{\mu^{(i)} + \mu_-^{(i)}}{2}, 0\right\}, \quad \beta_i \equiv \min\left\{\frac{\mu^{(i)} + \mu_+^{(i)}}{2}, 1\right\}. \tag{3.5}$$

Step (2): thresholding the image into three parts. Using the range $[\alpha_i, \beta_i] \subseteq [0, 1]$, we can separate the image $f^{(i)}$ into three parts—those below, inside, and above the range, see Fig. 3.1(b) for $i = 0$. Since our aim is to create a binary image, we threshold those pixel values that are smaller than α_i to 0, those larger than β_i to 1, and those in between, we stretch them between 0 and 1 using a simple linear contrast stretch, see [28]. If there are no pixels in between α_i and β_i, then the threshold image is binary and the algorithm stops. More precisely, let

$$\begin{aligned} M_i &= \max\{f_j^{(i)} \mid \alpha_i \leq f_j^{(i)} \leq \beta_i, j \in \Lambda^{(i)}\}, \\ m_i &= \min\{f_j^{(i)} \mid \alpha_i \leq f_j^{(i)} \leq \beta_i, j \in \Lambda^{(i)}\}, \end{aligned} \tag{3.6}$$

then we linear stretch the contrast of $f_j^{(i)}$ to (0,1):

$$f_j^{(i+\frac{1}{2})} = \begin{cases} 0, & \text{if } f_j^{(i)} \leq \alpha_i, \\ \frac{f_j^{(i)} - m_i}{M_i - m_i}, & \alpha_i \leq f_j^{(i)} \leq \beta_i, \\ 1, & \text{if } \beta_i \leq f_j^{(i)}, \end{cases} \qquad \text{for all } j \in \Omega. \tag{3.7}$$

Fig. 3.1(c) shows the threshold and stretched image from Fig. 3.1(b), where the yellow pixels are pixels we have classified not on the boundary, i.e. $f_j^{(i+\frac{1}{2})} = 0$ (signifying pixel j is in the background) or $f_j^{(i+\frac{1}{2})} = 1$ (signifying pixel j is inside the vessel). The set of the remaining pixels that have to be classified is denoted by:

$$\Lambda^{(i+1)} = \{j \mid 0 < f_j^{(i+\frac{1}{2})} < 1, j \in \Omega\}. \tag{3.8}$$

(a) Given MRA image (b) Thresholded image (c) Result from Step 2

FIG. 3.1. *(a) Given MRA image. (b) thresholding the image into three parts: green–below, red–in between, and yellow–above the range* $[\alpha, \beta]$*. (c) Result from step (2): yellow pixels have value either 0 or 1.*

Note that pixels with values m_i and M_i are mapped to 0 and 1 respectively and hence they are not in $\Lambda^{(i+1)}$. Next, we denoise and smooth $f^{(i+\frac{1}{2})}$ on $\Lambda^{(i+1)}$.

Step (3): tight-frame iteration. To denoise and smooth the image $f^{(i+\frac{1}{2})}$ on $\Lambda^{(i+1)}$, we apply the tight-frame iteration (2.4) on $\Lambda^{(i+1)}$. More precisely, if $j \notin \Lambda^{(i+1)}$, then we set $f_j^{(i+1)} = f_j^{(i+\frac{1}{2})}$; otherwise, we use (2.4) to get $f_j^{(i+1)}$. To write it out clearly, let $\mathbf{f}^{(i+\frac{1}{2})} = \text{vec}(f^{(i+\frac{1}{2})})$, and $P^{(i+1)}$ be the diagonal matrix where the diagonal entry is 1 if the corresponding index is in $\Lambda^{(i+1)}$, and 0 otherwise. Then

$$\mathbf{f}^{(i+1)} \equiv (I - P^{(i+1)})\mathbf{f}^{(i+\frac{1}{2})} + P^{(i+1)}\mathcal{A}^T\mathcal{T}_\lambda(\mathcal{A}\mathbf{f}^{(i+\frac{1}{2})}). \tag{3.9}$$

By reordering the entries of the vector $\mathbf{f}^{(i+1)}$ into columns, we obtain the image $f^{(i+1)}$. We remark that the effect of (3.9) is to denoise and smooth the image on $\Lambda^{(i+1)}$, see [5]. Note that the main cost of (3.9) is the forward and backward tight-frame transforms and hence the cost is proportional to the number of pixels. Since the values of all pixels outside $\Lambda^{(i+1)}$ are either 0 or 1, the cost can be reduced significantly by applying the transforms on pixels around $\Lambda^{(i+1)}$ only.

Stopping criterion. The iteration is terminated as soon as all the pixels of $f^{(i+\frac{1}{2})}$ are either of value 0 or 1, or equivalently when $\Lambda^{(i)} = \emptyset$. Then for the binary image $f^{(i+\frac{1}{2})}$, all the pixels with value 0 are considered as background pixels and pixels with value 1 constitute the tubular structures.

Below we summarize the steps required to segment an image f by our tight-frame method. We refer to our method as TFA.

Tight-Frame Algorithm (TFA)

1. Input: given image f.
2. Set $f^{(0)} = f$ and $\Lambda^{(0)}$ by (3.1)
3. Do $i = 0, 1, \ldots,$ until convergence
   (a) Compute $[\alpha_i, \beta_i]$ by (3.5).
   (b) Compute $f^{(i+\frac{1}{2})}$ by (3.7).
   (c) Stop if $f^{(i+\frac{1}{2})}$ is a binary image.
   (d) Compute $\Lambda^{(i+1)}$ by (3.8).
   (e) Update $f^{(i+\frac{1}{2})}$ to $f^{(i+1)}$ by (3.9).
4. Output: binary image $f^{(i+\frac{1}{2})}$.

Our TFA always converges to a binary image, as is shown in the following result.

THEOREM 3.1. *TFA converges to a binary image within a finite number of steps.*

Proof. From (3.8), we need to prove that $|\Lambda^{(i)}| = 0$ at some finite step $i > 0$. By (3.6), if $f^{(i+\frac{1}{2})}$ is not yet a binary image, then there will be at least one $j \in \Lambda^{(i)}$ such that $f_j^{(i)} = M_i$. By (3.7), $f_j^{(i+\frac{1}{2})}$ will be set to 1 and hence by (3.8), $j \notin \Lambda^{(i+1)}$. Hence $|\Lambda^{(i+1)}| < |\Lambda^{(i)}|$. Since $|\Lambda^{(0)}|$ is finite, there must exist some i such that $|\Lambda^{(i)}| = 0$. ∎

We emphasize that the algorithm numerically converges within 10 iterations for all the synthetic and real 2D/3D images we have tested, see Section 4.

Finally, let us estimate the computation cost of our method for a given image with n pixels. Since the costs of computing $\mu^{(i)}$, $\mu_-^{(i)}$, $\mu_+^{(i)}$, and $[\alpha_i, \beta_i]$ are all of $O(n)$ operations, see (3.2)–(3.5), and the cost of a tight-frame transform is also linear in n (see e.g. (2.2) where H_i are all tri-diagonal matrices), the complexity of our algorithm is $O(n)$ per iteration. Moreover, the computation can be further speed up by applying the algorithm only to pixels around $\Lambda^{(i)}$ instead of applying them on the entire image domain Ω. To understand how much we can gain by restricting the domain, we report in Tables 4.1 and 4.2, the cardinality of $\Lambda^{(i)}$ after each iteration of TFA. We see that after just 3 iterations, $\Lambda^{(3)}$ only has 0.3% (respectively 0.04%) of the number of pixels in Ω for 2D (respectively 3D) images. That is, more than 99% (respectively 99.9%) of the pixels in Ω are classified either in or outside of the boundary just after 3 iterations.

