Paper:

Sazonov, I., Xie, X. & Nithiarasu, P. (2016). An improved method of computing geometrical potential force (GPF) employed in the segmentation of 3D and 4D medical images. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 1-10.

http://dx.doi.org/10.1080/21681163.2015.1124027

To appear in *Computer Methods in Biomechanics and Biomedical Engineering*
Vol. 00, No. 00, Month 20XX, 1–18

An improved method of computing geometrical potential force (GPF) employed in the segmentation of 3D and 4D medical images

Igor Sazonov[a*], Xianghua Xie[b] and Perumal Nithiarasu[a]

[a] *Zienkiewicz Centre for Computational Engineering, College of Engineering;*
[b] *Department of Computer Science*
Swansea University, Singleton park, Swansea, SA2 8PP, UK

(*v4.0 released February 2014*)

The geometric potential force (GPF) used in segmentation of medical images is in general a robust method. However, calculation of the GPF is often time consuming and slow. In the present work, we propose several methods for improving the GPF calculation and evaluate their efficiency against the original method. Among different methods investigated, the procedure that combines Riesz transform and integration by part provides the fastest solution. Both static and dynamic images have been employed to demonstrate the efficacy of the proposed methods.

Keywords: Image segmentation; Geometric potential force; Active surface methods; Riesz transform; Level set methods; Static and dynamic images

1. Introduction

In scan based modelling, segmentation is one of the difficult and time consuming processes. This can especially be non trivial if the scan quality is not sufficiently high (Sazonov et al. 2011). To address the lack of robustness and automation in dealing with typical quality medical images (containing noise, blurring, non-uniform brightness, etc.), a new segmentation method used on level set approach along with a potential force was introduced in (Yeo et al. 2011). The force introduced is referred to as the geometrical potential force (GPF), which belongs to active contour/surface gradient based segmentation methods (Malladi et al. 1995; Whitaker 2004). The GPF has been compared with several well known methods, such as the Generalized Gradient Vector Flow (GGVF) (Caselles et al. 1997; Xu and Prince 1998), the Elastic Interaction (EI) (Xiang et al. 2005), Chan&Vese's algorithm (Chan and Vese 2001). The comparison presented in (Yeo et al. 2011) demonstrates that the GPF method is robust and accurate in segmenting objects of complex topology and objects with weak edges. The noisy images can also be effectively handled by GPF based segmentation and the method is not sensitive to initialisation for the deformable model. For 2D images, the method can be reduced to the magneto-active contour (MAC) method proposed in (Xie and Mirmehdi 2008) and inherits all its advantages and strong theoretical basis. In contrast to the MAC method, the GPF method can be easily generalized to any dimension n of the image including 3D static ($n = 3$) and dynamic ($n = 4$) scans (Fleureau et al. 2006; Yang et al. 2013). Theoretically, the method can be applied to even higher dimensions (Boykov and Funka-Lea 2006), for example, segmenting coloured images, also the extraction of material properties from scans can be treated as an additional dimension. However, the GPF method in its current form is slow and it requires a huge memory overhead. As a result, segmentation is often limited to smaller image

[*] Corresponding author. Email: i.sazonov@swansea.ac.uk

sizes. Thus, present work is focused on improving the efficiency of the GPF calculation. A brief summary of the GPF based segmentation method is given below before introducing the proposed improvements.

The GPF based segmentation method is a two stage procedure. At the first stage, the so-called geometrical potential G is computed through the convolution of the image gradient ∇I with the vector kernel $\boldsymbol{K}$, i.e.,

$$G(\boldsymbol{x}_i) = \sum_{j \neq i} \nabla I(\boldsymbol{x}_j) \cdot c_n \frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{|\boldsymbol{x}_i - \boldsymbol{x}_j|^{n+1}} \equiv \nabla I * \boldsymbol{K} \tag{1}$$

where n is the domain dimension, $\cdot$ stands for the dot-product of two vectors; $\boldsymbol{x}_i$ is coordinate of the ith image grid point (voxel centre)[1]; c_n is a coefficient explained in Section 2.2, and $|\boldsymbol{x}_i - \boldsymbol{x}_j|$ is the Euclidean distance between grid points. In Equation 1, summation is performed through all the voxels constituting the image.

Once computed, the geometric potential G is used at the second stage to compute the force acting on the active surface. If the active surface is defined implicitly through the level set function $\Phi(\boldsymbol{x}_i, t)$, then the following partial differential equation (PDE) describing the evolution of Φ has to be numerically solved (see, e.g., (Malladi et al. 1995))

$$(\partial/\partial t)\Phi = \alpha\, g\kappa|\nabla\Phi| - (1-\alpha)\, \boldsymbol{F}\nabla\Phi \tag{2}$$

where α is the tuning parameter, $g = 1/(1 + |\nabla I|^2)$ is the stopping function, $\kappa = \nabla\hat{\boldsymbol{n}}$ is the curvature of isosurfaces of Φ, $\hat{\boldsymbol{n}} = \nabla\Phi/|\nabla\Phi|$ is the unit vector normal to isosurfaces of Φ, $\boldsymbol{F} = G\,\hat{\boldsymbol{n}}$ is the force moving the active surface, t is the virtual time used to evolve the level set function in order to find its equilibrium position in which the r.h.s. of (2) is zero.

The direct method of computing the geometrical potential G (see Eq. (1)) requires extremely large CPU time (see Table 1) and therefore can only be applied to small size 2D images (few hundred-by-few hundred pixels). Thus, direct computing is not recommended. Since the geometrical potential computation (1) is written in the form of convolution, a natural way to compute it is to apply the fast Fourier transform (FFT) as described in (Yeo et al. 2011). However, the FFT based method in its standard form requires rather large amount of memory when applied to any 3D scan and the memory requirement easily reaches limits of modern PCs when 4D scans are segmented. In fact, the image gradient and the kernel are nD vector functions and the direct nD FFT should be applied $2n$ times. The results of the FFT calculations have to be stored in $4n$ arrays of the size of the initial image(note that one has to store the real and imaginary parts of the Fourier spectrum). For a 3D image, approximately 20 arrays of initial image size are needed and more for a 4D images. Thus, currently used standard FFT can only be applied to small sized images (smaller that 256^3). This is the main reason why the GPF based method (Yeo et al. 2011) is not widely used in spite of all its other advantages.

To avoid the use of large amount of memory, we propose here a new method for computing geometric potential G. The method represents a combination of two independent approaches(Sazonov et al. 2012). One of the approaches is based upon the use an analytical formula for the Fourier spectrum of kernel $\boldsymbol{K}$ but can be calculated and multiplied to the kernel spectrum directly on every grid point. Thus there is no need to store all the components of the kernel and its Fourier spectrum. This approach results in two fold reducing the memory footprint. In another approach we modify Eq. (1) such that G can be computed through a convolution of scalar values: the scalar kernel $Q = \operatorname{div}\boldsymbol{K}$ and the image itself I rather than its gradient. This approach gives approximately n fold memory economy. The two approaches mentioned can be combined if an analytical formula of the Fourier spectrum of the scalar kernel used in the second approach can be derived. This can

[1] For the sake of brevity, we use the term voxel for all dimensions n instead of pixel, hypervoxel, etc. Analogously we use the term active surface for active contour (2D), active surface (3D), active hypersurface or n-surface (4D and higher).

result in further $2n$ fold memory economy. The new methods are however slightly more sensitive to noise when applied to real images and thus they need some regularization of the kernel and high spectral component filtering when applied to noisy images. Despite this, the numerical examples presented in Section 4 show that the proposed method is much more effective both in terms of memory size and CPU time requirements.