Since we have not optimized the code yet, in the numerical tests we carried out the computations in Ω instead of in a neighborhood of $\Lambda^{(i)}$.

4. Numerical Examples. In this section, we test our tight-frame segmentation algorithm (TFA) on seven different synthetic and real medical images. The thresholding parameters λ_k used in (2.5) are all chosen to be $\lambda_k \equiv 0.1$. The tight-frame used is the D$\mathbb{C}$WT downloaded from [7] where all parameters in the code are chosen to be the default values. In particular, the number of wavelet levels used is 4.

We compare our method with the following methods from different approaches: Chan-Vese active contour model [14] (CV), mixed tight-frame and Chan-Vese approach [21] (CVTF), extended geodesic active contour method [35] (CURVES), and anisotropic deformable approach [24, 25] (ADA). CV, though designed for generic image segmentation, is included here as it is one of the most popular segmentation models. We will like to show that vessel segmentation is a difficult problem that successful generic segmentation algorithms do not work well for it. CVTF is a tight-frame method that is based on the Chan-Vese active contour approach but using the tight-frame coefficients as the regularization term. CURVES follows from geodesic active contour approach and ADA is a geometric deformable model; and they are both specifically designed for vessel segmentation. All the results in the tests were obtained by tuning the respective parameters to the best. We remark that CURVES is designed only for 3D segmentation. The programs for CV, CVTF, CURVES and ADA are either provided by the authors or publicly available. The 2D images are tested in a MacBook with 2.4 GHz processor and 4GB RAM, while the 3D images, because of their sizes, are tested on a node with 120GB RAM in a PC-cluster. All computations are carried out in MATLAB except the CURVES which is computed in the software ITK. The curve boundaries and the iso-surfaces of the results of all methods are extracted and visualized by the Matlab commands "`contour`" and "`isosurface`" for 2D and 3D images respectively.

We will start with two synthetic images where we have the ground truths that can be used to quantitatively compare the different methods. For the real 2D/3D medical

images, we can only compare the results qualitatively, as there are no ground truth segmentations for those images. Even manually segmentation could not be considered as ground truth since many thin vessels are hard to detect and many faulty detections can be included due to the high noise presented in the data set.

4.1. Synthetic vessel segmentation. Here we tested on two synthetic images.

Example 1: Annuluses and stripes segmentation. The ground truth 256×256 image, given in Fig. 4.1(a), is composed of four annuluses and four straight stripes with widths ranging from 2 to 21 pixels. To simulate blood flowing in the vessels, which has higher intensity near the center line of the vessels and less towards the boundary, we generated the synthetic clean image as follows. We assign different values to the center line and then gradually decreases the values when moving towards the boundary. The resulting clean image is shown in Fig. 4.1(b). The corrupted image, given in Fig. 4.1(c), is obtained by applying Gaussian blur and additive Gaussian noise to the clean image. The Gaussian blur filter is represented by a 3×3 window with standard deviation 0.5, and the Gaussian noise used has mean 0.01 and variance 0.001. The small intensities at the thin structures and the weak boundaries at the thick structures, together with the added noise and blur, will make the segmentation problem difficult.

To evaluate the performance of the different methods applied to this synthetic example, we provide in the second row of Fig. 4.1 the resulting images, and in the third row of Fig. 4.1 their difference from the ground truth image. The four columns there represent respectively results from CV, CVTF, ADA, and our TFA. For our TFA, we used $\epsilon = 0.003$ in (3.1). The number of the wrongly-detected pixels are given there too. They are $4402, 4571, 1072$ and 860, respectively. From the numbers and also the figures, we see that our TFA is the best method. In particular, almost all the pixels on the thin and the thick structures are detected correctly except for a few pixels located on the boundaries. We notice that the pixels on the outermost annulus and on the thinnest stripe are detected only by ADA and TFA algorithms, and only our method give smooth boundaries. This is because of the built-in denoising property of tight-frames in formula (2.4) which regularizes and smoothes the boundaries. We highlight the differences between ADA and TFA by the rectangular boxes in Fig. 4.1(j) and (k). Our method converges in 7 iterations and requires 0.98 seconds. The first column of Table 4.1 gives the numbers of possible boundary pixels $|\Lambda^{(i)}|$ at each iteration of TFA. We see that just after 2 iterations more than 99% of the pixels are classified already either on the vessels or outside the vessels.

Example 2: Tree-like segmentation. The ground truth 512×512 image, given in Fig. 4.2(a), is composed by synthetic vessels with different widths. The synthetic clean image and the corrupted image, shown in Fig. 4.2(b) and (c), are generated similarly as in Example 1. Again the second row of Fig. 4.2 gives the resulting images from the methods, and the third row gives their differences from the ground truth image. The number of wrongly-detected pixels are $2785, 7038, 2834$ and 402, respectively. We used $\epsilon = 0.3$ in (3.1) for our TFA, and it converges in 6 iterations and requires 3.68 seconds. Clearly from the numbers of wrongly-detected pixels and also from the figures, we see that TFA gives the best result. In particular, almost all the pixels on the thin and the thick structures are detected correctly by our method except for a few pixels located on the boundaries.

4.2. 2D vessel segmentation. Here we tested on two 2D real images.

Example 3: Carotid vascular system segmentation. The test image is a 182×182 MRA image of a carotid vascular system, see Fig. 4.3(a). The blood vessels contain