In the section below we rigorously derive all the analytical formulas necessary for implementing an effective version of the GPF method (Section 2), and we consider regularization and filtering of the kernel in Section 3. The numerical examples concentrating on 3D and 4D examples are provides in Section 4. All rigorous mathematical derivations and proofs are presented in Appendix A.

2. Methods for Computing Geometrical Potential (GP)

2.1 *The standard FFT based method*

Let $I(\boldsymbol{x})$, $I : \Omega \to \mathbb{R}$, describe a grayscale nD image in the domain $\Omega \subset \mathbb{R}^n$. We assume that $I \in C^\infty(\bar{\Omega}) \cap L^2(\Omega)$. The geometric potential $G(\boldsymbol{x})$ is a function defined by the integral with the integrand calculated via the dot product of two vector functions, i.e.,

$$G(\boldsymbol{x}) = \lim_{\varepsilon \to 0^+} \int_{\Omega_\varepsilon} \nabla I(\boldsymbol{x}') \cdot c_n \frac{\boldsymbol{x} - \boldsymbol{x}'}{|\boldsymbol{x} - \boldsymbol{x}'|^{n+1}} \, \mathrm{d}^n \boldsymbol{x}' \tag{3}$$

where $\Omega_\varepsilon = \Omega \setminus \{|\boldsymbol{x} - \boldsymbol{x}'| < \varepsilon\}$, i.e. an infinitesimal volume should be excluded from the integration domain Ω otherwise the integral would diverge when $\boldsymbol{x}' \to \boldsymbol{x}$ in all points in which $\nabla I(\boldsymbol{x}) \neq \boldsymbol{0}$. We can show that the limit in (3) exists if $\boldsymbol{x} \notin \partial\Omega$.

We now define kernel $\boldsymbol{K}$ and study its properties in the continuous space. We introduce the n-dimensional distribution $\boldsymbol{K} \in \mathcal{D}'$ as

$$\boldsymbol{K}(\boldsymbol{x}) = c_n \, \mathrm{p.v.} \frac{\boldsymbol{x}}{|\boldsymbol{x}|^{n+1}}. \tag{4}$$

where $\boldsymbol{K}$ should be defined as a continuous linear functional $\langle \boldsymbol{K}, \phi \rangle$ ($\mathcal{D} \mapsto \mathbb{R}^n$) acting on a probe function $\phi \in \mathcal{D}$ where $\mathcal{D}$ is a space of infinitely differentiable functions ($\phi \in C^\infty$) with compact support (e.g., (Schwartz 1954; Lighthill 1964; Gel'fand and Shilov 1964; Bremermann 1965; Jones 1982; Friedlander and Joshi 1998; Vladimirov 2002)). The functional is defined as the following inner product

$$\langle \boldsymbol{K}, \phi \rangle = c_n \lim_{\varepsilon \to 0^+} \int_{|\boldsymbol{x}| > \varepsilon} \phi(\boldsymbol{x}) \frac{\boldsymbol{x}}{|\boldsymbol{x}|^{n+1}} \, \mathrm{d}^n \boldsymbol{x}. \tag{5}$$

The discrete analogue of kernel $\boldsymbol{K}$ can be computed as

$$\boldsymbol{K}(\boldsymbol{x}_j) = \begin{cases} c_n \boldsymbol{x}_j / |\boldsymbol{x}_j|^{(n+1)}, & \boldsymbol{x}_j \neq \boldsymbol{0} \\ \boldsymbol{0}, & \boldsymbol{x}_j = \boldsymbol{0} \end{cases}. \tag{6}$$

Now, the following form of the direct and inverse Fourier transform can be employed:

$$\widetilde{f}(\boldsymbol{k}) \equiv \mathcal{F}[\, f\,](\boldsymbol{k}) = \int_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x})\, e^{-\mathrm{i}\boldsymbol{k}\boldsymbol{x}} \, \mathrm{d}^n \boldsymbol{x}$$

$$f(\boldsymbol{x}) \equiv \mathcal{F}^{-1}[\, \widetilde{f}\,](\boldsymbol{x}) = \frac{1}{(2\pi)^n} \int_{\boldsymbol{k} \in \mathbb{R}^n} \widetilde{f}(\boldsymbol{k})\, e^{\mathrm{i}\boldsymbol{k}\boldsymbol{x}} \, \mathrm{d}^n \boldsymbol{k}$$

where $\boldsymbol{k} = [k_1, \ldots, k_n]^T$ is the wavenumber and the Fourier image $\widetilde{f}$ is a spectrum of f. Now, applying the Convolution theorem (e.g., (Lighthill 1964)) we see that geometrical potential G can be calculated through the sequence of direct and inverse Fourier transforms, i.e,

$$G(\boldsymbol{x}) = \mathcal{F}^{-1}\big[\mathcal{F}[\nabla I](\boldsymbol{k}) \cdot \mathcal{F}[\boldsymbol{K}](\boldsymbol{k})\big] \tag{7}$$

The FFT based method described in (Yeo et al. 2011) utilizes the discrete analogue of Eq. (7). As mentioned in the introduction, we have to compute all the components of ∇I and $\boldsymbol{K}$ and store them, then we have to perform FFT for every component and store the real and imaginary parts. If we use an analytical formula for the kernel spectrum $\widetilde{\mathbf{K}}$, we can avoid all these operations on the kernel and reach the two-fold memory economy as discussed in the following subsection.

2.2 *Proposed modified approach*

If we define the coefficient $c_n = \frac{2}{\sigma_{n+1}}$ with σ_{n+1} being the area of unit hypersphere in $\mathbb{R}^{n+1}$, then for the infinite domain ($\Omega = \mathbb{R}^n$), integral (3) coincides with the Riesz transform in which kernel (4) has an explicit expression for its spectrum (e.g., (Stein 1970; Stein and Wiess 1971))

$$\mathcal{F}\left[\boldsymbol{K}\right](\boldsymbol{k}) = -\mathrm{i}\frac{\boldsymbol{k}}{|\boldsymbol{k}|}. \tag{8}$$

Note that

$$\sigma_n = \oint_{|\boldsymbol{x}|=1} \mathrm{d}^n\boldsymbol{x} = \frac{2\pi^{n/2}}{\Gamma(n/2)} \tag{9}$$

where $\Gamma(\cdot)$ is the gamma-function, that gives

$$c_1 = \frac{1}{\pi}, \quad c_2 = \frac{1}{2\pi}, \quad c_3 = \frac{1}{\pi^2}, \quad c_4 = \frac{3}{4\pi^2}, \ldots \tag{10}$$

Substituting the analytical expression (8) into Equation (7) we obtain

$$G(\boldsymbol{x}) = \mathcal{F}^{-1}\left[\mathcal{F}[\nabla I](\boldsymbol{k}) \cdot \frac{-\mathrm{i}\boldsymbol{k}}{|\boldsymbol{k}|}\right] \tag{11}$$

instead of computing the kernel in the x-space and then applying the FFT method as in (7).