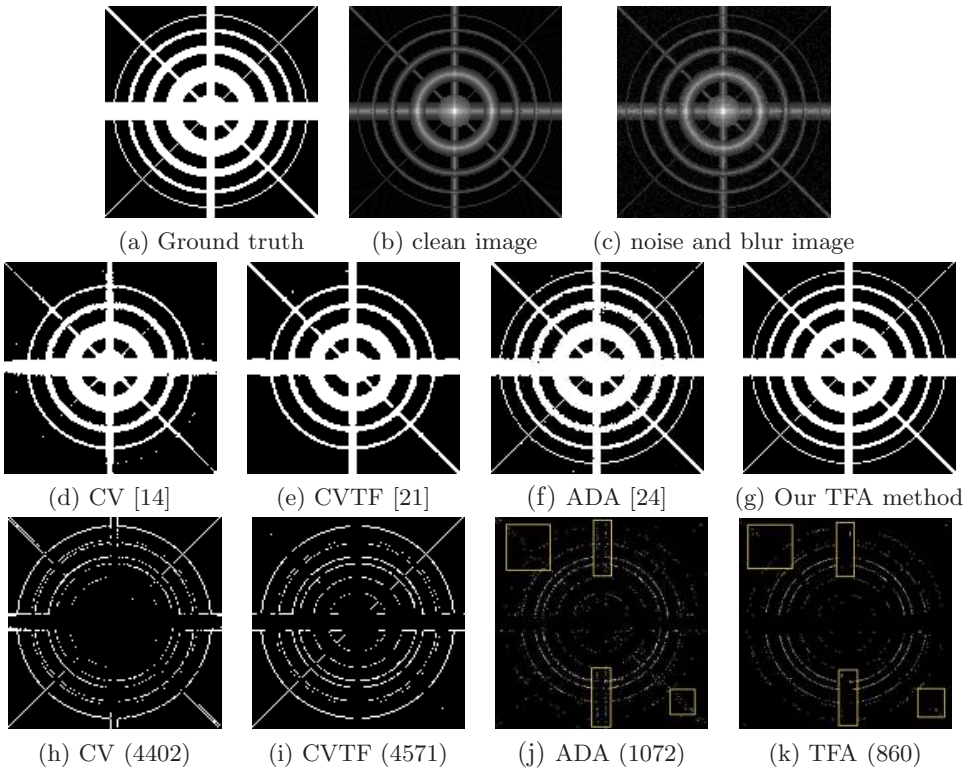
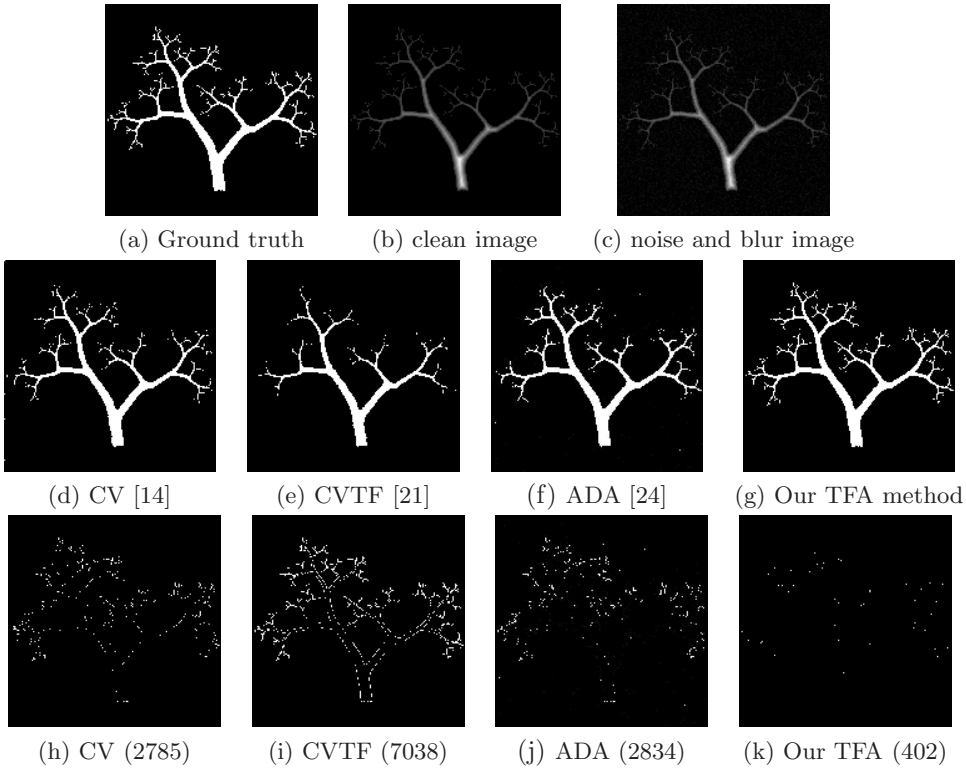


FIG. 4.1. *Example 1. (a) Ground truth image; (b) Clean image; (c) Noisy and blurred image; (d)–(g) Results of methods CV [14], CVTF [21], ADA [24] and our TFA method, respectively; (h)–(k) Differences between the ground truth image (a) and the results of methods. Numbers in braces are the numbers of wrongly-detected pixels.*

regions with high and low intensities, including some very thin vessels in the middle with intensities as low as the intensity of the background. Intersections of partial structures even increase the difficulty of the segmentation. The segmentation results by CV, CVTF, ADA and TFA are shown in Fig. 4.3(b)–(e) respectively, where the given image is overlaid so that we can compare the accuracy of the methods. The TFA uses $\epsilon = 0.003$ in (3.1) and converges in 5 iterations with a computational time of 0.64 seconds.

Similar to the conclusion we have in Section 4.1, the results by CV and CVTF are not satisfactory since the vessels obtained by CV are disconnected and CVTF cannot detect a large part of the vessels. By comparing the parts inside the rectangular boxes in Fig. 4.3(d) with those in Fig. 4.3(e), we notice that our method is able to extract smoother boundaries than ADA method and, in particular, it avoids some artifacts near the boundary.

Example 4: Kidney vascular system segmentation. The test image shown in Fig. 4.4(a) is a 256×256 MRA image of a kidney vascular system. The segmentation results from CV, CVTF, ADA, TFA are given in Fig. 4.4(b)–(e). For our TFA, we used $\epsilon = 0.003$ in (3.1), and it converges in 6 iterations with 0.78 seconds. The segmented vessels by CV and CVTF give unsatisfactory results since they are unable to recover the small occlusions along the coherence direction, while ADA and TFA both performed well. To compare ADA and TFA more closely, we enlarge the rectangular boxes in Fig.

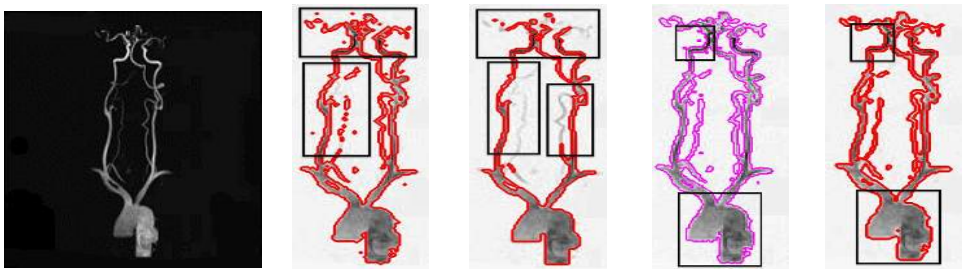


(a) Ground truth (b) clean image (c) noise and blur image

(d) CV [14] (e) CVTF [21] (f) ADA [24] (g) Our TFA method

(h) CV (2785) (i) CVTF (7038) (j) ADA (2834) (k) Our TFA (402)

FIG. 4.2. *Example 2. (a) Ground truth image; (b) Clean image; (c) Noisy and blurred image; (d)–(g) Results of methods CV [14], CVTF [21], ADA [24] and our TFA method, respectively; (h)–(k) Differences between the ground truth image (a) and the results of methods. Numbers in braces are the numbers of wrongly-detected pixels.*

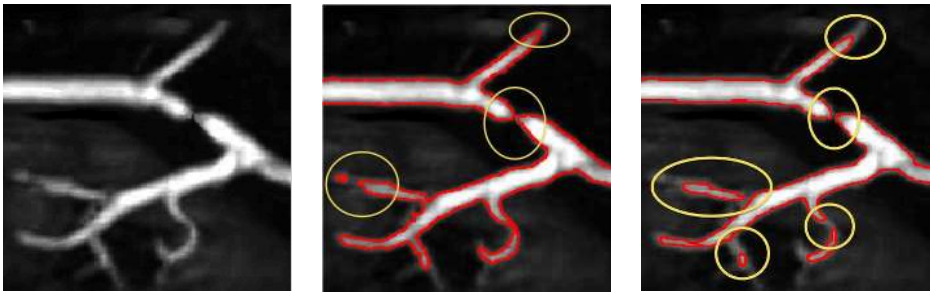


(a) Given MRA image (b) CV [14] (c) CVTF [21] (d) ADA [24] (e) Our TFA

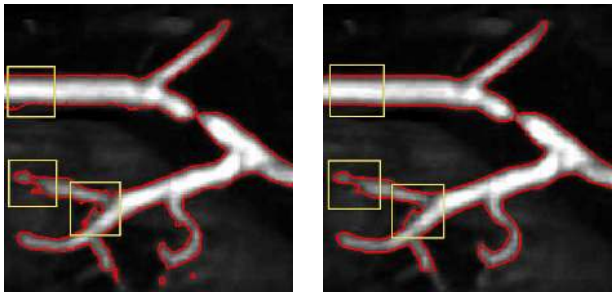
FIG. 4.3. *Example 3. Carotid vascular system segmentation. (a) Given MRA image; (b) CV; (c) CVTF; (d) ADA; (e) TFA.*

4.4(d) and (e) and depict them in Fig. 4.4(f)–(k). They show the smoothness of the detected boundaries using TFA and that ADA exhibits many artifacts which are well removed by TFA.

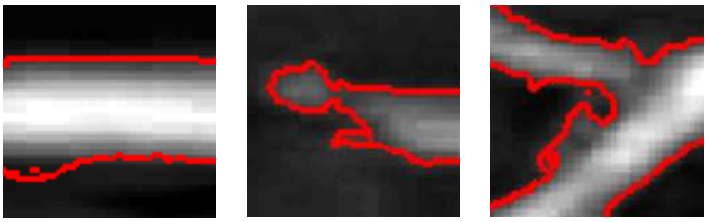
This example also shows the ability of our TFA to reconstruct structures which present small occlusions along the coherence direction. In Fig. 4.5, we report the first three and the final results obtained by the iterations of TFA. From $i = 0, \ldots, 3$, the results show that the pixels on the tip of the vessels are detected step by step



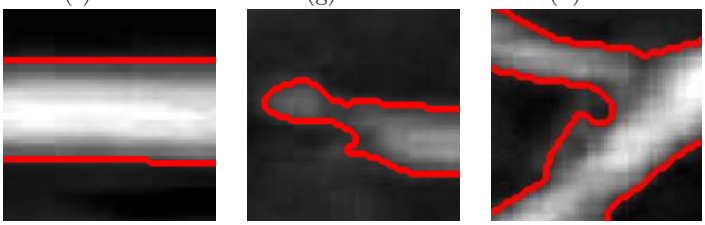
(a) Given MRA image (b) CV [14] (c) CVTF [21]



(d) ADA [24] (e) Our TFA



(f) ADA (g) ADA (h) ADA



(i) TFA (k) TFA (k) TFA

FIG. 4.4. *Example 4. Kidney vascular system segmentation. (a) Given MRA image; (b) CV; (c) CVTF; (d) ADA; (e) TFA; (f)–(h) zoomed-in details from rectangular boxes in (d); (i)–(k) zoomed-in details from rectangular boxes in (e).*

by our method, and they are used to connect the broken branches in the vessels. This demonstrates the ability of TFA to repair small occlusions and detect vessels effectively.

From these four 2D examples, we conclude that CV (the Chan-Vese active contour method [14]) and CVTF (the mixed tight-frame and Chan-Vese approach [21]) are not good. Thus in the next section where we consider 3D images, we will not compare with these two methods.

To illustrate how fast our TFA converges, we give in Table 4.1 the cardinality of

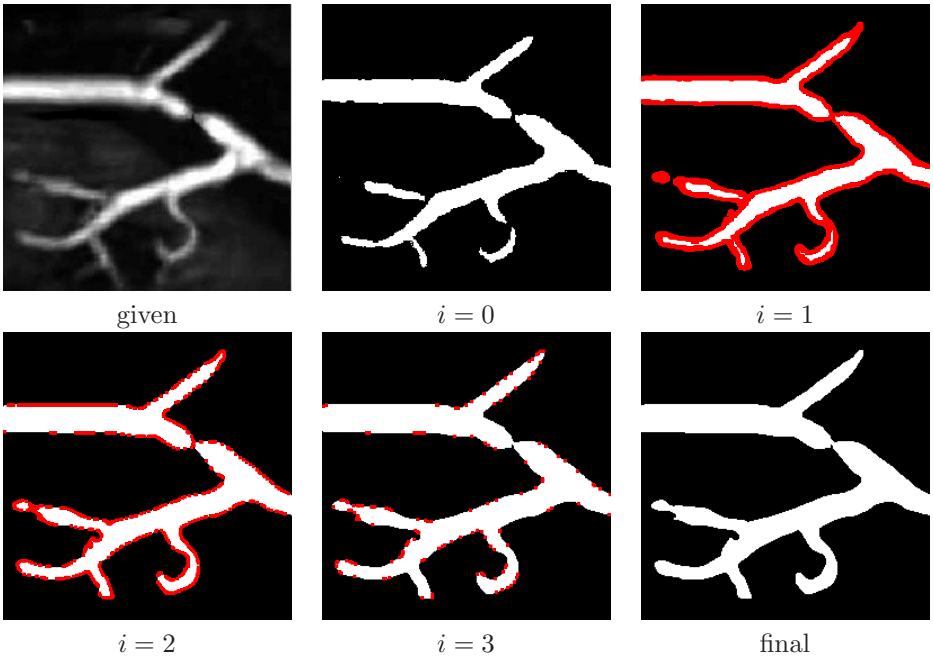


FIG. 4.5. *Example 4. Some iterations of the TFA applied to the kidney image. The white color represents the detected pixels on the vessels, and the red color represents the newly detected pixels on the vessels in step i for $i = 1, 2, 3$.*

$\Lambda^{(i)}$ at each iteration of TFA, i.e. the number of pixels that are unclassified yet in the ith iteration. The total number of pixels in the image is given by $|\Omega|$ on the top row. We see that our method converges very fast (within 6 iterations) and after 3 iterations, there are less than 200 pixels to be classified.

TABLE 4.1
Cardinality of $\Lambda^{(i)}$ at each iteration of TFA applied to the 2D examples

$\lvert\Lambda^{(i)}\rvert$	Example 1 $\lvert\Omega\rvert = 65536$	Example 2 $\lvert\Omega\rvert = 262144$	Example 3 $\lvert\Omega\rvert = 33124$	Example 4 $\lvert\Omega\rvert = 65536$
$i = 0$	9282	8472	1721	9444
$i = 1$	1848	1035	354	1943
$i = 2$	532	291	82	464
$i = 3$	162	84	26	133
$i = 4$	51	22	4	32
$i = 5$	12	5	0	8
$i = 6$	1	0	-	0
$i = 7$	0	-	-	-

4.3. 3D vessel segmentation. Here we tested on three 3D real images.