Another approach to decrease the memory used is via re-arranging the integrand in (3) as a product of scalar function and a scalar kernel. Applying a sort of integration by parts, we can transfer the nabla operator from the image gradient to the kernel, i.e.,

$$G(\boldsymbol{x}) = \int I(\boldsymbol{x}')\,(\nabla\boldsymbol{K})(\boldsymbol{x}-\boldsymbol{x}')\,\mathrm{d}^n\boldsymbol{x}' \tag{12}$$

where $\nabla\boldsymbol{K} \equiv \operatorname{div}\boldsymbol{K}$ is a formal scalar kernel. As kernel $\boldsymbol{K}$ is a distribution, we should show that the scalar kernel $\nabla\boldsymbol{K} \in \mathcal{D}'$ and integral (12) make sense. To do this we introduce the distribution

$$Q(\boldsymbol{x}) = -c_n\mathcal{P}\frac{1}{|\boldsymbol{x}|^{n+1}} \tag{13}$$

defined as the following inner product

$$\left\langle \mathcal{P}\frac{1}{|\boldsymbol{x}|^{n+1}}, \phi(\boldsymbol{x}) \right\rangle = \text{p.v.} \int_{\mathbb{R}^n} \frac{\phi(\boldsymbol{x}) - \phi(\boldsymbol{0})}{|\boldsymbol{x}|^{n+1}} \, \mathrm{d}^n\boldsymbol{x}. \tag{14}$$

It can be shown that $Q(\boldsymbol{x}) = \nabla \boldsymbol{K}(\boldsymbol{x})$ in the $\mathcal{D}'$ space means $\langle Q, \phi \rangle = -\langle \boldsymbol{K}, \nabla\phi \rangle$. Kernel Q is a regular function: $Q = -c_n\,|\boldsymbol{x}|^{-n-1}$ for all $\boldsymbol{x}$ except the origin and has a singularity at $\boldsymbol{x} = \boldsymbol{0}$. The singularity is such that integral of Q through the total space is zero $\int\limits_{x\in\mathbb{R}^n} Q(\boldsymbol{x})\,\mathrm{d}^n\boldsymbol{x} = 0$.

Proposition 2.1. *The geometrical potential can be calculated through the following convolution of scalar functions*

$$G(\boldsymbol{x}) = c_n \lim_{\varepsilon\to 0^+} \int\limits_{|\boldsymbol{x}'|>\varepsilon} \frac{I(\boldsymbol{x}') - I(\boldsymbol{x} - \boldsymbol{x}')}{|\boldsymbol{x}'|^{n+1}} \, \mathrm{d}^n\boldsymbol{x}'. \tag{15}$$

The proof of the above expression is presented in Appendix A. A discrete analogue of scalar kernel Q can be approximated as follows

$$Q(\boldsymbol{x}_j) = \begin{cases} -\dfrac{c_n}{|\boldsymbol{x}_j|^{n+1}},\ \boldsymbol{x}_j \neq \boldsymbol{0} \\ S,\ \boldsymbol{x}_j = \boldsymbol{0} \end{cases}, \qquad S = \sum_{\boldsymbol{x}_j\neq\boldsymbol{0}} \frac{c_n}{|\boldsymbol{x}_j|^{n+1}}. \tag{16}$$

This guaranties that $\sum_{\boldsymbol{x}_j\in\Omega} Q(\boldsymbol{x}_j) = 0$. Thus the geometric potential may now be computed as

$$G(\boldsymbol{x}) = \mathcal{F}^{-1}\big[\mathcal{F}[I](\boldsymbol{k})\,\mathcal{F}[Q](\boldsymbol{k})\big] \tag{17}$$

where Q is defined by (16) in the discrete case.

The equations (11) and (17) may be combined together to give a new method. The nabla operator the in $\boldsymbol{x}$-space is equivalent to the multiplication by i$\boldsymbol{k}$ in the $\boldsymbol{k}$-space. Since an analytical formula for vector kernel is known and given by (8), we easily derive the analytical formula for the scalar kernel spectrum as

$$\widetilde{Q} = \mathrm{i}\boldsymbol{k}\boldsymbol{K} = |\boldsymbol{k}|\,. \tag{18}$$

Therefore, in the new method, the geometric potential is computed through direct and inverse FFT of the scalar valve, i.e.,

$$G(\boldsymbol{x}) = \mathcal{F}^{-1}\Big[\mathcal{F}[I](\boldsymbol{k})\,\big(\,|\boldsymbol{k}|\,\big)\Big]. \tag{19}$$

In this new method we have to store only three arrays of the original image size. Thus, this method substantially reduces the memory requirement compared to the standard FFT procedure described earlier.

Table 1 shows the comparison of CPU time for various options available. The numerical computations confirm that the new method is the most efficient in terms of memory and CPU time requirements for computing geometrical potential. The model problem used to produce the results of Table 1 is the synthetic image described in Section 4.

Although the new method increases the efficiency substantially, as soon as noise is added to synthetic images or when dealing with real medical scans, this method is sensitive. This results in unsatisfactory value of geometrical potential. Thus, some kernel regularization methods are explained in the following section to reduce sensitivity to noise.

Table 1. CPU time and memory recruitment for computation of G for a 3D image size of $256^3 = 0.14$G.

	Memory required	CPU time
Direct method (Equation (1))	0.6G	∼ 7days
Standard FFT based method (Yeo et al. 2011) (Equations (6)–(7))	1.8G	91s
Approach based on Equation (11)	1.0G	55s
Approach based on Equations (16)–(17)	0.6G	42s
Recommended new method: (Equation (19))	0.4G	30s

3. Kernel Regularization

3.1 *Smoothing the impact of discrete domain*

The enhanced noise sensitivity of the new method is mainly caused by the way in which gradient of $I(\boldsymbol{x})$ is computed. In the direct and FFT method described in (Yeo et al. 2011), it is calculated via the central differences which, for example, in 1D case has the form,

$$\frac{\partial I}{\partial x} \approx \frac{I(x+h) - I(x-h)}{2h}. \tag{20}$$

whereas in the new method it is computed through the FFT, i.e.,

$$\frac{\partial I}{\partial x} \approx \mathcal{F}^{-1}\big[\mathrm{i}k\,\mathcal{F}[\,I\,]\big]. \tag{21}$$

Computation of derivative through the FFT transform provides a higher order accuracy (comparable with the number of grid points) whereas the central difference provides second order accuracy. Therefore on a smooth synthetic image having the spatial frequencies much smaller than the Nyquist cutoff wavenumber, the FFT based method can approximate its gradients better. Nevertheless, for a real image, often containing δ-correlated random noise (i.e with the spatial frequencies exceeding the Nyquist cutoff wavenumber), the FFT based method can be less accurate compared with the lower order methods and therefore needs some filtering (see Aprovitola and Gallo (2014)).