Example 5: Human head vessel segmentation. Figure 4.6 is a Magnetic Resonance Tomography Angiography (MRTA) image of a human head with skull removed, and viewed from three different orthogonal viewpoints. The volume data set has size of $256 \times 320 \times 128$ voxels and is provided by the Institute for Neuroradiology, Frankfurt,

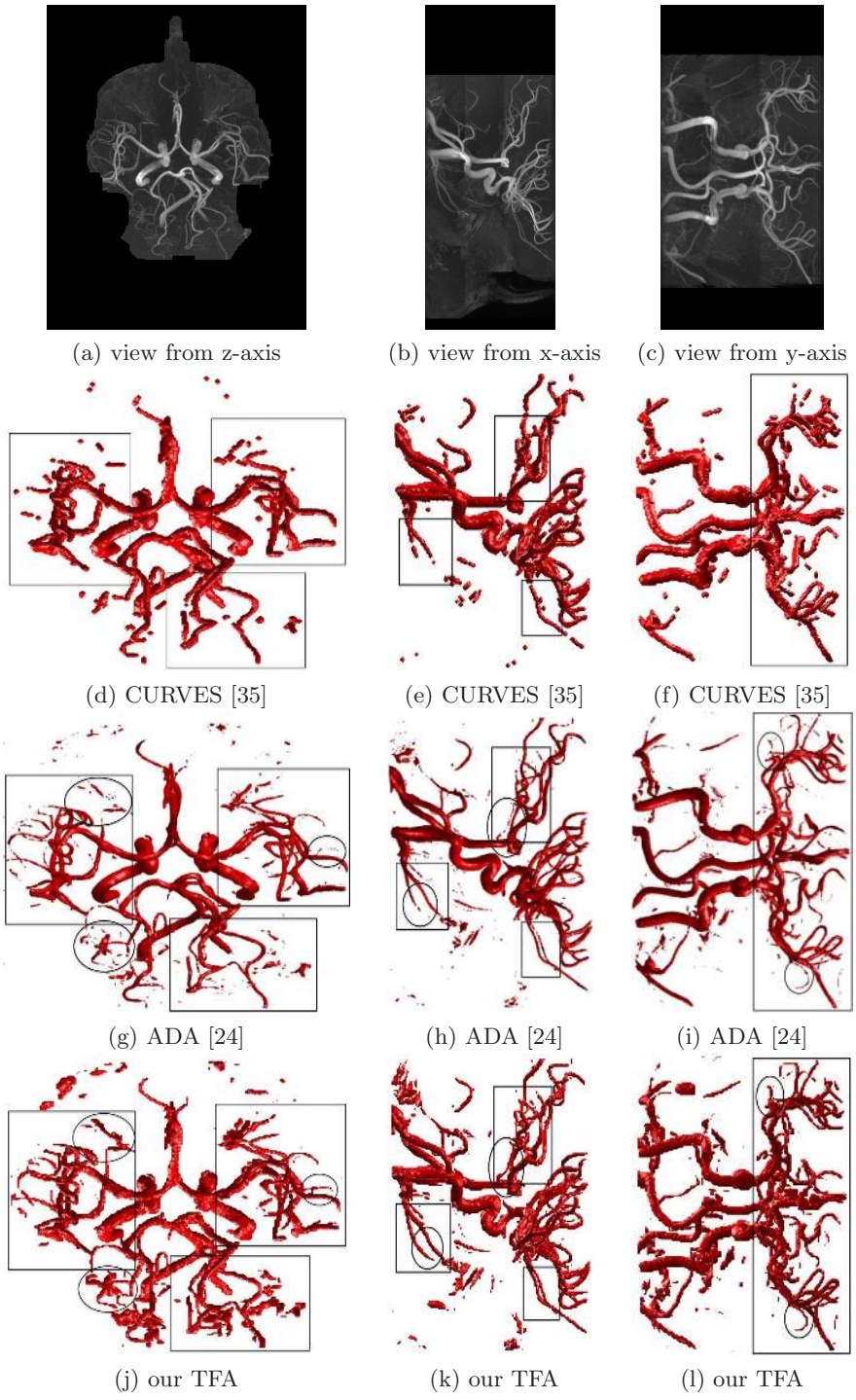
Germany [47]. The three columns in Figure 4.6 represent the three different viewpoints; and for each viewpoint, the maximum intensity projection of the raw data is shown in the first row, followed by the segmentation obtained by CURVES [35] (second row), ADA [24] (third row), and by TFA (fourth row). For our TFA, we used $\epsilon = 0.2$ in (3.1) and it converges in 8 iterations. The first column in Table 4.2 gives the cardinality $|\Lambda^{(i)}|$ at each iteration, which also shows how fast our method converges.

The volume data set presents a considerable number of thin vessels distributed around the thick vessels, and they are very hard to detect because of the weak contrast and the underlying noise. The visual comparison between corresponding viewpoints in the second, third and fourth rows highlights that CURVES fails to detect most of the thin vessels which are, instead, well captured by ADA and TFA. The differences between the results of ADA and TFA are highlighted by the ellipses in Fig. 4.6(g) – (l), moreover, Fig. 4.7 gives the zoomed-in results of these ellipses. Obviously, our TFA can detect thin vessels much better when compared with ADA which just detects some fragments belonging to the thin vessels in these ellipses. Our TFA is able to capture almost all the thin vessels correctly with much more details on them.

To assess the quality of the 3D results better, in the first row of Fig. 4.8, we show the axial, sagittal and coronal slices of the given volume data. The coordinate used to get the three directional slices is (100, 200, 100). Because of the extremely low intensity of most of the pixels in the images, it is difficult to see the boundary pixels clearly. To circumvent this, in Fig. 4.8 we show the figures by linearly inverting the intensity, i.e. mapping pixel value 0 to 1 and pixel value 1 to 0. The potential boundary pixels are those pixels with very small intensities (nearly black to completely black) in the figures. The segmentation results by CURVES, ADA and TFA are shown in the second, third and fourth rows of Fig. 4.8 respectively. We see that some thin vessels are not detected by CURVES and ADA whereas TFA has segmented the images with better accuracy.

Example 6: Kidney segmentation. The given data set is a Computed Tomographic Angiography (CTA) image of the kidney vascular system. It is of size $201 \times 201 \times 201$ voxels and is shown in Fig. 4.9(a). The image is severely corrupted by noise and blur, and exhibits vessels of different curvatures, diameters, as well as many complex bifurcations. We provide a comparison of TFA with CURVES and ADA, see Figs. 4.9(b), (c) and (d) respectively. The TFA is applied with $\epsilon = 0.06$ in (3.1) and converges in 9 iterations. From the parts in the Figs. 4.9(b) and (c) highlighted by the ellipses, we can see the improvement of ADA when comparing with CURVES. Comparisons between the images in Fig. 4.9 indicate that TFA is much less sensitive to blur and noise in the images than CURVES and ADA which fail to detect many thin vessels.

Example 7: Brain-neck vasculature segmentation. The given data set is an MRA image of the brain-neck vasculature system with aneurysms. It is of size $120 \times 250 \times 200$ voxels and is shown in Fig. 4.10(a). The data set is corrupted by strong noise which makes the thin vessels hard to see even by the naked eyes, and it has vessels of various diameters, curvatures and bifurcations. The segmentation results by CURVES, ADA and TFA are given in Figs. 4.10(b), (c) and (d) respectively. For TFA, we used $\epsilon = 0.06$ in (3.1), and it converges in 9 iterations only. The results clearly show that TFA can get more features out from the image. In Fig. 4.10(d), one may argue that there are some small isolated points in the image and the surface of the vessels is not smooth. This can easily be improved by smoothing our final binary image using



(a) view from z-axis (b) view from x-axis (c) view from y-axis

(d) CURVES [35] (e) CURVES [35] (f) CURVES [35]

(g) ADA [24] (h) ADA [24] (i) ADA [24]

(j) our TFA (k) our TFA (l) our TFA

FIG. 4.6. *Example 5. MRTA image of a human head volume data set. Row one: the maximum intensity projection of the volume dataset from three orthogonal viewpoints; Row two: CURVES segmentation; Row three: ADA segmentation; Row four: TFA segmentation.*

FIG. 4.7. *Example 5. Rows one and two: zoomed-in the ellipses of ADA and TFA results in Fig. 4.6 respectively.*

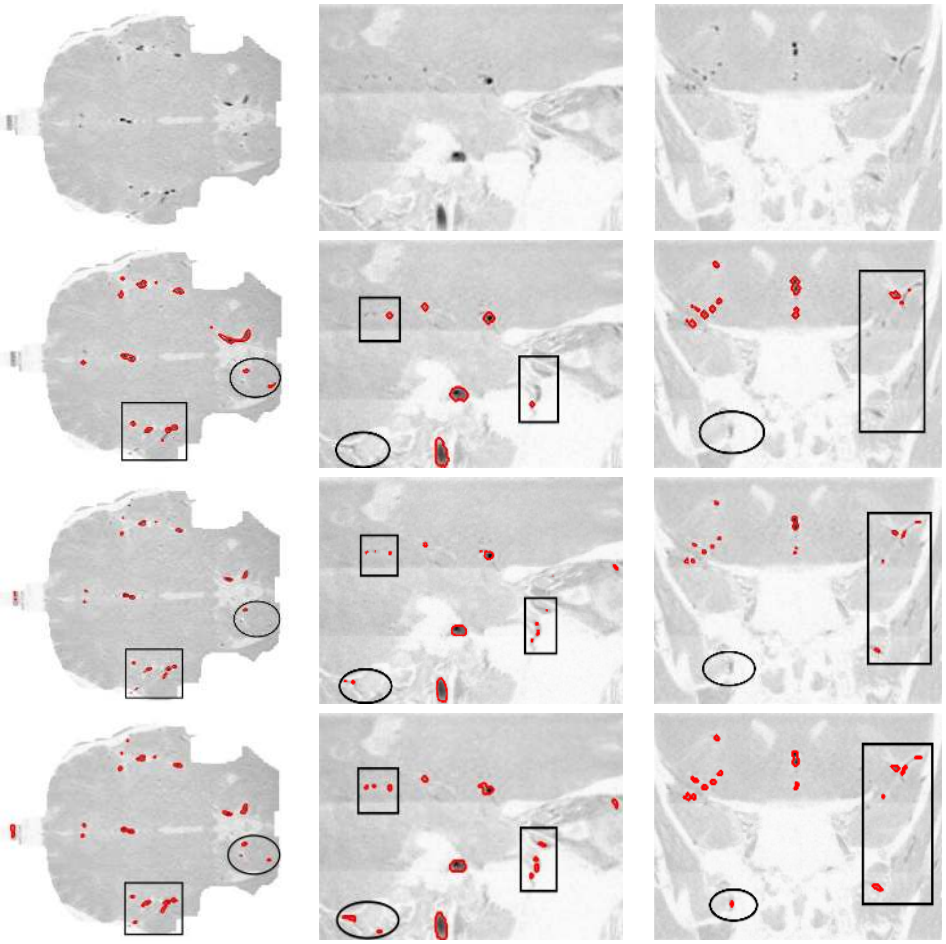
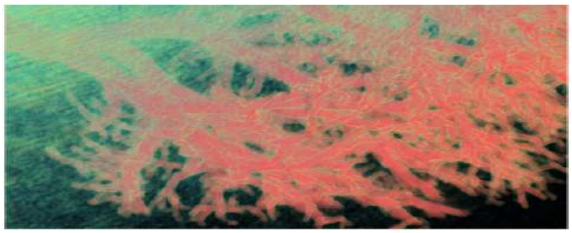
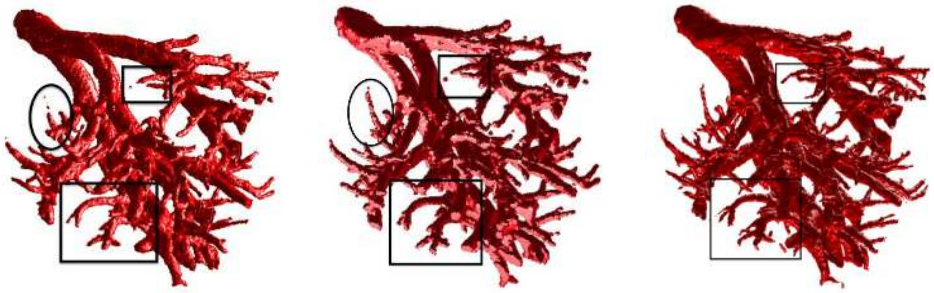


FIG. 4.8. *Example 5. Row one: the axial, sagittal and coronal slices of the MRTA image of a human head volume dataset used in Fig. 4.6; Row two: slices results by CURVES; Row three: slices results by ADA; Row four: slices results by TFA.*

the tight-frame formula (2.4). Fig. 4.10(e) shows the denoised-and-smoothed image after one iteration of (2.4) on the final image from TFA. We finally remark that our segmentation technique can be used to compute useful clinically measurements such as vessel radii in aneurysms.



(a) Given CTA image



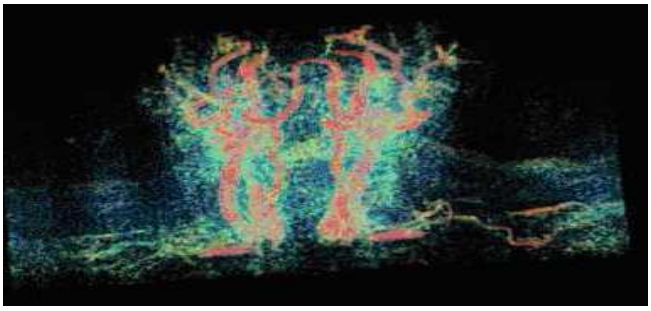
(b) CURVES [35] (c) ADA [25] (d) our TFA

FIG. 4.9. *Example 6. Segmentation of the kidney volume data set. (a) Given CTA image; (b) CURVES segmentation; (c) ADA segmentation; (d) TFA segmentation.*

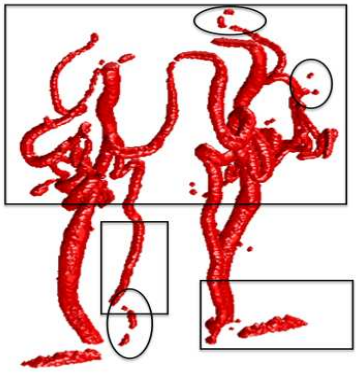
Again to assess the quality of the 3D results better, in the first row of Fig. 4.11, we show the axial, sagittal and coronal slices of the given volume data. The coordinate used to get the three directional slices is (80, 100, 100). Because of the extremely low intensity of most of the pixels in the images, we have again linearly inverted the intensity, i.e. mapping 0 to 1 and 1 to 0. The potential boundary pixels are those pixels with very small intensities (nearly black to completely black) in the figures. The segmentation results by CURVES, ADA and TFA are shown in the second, third and fourth rows of Fig. 4.11 respectively. We see that CURVES and ADA methods cannot detect many pixels that should be on the boundary of the vessels, especially those on the tips of the thin vessels. In contrast, TFA detects much more details of the thin vessels and with higher accuracy.