To illustrate this, we consider, a smooth 1D function $I(x) = \exp\{-x^2\}$ set on $x_j = \{-7, -6, \ldots, 8\}$. The error of this function derivative computed by the central difference method is 0.24, whereas the error of derivative computed through the FFT is only 0.08 as shown in Figure 1(left).

However if the function is not smooth (for example, contains δ-correlated noise) the situation is quite opposite. Consider, as an example, a discrete implementation of Dirac's delta function, $\delta(x)$. The derivative computed by the central differences gives a reasonable approximation for $\delta'(x)$ coinciding with its numerical implementation (dashed green), whereas the FFT method gives an oscillatory and unrealistic result as depicted in Figure 1(right).

The oscillatory response of the FFT method may be reduced by computing the central difference using FFT. To do this, recollect that in a continuous, infinite space the derivative can be expressed as a convolution using $\delta'(x)$ as

$$\partial I/\partial x = I(x) * \delta'(x) = \int_{-\infty}^{+\infty} I(u)\,\delta'(x-u)\,\mathrm{d}u.$$

The Fourier transform results in the spectrum $\mathcal{F}[\delta'(x)](k) = \mathrm{i}k$. The central differences can also be computed analogously as a convolution with the function $\frac{1}{2h}\big(\delta(x+h) - \delta(x-h)\big)$ having a spectrum of

$$\mathcal{F}\Big[\frac{1}{2h}\big(\delta(x+h) - \delta(x-h)\big)\Big] = \frac{\mathrm{i}}{h}\sin(kh) \tag{22}$$

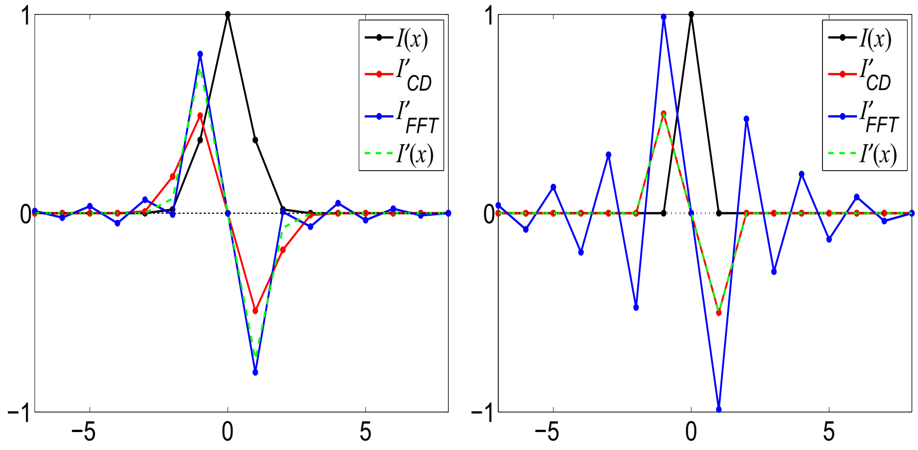


Figure 1. Function $I(x)$ (black): Right: $I(x) = \exp\{-x^2\}$, left: $I(x) = \delta(x)$; derivative computed by central-difference (red), the same—through FFT (blue). The dashed green line represents the exact analytical derivative (left) or the numerical implementation of $\delta'(x)$ (right).

which tends to ik if $h \to 0$. In an nD case, spectrum of the gradient operator, i$\boldsymbol{k}$, should be replaced by vector i$\boldsymbol{q}$ where

$$\boldsymbol{q}(\boldsymbol{k}, \boldsymbol{h}) = \left[\frac{\sin k_1 h_1}{h_1}, \frac{\sin k_2 h_2}{h_2}, \ldots, \frac{\sin k_n h_n}{h_n}\right]^T. \tag{23}$$

Here $\boldsymbol{h} = [h_1, h_2, \ldots, h_n]^T$ is a vector of the grid size. Now Eq. (19) may be transformed to give,

$$G(\boldsymbol{x}) = \mathcal{F}^{-1}\left[\mathcal{F}[I](\boldsymbol{k})\left(\mathrm{i}\boldsymbol{q} \cdot (-\mathrm{i})\frac{\boldsymbol{k}}{|\boldsymbol{k}|}\right)\right], \tag{24}$$

As seen in Figure 1(right) the FFT based central difference calculation gives a solution that is identical to direct central difference method. Thus, the proposed new method (Equation (24)) with the FFT based central difference is expected to give more accurate solution.

3.2 *Improving the impact of finite domain*

The analytical solution for the spectrum derived previously (Equation(8)) is for an infinite domain. To employ this expression on finite domains, a discrete approximation of Equation (8) is essential. However, such a discrete approximation develops spatial oscillations of the kernel in the physical domain. To demonstrate this, lets consider a 1D problem for which integral (3) coincides with the standard Hilbert transform. A discrete analogue of kernel $K = \text{p.v.}\frac{1}{\pi x}$ in a 1D finite domain (16 grid points) is depicted in Figure 2(left) by the blue line, and its FFT image, $\mathcal{F}[K]$, is drawn by the same colour line in Figure 2(right). The analytical kernel spectrum computed by (8) has a constant magnitude: $|\widetilde{\mathbf{K}}| = const$ everywhere except at origin as shown by dashed line in Figure 2(right) and Its inverse FFT image, $\mathcal{F}^{-1}[\widetilde{K}]$, is drawn by the same dashed line in Figure 2(left). As we see from the figures, the numerical spectrum $\mathcal{F}[K]$ decays approximately linearly through the k-domain in contrast to constant magnitude of the analytical spectrum $\widetilde{K}$. Also, the numerical spectrum has some unwanted oscillation. Both the disagreement between analytical and numerical spectrums and oscillations in numerical spectrum are caused by the finiteness of the domain.

Although we use finite domains, the use of FFT treats kernel $\boldsymbol{K}$ as a periodically repeated function in an infinite domain. Thus, the kernel has a discontinuity at the boundary of any finite domain as shown in Figure 2(left) by a light blue segment. The FFT shift is used just to display this border discontinuity more clearly. This border discontinuity generates the unwanted oscilla-

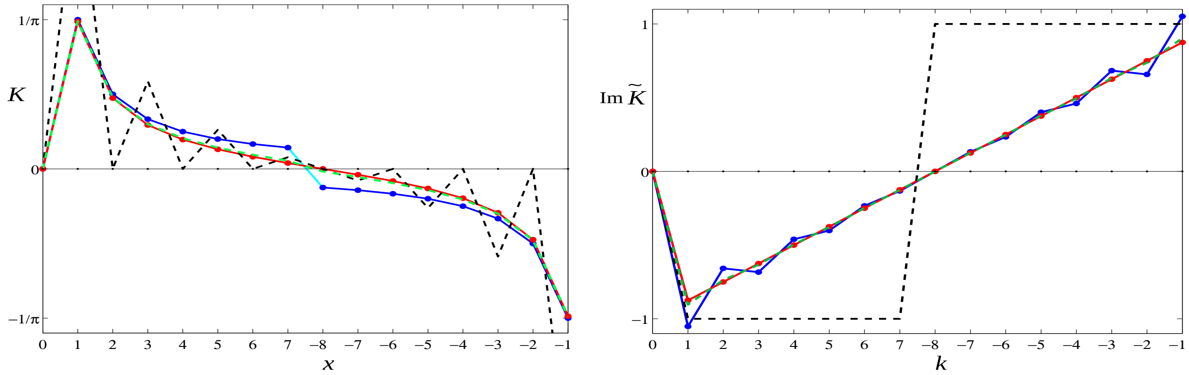


Figure 2. Left: discretized kernels; Right: their spectra. Red: kernel p.v.$\frac{1}{x}$. Black-dashed: kernel having spectrum iπ sign k. Green: the kernel based on the Hilbert transform on a circle. The kernels and their spectra are plotted in the FFT shifted domain: negative values are located after the positive ones.

tions in the kernel spectrum as shown in Figure 2(right, blue). This is manifestation of the Gibbs phenomenon (e.g., (Arfken and Weber 1995)).