To illustrate how fast our TFA converges, we give in Table 4.2 the cardinality of $\Lambda^{(i)}$ at each iteration of TFA for the 3D test images. We see that our method converges within 9 iterations for all three images. Notice that the number of pixels in the images are large (more than 6 millions) and yet just after 4 iterations, there are just less than one thousand pixels not yet classified. This will be one important advantage of our algorithm when real-time processing is needed.

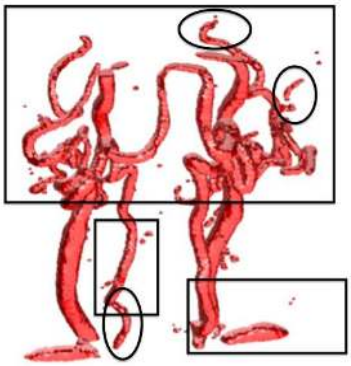
5. Conclusion and Future Work. In this paper, we introduced a new and efficient segmentation method based on the tight-frame approach. The numerical results demonstrate the ability of our method in segmenting tubular structures. The method can be implemented fast and give very accurate, smooth boundaries or surfaces. In addition, since the pixel values of more and more pixels will be set to either 0 or 1 during the iterations, by taking advantage of this, one can construct a sparse data structure to accelerate the method. Moreover, one can use different tight-frame systems such as those from contourlets, curvelets or steerable-wavelet [20, 10, 26] to capture more directions along the boundary. Though we have proved that our al-



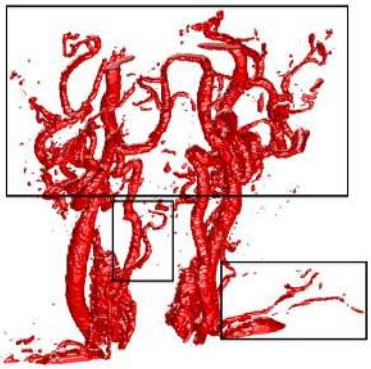
(a) Given MRA image



(b) CURVES [35]



(c) ADA [24]



(c) our TFA

(e) Result of TFA after smoothing

FIG. 4.10. *Example 7. Segmentation of the brain volume data set. (a) Given MRA image; (b) CURVES segmentation; (c) ADA segmentation; (d) TFA segmentation; (e) Result of TFA after smoothing by (2.4) once.*

gorithm will always converge to a binary image, it will be interesting to see what functional the binary image is minimizing. The framework for proving convergence for tight-frame algorithms, as developed in [5], may be useful here. These are the directions we will explore in the future.

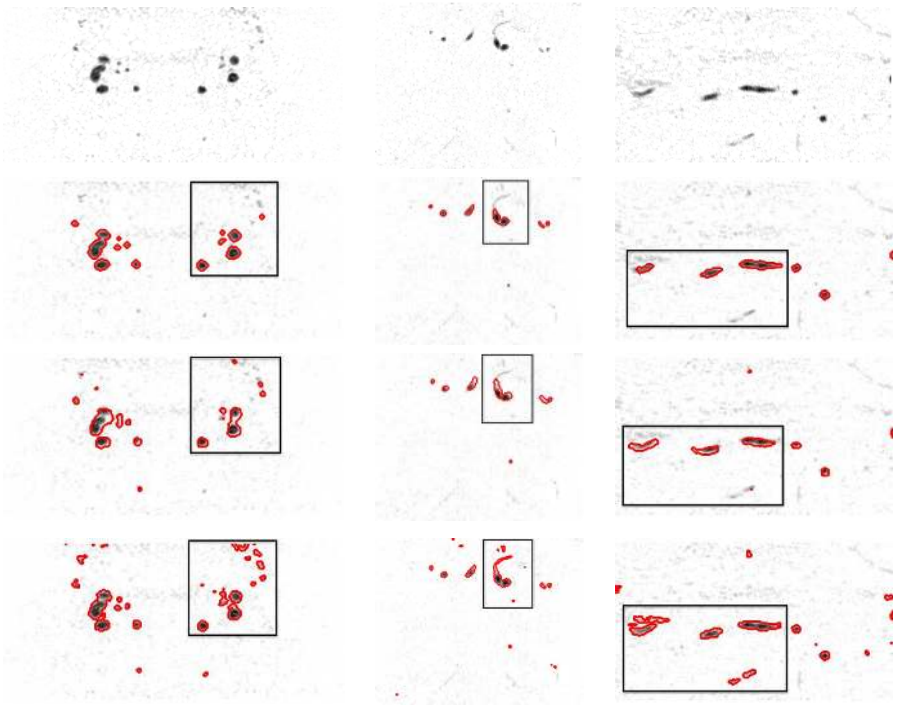


FIG. 4.11. *Example 7. Row one: the axial, sagittal and coronal slices of the brain volume dataset used in Fig. 4.10; Row two: slices results by CURVES; Row three: slices results by ADA; Row four: slices results by TFA.*

TABLE 4.2
Cardinality of $\Lambda^{(i)}$ at each iteration of TFA applied to the 3D examples

$\lvert\Lambda^{(i)}\rvert$	Example 5 $\lvert\Omega\rvert = 10485760$	Example 6 $\lvert\Omega\rvert = 8120601$	Example 7 $\lvert\Omega\rvert = 6000000$
$i = 0$	80686	137330	152898
$i = 1$	12938	32760	32064
$i = 2$	2981	8795	8565
$i = 3$	819	2475	2391
$i = 4$	227	689	650
$i = 5$	63	189	187
$i = 6$	16	56	59
$i = 7$	2	15	12
$i = 8$	0	3	2
$i = 9$	-	0	0

REFERENCES

[1] S. Arivazhagan and L. Ganesan. Texture segmentation using wavelet transform. *Pattern Recognition Letters*, 24, 3197–3203, 2003.

[2] J. Cai, R. Chan, L. Shen, and Z. Shen. Simultaneously inpainting in image and transformed domains. *Numer. Math.*, 112(4), 509–533, 2009.

[3] J. Cai, R. Chan, L. Shen, and Z. Shen. Restoration of chopped and nodded images by framelets.

SIAM J. Sci. Comput., 30, 1205–1227, 2008.
[4] J. Cai, R. Chan, L. Shen, and Z. Shen. Convergence analysis of tight framelet approach for missing data recovery. *Adv. Comput. Math.*, 31, 87–113, 2009.
[5] J. Cai, R. Chan, and Z. Shen. A framelet-based image inpainting algorithm. *Appl. Comput. Harmon. Anal.*, 24, 131–149, 2008.
[6] J. Cai, B. Dong, S. Osher, and Z. Shen. Image restoration: total variation; wavelet frames; and beyond. *J. Amer. Math. Soc.*, accepted for publication.
[7] S. Cai and K. Li. Matlab implementation of wavelet transforms. *http://taco.poly.edu/WaveletSoftware/*
[8] X. Cai, R. Chan, S. Morigi, and F. Sgallari. Framelet-based algorithm for segmentation of tubular structures. SSVM 2011, LNCS6667. *Springer,* 2011.
[9] E. Candes and D. Donoho. New tight frames of curvelets and optimal representations of objects with C2 singularities. *Comm. Pure Appl. Math*, 56, 219–266, 2004.
[10] E. Candès, L. Demanet, D. Donoho, and L. Ying. Fast discrete curvelet transforms. *Multiscale Modeling and Simulation*, 5(3), 861–899, 2006.
[11] R. Chan, T. Chan, L. Shen, and Z. Shen. Wavelet algorithms for high-resolution image reconstruction. *SIAM J. Sci. Comput.*, 24(4), 1408–1432, 2003.
[12] R. Chan, S. Setzer, and G. Steidl. Inpainting by flexible Haar-wavelet shrinkage. *SIAM J. Imaging Sci.*, 1(3), 273–293, 2008.
[13] T. Chan, S. Esedoglu, and M. Nikolova. Algorithms for finding global minimizers of image segmentation and denoising models. *SIAM J. Appl. Math*, 66(5), 1632–1648, 2006.
[14] T. Chan and L. Vese. Active contours without edges. *IEEE Trans. Image Process.*, 10(2), 266–277, 2001.
[15] B. Chapman, D. Parker, J. Stapelton, and D. Parker. Intracranial vessel segmentation from time-of-flight MRA using pre-processing of the MIP Z-buffer: accuracy of the ZBS algorithm. *Medical Image Analysis*, 8(2), 113–126, 2004.
[16] J. Chen and A. Amini. Quantifying 3D vascular structures in MRA images using hybrid PDE and geometric deformable models. *IEEE Trans. on Medical Imaging*, 23(10), 1251–1262, 2004.
[17] D. Cremers, M. Rousson, and R. Deriche. A review of statistical approaches to level set segmentation: Integrating color, texture, motion, and shape *Int. J. Comput. Vis.*, 72(2), 195–215, 2007.
[18] I. Daubechies. Ten lectures on wavelets. *vol. CBMS-NSF Lecture Notes, SIAM, nr.*, 61, 1992.
[19] I. Daubechies, B. Han, A. Ron, and Z. Shen. Framelets: MRA-based constructions of wavelet frames. *Appl. Comput. Harmon. Anal.*, 14(1), 1–46, 2003.
[20] M. Do and M. Vetterli. The contourlet transform: an efficient directional multiresolution image representation. *IEEE Trans. Image Process.*, 14(12), 2091–2106, 2004.
[21] B. Dong, A. Chien, and Z. Shen. Frame based segmentation for medical images. *Commun. Math. Sci.*, 32(4), 1724–1739, 2010.
[22] B. Dong and Z. Shen. MRA based wavelet frames and applications. *IAS Lecture Notes Series, Summer Program on The Mathematics of Image Processing, Park City Mathematics Institute*, 2010.
[23] D. Donoho. De-noising by soft-thresholding. *IEEE Trans. Inform. Theory*, 41(3), 613–627, 1995.
[24] E. Franchini, S. Morigi, and F. Sgallari. Segmentation of 3D tubular structures by a PDE-based anisotropic diffusion model. M. Dæhlen et al. (eds.): MMCS 2008, LNCS5862, pp. 224–241, 2010, *Springer-Verlag Berlin Heidelberg*, 2010.
[25] E. Franchini, S. Morigi, and F. Sgallari. Composed segmentation of tubular structures by an anisotropic PDE model. X.-C. Tai et al. (eds.): SSVM 2009, LNCS5567, pp. 75-86, 2009, *Springer-Verlag Berlin Heidelberg*, 2009.
[26] W. Freeman and E. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(9), 891–906, 1991.
[27] X. Gao, T. Nguyen, and G. Strang. Theory and lattice structure of complex paraunitary filterbanks with filters of (hermitian-) symmetry/antisymmetry properties. *IEEE Trans. Signal Processing*, 49(5), 1028–1043, 2001.
[28] R. Gonzales and R. Woods. *Digital Image Processing*, 3rd Ed., PrenticeHall, 2008.
[29] A. Gooya, H. Liao, et al. A variational method for geometric regularization of vascular segmentation in medical images. *IEEE Trans. Image Process.*, 17(8), 1295–1312, 2008.
[30] H. Hassan and A. Farag. Cerebrovascular segmentation for MRA data using levels set. *Proc. CARS*, pp. 246–252, 2003.
[31] N. Kingsbury. Image processing with complex wavelets. *Philos. Trans. R. Soc. London A, Math. Phys. Sci.*, 357(1760), 2543–2560, 1999.
[32] N. Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Appl. Com-*

put. Harmon. Anal., 10(3), 234253, 2001.
[33] C. Kirbas and F. Quek. A review of vessel extraction techniques and algorithms. *CV Computing Surveys*, 36, 81–121, 2004.
[34] K. Krissian, G. Malandain, N. Ayache, R. Vaillant, and Y. Trousset. Model-based detection of tubular structures in 3D images. *CVIU*, 80,130–171, 2000.
[35] L. Lorigo, O. Faugeras, E. Grimson, et al. Curves: curve evolution for vessel segmentation. *Medical Image Analysis*, 5, 195–206, 2001.
[36] T. Mcinerney and D. Terzopoulos. Deformable models in medical image analysis: A survey. *Medical Image Analysis*, 1(2), 91–108, 1996.
[37] A. Ron and Z. Shen. Affine systems in L2(Rd): the analysis of the analysis operator. *J. Funct. Anal.*, 148, 408–447, 1997.
[38] H. Scherl, J. Hornegger, M. Prummer, and M. Lell. Semi automatic level set segmentation and stenosis quantification of internal carotid artery in 3D CTA data sets. *Medical Image Analysis*, 11(1), 21–34, 2007.
[39] I. Selesnick, R. Baraniuk, and N. Kingsbury. The dual-tree complex wavelet transform. *IEEE Signal Process. Mag.*, 22(6), 123–151, 2005.
[40] F. Shi and I. Selesnick. Video denoising using oriented complex wavelet transforms. *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)* 2, 949–952, 2004.
[41] J. Starck, E. Candès, and D. Donoho. The curvelet transform for image denoising. *IEEE Trans. Image Processing*, 11(6), 670–684, 2000.
[42] K. Sum and P. Cheung. Vessel extraction under non-uniform illumination: A level set approach. *IEEE Trans. Biomed. Eng.*, 55(1), 358–360, 2008.
[43] M. Unser. Texture classification and segmentation using wavelet frames. *IEEE Trans. Image Process.*, 4(11), 1549–1560, 1995.
[44] P. Yan and A. Kassim. MRA Image Segmentation with Capillary Geodesic Active Contours. *Medical Image Analysis*, 10, 317–329, 2006.
[45] Z. Ye and C. Lu. A complex wavelet domain Markov model for image denoising. *Proc. IEEE Int. Conf. Image Processing, Barcelona.* 3, 365–368, 2003.
[46] D. Zonoobi, A. Kassim, and W. Shen. Vasculature segmentation in MRA images using gradient compensated geodesic active contours. *J. Sign. Process. Syst.*, 54, 171–181, 2009.
[47] *http://www.gris.uni-tuebingen.de/edu/areas/scivis/volren/datasets/new.html*