This border discontinuity can be removed in 1D case by the use of the Hilbert transform on the unit circle with the kernel $\frac{\mathrm{i}}{2\pi}$p.v. $\cot(x/2)$, $x \in [0, 2\pi)$ (e.g., (Kress 1989; Schneider 1998)) instead of the standard Hilbert transform. This approach is equivalent to replacing the original kernel by an infinite sum of the same kernels periodically distributed along the domain, i.e,

$$K = \frac{1}{\pi}\sum_{m=-\infty}^{+\infty}\frac{1}{x - Lm} = \frac{1}{L}\cot(\frac{x\pi}{L}), \quad x \neq 0, \pm L, \ldots$$

where L is the domain size. This kernel and its spectrum are drawn by red lines in Figure 2. One can see that the function is smooth everywhere except at the singular point ($x = 0$) and its spectrum behaves linearly up to the highest wavenumber $k_{\max}$. The discrete spectrum values calculated is exactly represented by the following formula:

$$\widetilde{K} = -\mathrm{i}\,\mathrm{sign}(k)\left(1 - k/k_{\max}\right), \qquad k_{\max} = \pi/h \tag{25}$$

where h is the grid size.

In nD, the analogous approach gives the following sum for the kernel

$$\boldsymbol{K}(\boldsymbol{x}) = \sum_{m_1,\ldots,m_n=-M}^{M}\frac{\boldsymbol{x} - \boldsymbol{m}\cdot\boldsymbol{L}}{|\boldsymbol{x} - \boldsymbol{m}\cdot\boldsymbol{L}|^{n+1}}, \qquad \boldsymbol{x} \neq \boldsymbol{0} \tag{26}$$

where $\boldsymbol{L} = [L_1, \ldots, L_n]^T$ is a vector of the physical domain size; $\boldsymbol{m} = [m_1, \ldots, m_n]^T \in \mathbf{Z}^n$. For $M = \infty$, equation (26) can be treated as a nD generalization of the Hilbert transform on a circle. Unfortunately, unlike1D case, there is no known explicit formula for its spectrum. For segmentation purposes, we propose a suitable approximation for $\widetilde{\boldsymbol{K}}(\boldsymbol{k})$ below.

Analysis of the spectrum computed for the kernel evaluated through (26) and also a comparison with 1D spectrum (25), give the following approximation for the spectrum in multi dimensions

$$\widetilde{\boldsymbol{K}} = -\mathrm{i}\frac{\boldsymbol{k}}{|\boldsymbol{k}|}\left[\left(1 - |\boldsymbol{k}'|\right) + V(\boldsymbol{k})\right], \qquad \boldsymbol{k}' = \left[\frac{k_1}{k_{1,\max}}, \ldots, \frac{k_n}{k_{n,\max}}\right]^T \tag{27}$$

where $\widetilde{\boldsymbol{K}}(\boldsymbol{0}) = \boldsymbol{0}$. Here $V(\boldsymbol{k})$ is a function making the spectrum smoother at the highest wave number: vanishing when $k_i \to k_{i,\max}$; $k_{i,\max} = \pi/h_i$. The following approximation of $V(\boldsymbol{k})$ gives a

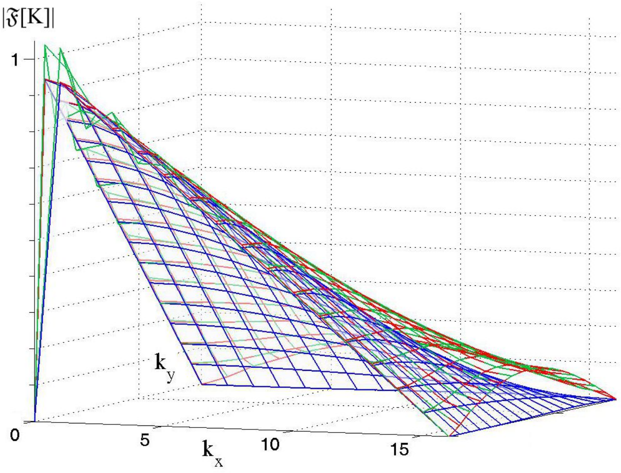


Figure 3. Absolute value of the spectrum for a 32×32. Green: computed via (6) and FFT. Red: computed via (26) for $M = 5$ and FFT. Blue: computed analytically by (27)–(29).

rather smoother spectrum

$$V_n = \sum_{m=2}^{n} a_m \sum_{i_1 \neq \ldots \neq i_m} v_{i_1} \cdots v_{i_m},\ v_i = 1 - \cos\frac{\pi k_i}{2k_{i,\max}} \tag{28}$$

with

$$a_n = -f_n - \sum_{m=2}^{n-1} \binom{m}{n} a_m, \qquad f_n = 1 - \sqrt{n} \tag{29}$$

In multi dimensions with $n = 2, 3$ and 4 the equations take the forms

$$V_2 = a_2 v_1 v_2, \qquad a_2 = -f_2 = 1 - \sqrt{2},$$

$$\begin{aligned} V_3 &= -a_2 \left(v_1 v_2 + v_2 v_3 + v_3 v_1\right) - a_3 v_1 v_2 v_3 \\ a_3 &= -f_3 - 3a_2 = 2 - 3\sqrt{2} + \sqrt{3} \end{aligned}$$

and

$$\begin{aligned} V_4 &= a_2 \left(v_1 v_2 + v_2 v_3 + v_3 v_1 + v_1 v_4 + v_2 v_4 + v_3 v_4\right) \\ &\quad + a_3 \left(v_1 v_2 v_3 + v_2 v_3 v_4 + v_3 v_4 v_1 + v_4 v_1 v_2\right) + a_4 v_1 v_2 v_3 v_4 \\ a_4 &= -f_4 - 6a_2 - 4a_3 = -1 + 6\sqrt{2} - 4\sqrt{3} \end{aligned}$$

respectively.

The method proposed appears to have a large number of steps. The standard computation of geometrical potential for a 256^3 needs 22.12s without correction, 22.65s with correction (23)–(24) and the method takes only 22.95s if correction (27)–(29) is also added.

Kernel spectra computed by different methods for a 2D image are shown in Figure 3. The green mesh shows kernel spectrum computed as in (Yeo et al. 2011). It has unwanted oscillations at low wave numbers. The red mesh shows spectrum for a generalized Hilbert kernel on a circle, Eq. (6) combined with FFT. One can see that it behaves smoothly for low wave numbers and almost coincides with the green mesh at high wave numbers. The kernel spectrum computed directly in the k-space via (27)–(29) is shown using the blue mesh. It behaves smoothly and almost coincides with that shown by the red mesh but decays faster when wave number approaches the highest value.

4. Numerical examples

In this section, some synthetic and real images used to calculate G are segmented using the improved algorithm proposed. The first step of the algorithm involves calculation of the geometrical potential (1) which is computed through Equations (6)–(7) (Yeo et al. 2011) and, accounting for the kernel regularization corrections, through

$$G = \mathcal{F}^{-1}\left[\mathcal{F}[I]\,\widetilde{Q}\right], \qquad \widetilde{Q} = \frac{\mathbf{q}\cdot\mathbf{k}}{|\mathbf{k}|}(1 - |\mathbf{k}'| + V(\mathbf{k})) \tag{30}$$

where $\mathbf{q}$ is given by (23), $\mathbf{k}'$ is defined in (27) and $V(\mathbf{k})$ is given by (28)–(29).

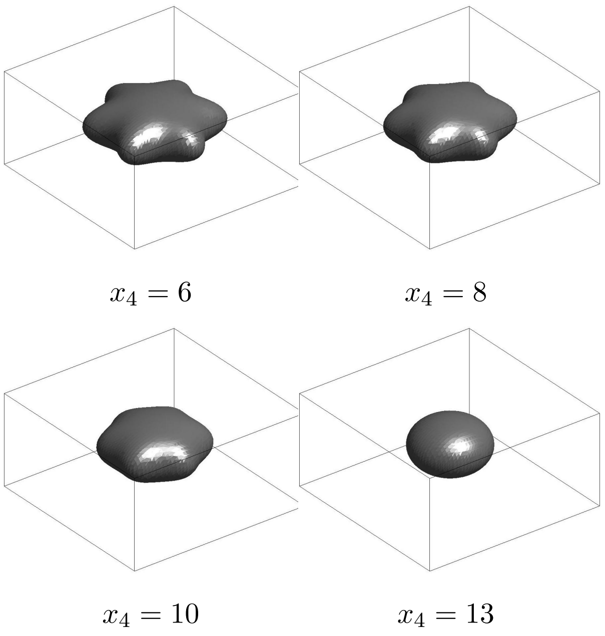
Been computed once, the geometric potential is used for the integration of PDE (2) for level set function $\Phi(\mathbf{x}, t)$ which defines implicitly ($\Phi(\mathbf{x}, t) = 0$) the active surface. The implementation of the stage is described in (Yeo et al. 2011) and (Sazonov et al. 2011) in detail. The initial surface is set inside the object to be segmented. For the real medical image, it has to be set manually (as it can contain a number of objects) but rather arbitrary as the algorithm is quite robust for the choice of initial surface. In (Yeo et al. 2011) initial surface $\Phi(\mathbf{x}, 0) = 0$ is set as a parallelepiped and the initial level set then is calculated through the distance transform from its boundary. Here we prefer to set the initial surface as a ball due to smaller number of parameters needed and to work with the non-smooth level set function defined as $\Phi(\mathbf{x}, 0) = -1$ inside the ball and $\Phi(\mathbf{x}, 0) = 1$ outside it which does not affect the final result. The integration is stopped when the binary image of the segmented object (defined as a set of voxels in which $\Phi(\mathbf{x}, t) \le 0$) remains the same in the subsequent time steps.

4.1 *Synthetic images*

The 3D synthetic geometry used here is a three dimensional star as shown in Figure (4). Analytical level set functions are used on a grid to describe the object. A small Gaussian delta-correlated noise is added to the analytical level set. The total domain size is $64 \times 64 \times 32$ and the centre of the star coincides with the domain centre. The 4D image of the star is created by elongating and shortening the star arms along the x_4-axis which is the physical time coordinate in contrast to the virtual time t used in (2) for evolution of the level set function. Its size is $64 \times 64 \times 32 \times 16$. Thus, the image contains 16 different instances of 3D images and some of them are shown in Figure 4. The star arms vary in time periodically with the period of 16 but not harmonically (to have a richer spectrum along the x_4-coordinate).

In Figures 5, 6, 7, the variation of $G(\boldsymbol{x})$ computed along x_1-axis for the 3D object is shown. The remanning coordinates are fixed at the centre, i.e., $x_2 = 32$ and $x_3 = 16$. Thus, the geometrical potential is shown along the line passing through the centre of the object. Also in these figures, the black curve indicates the segmentation method based on threshold, i.e, the function $I_m - I(\boldsymbol{x})$ is plotted with $I_m = \frac{1}{2}(I_{\max} + I_{\min})$ being the threshold value.

In Figure 5, the geometrical potential computed using Equation (19) without any kernel correction is compared against the threshold method, direct calculation of G and standard form of FFT used in reference (Yeo et al. 2011) . One can see that despite all the methods produce a geometrical potential distribution suitable for segmentation (points $G = 0$ are close to points $I = I_m$). However the proposed new method, despite giving a sharp distribution of G, is more sensitive to noise and produced oscillatory G distribution.



$x_4 = 6$ $x_4 = 8$

$x_4 = 10$ $x_4 = 13$

Figure 4. Object shape at various instances.

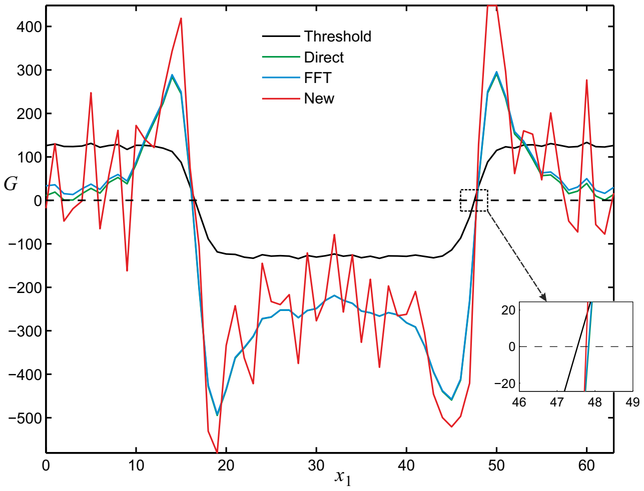


Figure 5. The G variation along the x_1 direction through the centre of the 3D image. Comparison of the proposed new method without corrections against other less efficient methods.

In Figure 6, the G values computed with the proposed method and corrections described in Section 3.1 are shown. As seen the oscillations are reduced but the results still not as good as the standard FFT based method. In Figure 7, the both corrections introduced in Sections 3.1 and 3.2 are employed along with the new method proposed. As seen the oscillations are vanished the the proposed method provides a result that is accurate and it is more than four times memory efficient and three times faster than standard FFT based method used in reference (Yeo et al. 2011).

The improvements in G calculations obtained for 3D images are directly extendable to 4D images as shown in Figures 8 and 9. In Figure 8 the spatial variation along the centre at $x_4 = 6$ (the physical time) are shown. As seen the proposed new method provides an acceptable and accurate spatial variation. Figure 9 shows the time (x_4) variation of $G(\boldsymbol{x})$ in a single spatial point $[x_1, x_2, x_3]$ selected in such a way that it is outside the object at different instances. As seen, the time variation of G is accuracy and discrepancy against the standard FFT method is within one time step.

The most important for segmentation are the discrepancies between surfaces $G(\boldsymbol{x}) = 0$. Regions where the geometrical potential passes zero value are specially zoomed in Figures 7–8. Form these

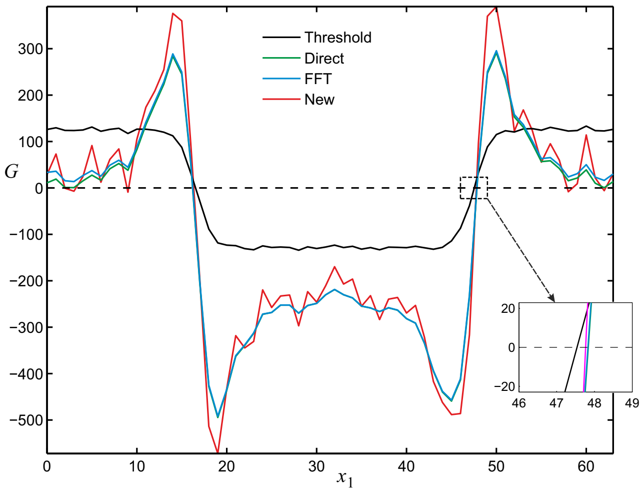


Figure 6. The G variation along the x_1 direction through the centre of the 3D image. Comparison of the proposed new method with correction described in Section 3.1 against other less efficient methods.

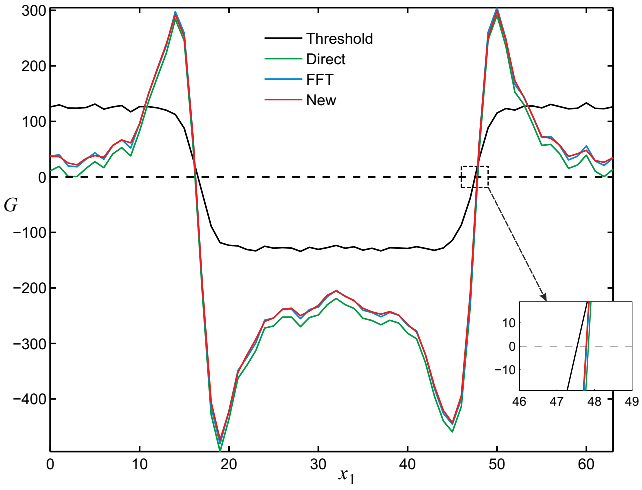


Figure 7. The G variation along the x_1 direction through the centre of the 3D image. Comparison of the proposed new method with both corrections described in Sections 3.1 and 3.2 against other less efficient methods.

figures it is seen that distances between surfaces $G(\boldsymbol{x}) = 0$ computed by old and the new methods are essentially less than voxel size that is absolutely satisfactory.

5. Segmentation of a 3D medical scan

Now that the accuracy, speed and memory efficiency of the proposed method is proved, further tests on real medical images are carried out in this and following subsection. In this section we provide the segmentation results for a 3D scan of a thoracic part of an aorta. The results are compared against the ones presented in (Yeo et al. 2011) to estimate approximate accuracy. The equation describing the evolution of the 3D level set function $\Phi(\boldsymbol{x}, t)$ and its solution are same as the ones presented in (Yeo et al. 2011). The numerical scheme used is described in detail in (Sazonov et al. 2011). In Figures 10 and 11 the stages of level set evolution are shown at $t = 0$ (initial surface was a ball inside the lower part of the aorta), $t = 100, 200$ and 260. The steady state is reached at $t = 260$ at which Φ change is negligibly small. In Figure 10 the standard method of FFT used in reference (Yeo et al. 2011) for calculation of G is used and Figure 11 is generated using the proposed new method with corrections described in Sections 3.1 and 3.2. As seen the proposed new method is as accurate as the original method but the cost is substantially lower. Analysis of non-coinciding

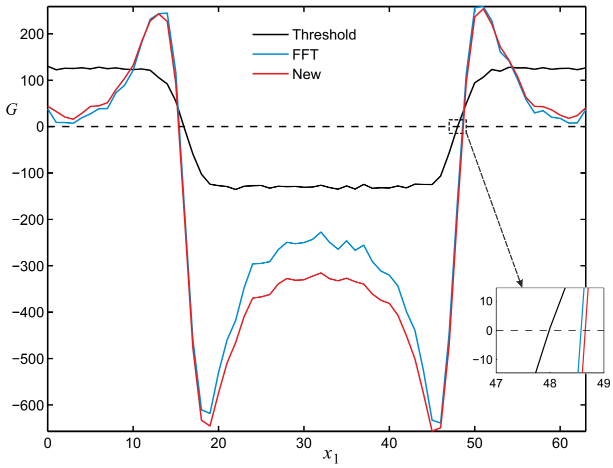


Figure 8. The G variation along the x_1 direction through the centre of the 4D image in the y, z and t directions.

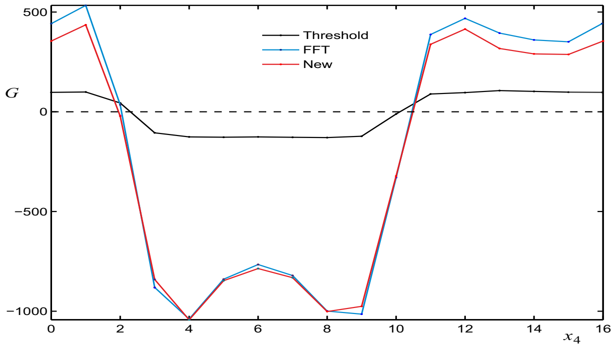


Figure 9. Time variation of the geometrical potential at a single spatial point.

voxels (background-image) in the objects segmented by the old and the new methods shows that their share is less than 1% in the segmented object, and the 'thickness' of the non-coinciding layer of voxel does not exceed one voxel).

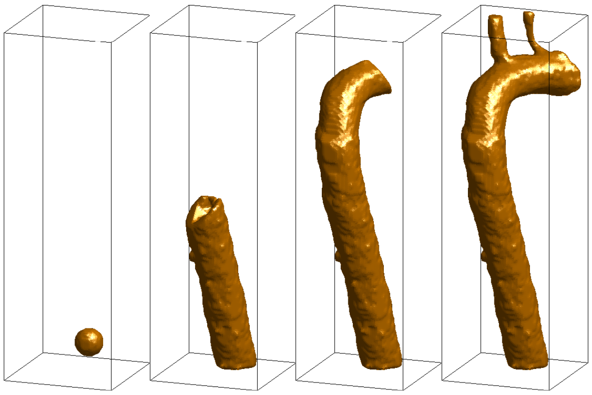


Figure 10. Segmentation of an aorta. Standard FFT of reference (Yeo et al. 2011) used in the G calculations.

6. Segmentation of a 4D medical scan

As mentioned previously, the proposed new method is more than four times memory efficient and three times faster than the original geometrical potential based segmentation method. Thus, it is

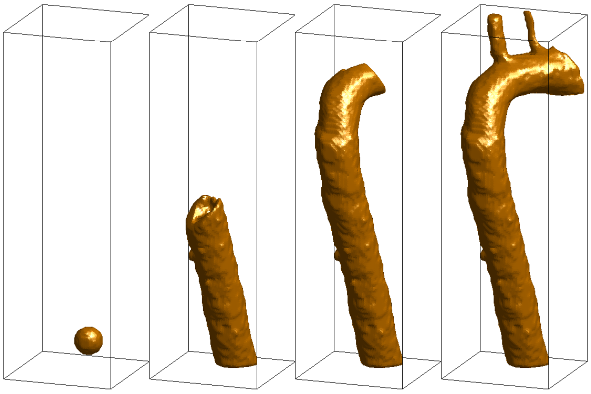


Figure 11. Segmentation of an aorta. Proposed new method used in the G calculations. .

now possible to segment much larger problems of practical interest. In this section, we provide a 4D problem that is often very difficult to segment using many of the existing methods. Although it is possible to segment the 4D image by dividing into 3D + 1D image, the proposed new method is able to handle 4D image as a single problem. This way, the topology of the object can be preserved.

The example of 4D object presented here is a 3D scan of a beating heart with 10 different instances in time. The internal part of the heart ventricle considered here has a complicated geometry due to trabeculaes. The geometry of trabeculaes is important as it can affect the flow parameters as it injects blood into the aorta or to the pulmonary vein. The size of the scan is $256 \times 256 \times 155 \times 10$. Since the number of frames in the time direction is only 10, discrete Fourier transform (DFT) is used here in stead of FFT. With smaller number of points FFT has no advantage over the DFT. The FFT is however used in all the three partial directions.

Note that coupling between spatial x_1, x_2, x_3 and temporal, x_4 coordinates occurs in computing $G(\boldsymbol{x})$ by use of a 4D kernel and also during the integration of PDE (2) for level set function $\Phi(\boldsymbol{x}, t)$ with respect to virtual time t.

The heart scan voxel size is $0.570313 \times 0.570313 \times 0.9$mm. The voxel grid is used as the grid for solving the PDE governing the level set functions. An important issue to be considered in 4D images is the relationship between spatial and temporal intervals. The ratio between the temporal and spatial steps should be finite. Unless this ratio is correctly selected the interaction between the spatial and temporal variation may be non-existent. A reasonable assumption is that the ratio between the time step and spatial step should be approximately 10 for the geometrical potential calculation. A large value of the order of 50 may be used when solving the PDE for the level set functions. In the problem considered, computation of the geometrical potential for the $256 \times 256 \times 155 \times 10$ image using the proposed method with both kernel corrections described in Section 3 needs 257s $\approx$ 4 min and 1.9G of memory. A solution to Eq. (2) for 2000 steps needs approximately two hours.

The initialisation of the level set function is selected in such a way that the active 4-surface represents a set of ellipsoids (balls in the grid coordinates) of different sizes but centered in the same point in every frame as shown in Figure 12 (left). The location is selected arbitrarily but the initial active surface is allowed to intersect the segmented object in some parts. As seen from Figure 12 (left), the initial surface is absent in some frames. This is done on purpose to show that active surfaces is moving from frame to frame during its evolution to the final state, i.e., the spatial and temporal variations are coupled. As seen the active surfaces penetrates empty frames gradually with time. The final segmented object is depicted in Figure 13 for all 10 frames. All the computations are carried out on Linux, Intel(R) Xeon 3.00GHz, RAM 4G (no parallelization or GPU are employed).

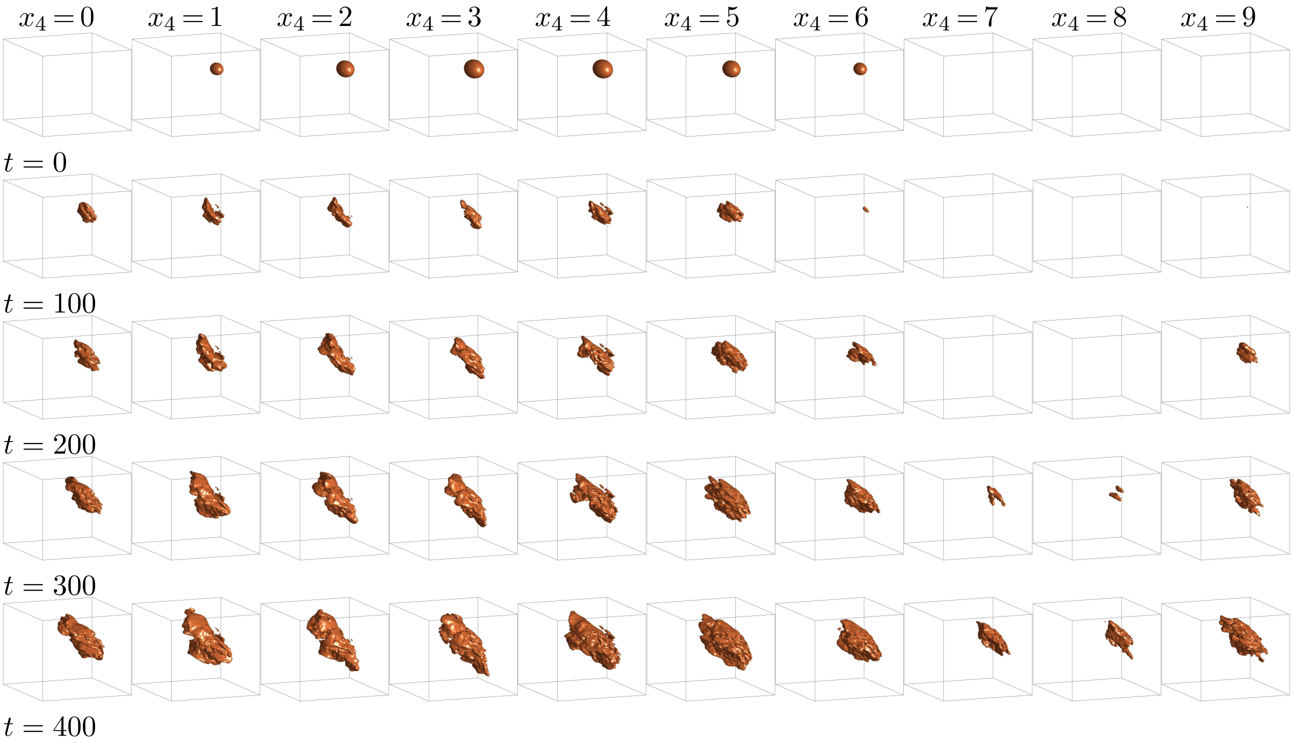


Figure 12. Earlier stages of integration (2) for the 4D segmentation of the heart.

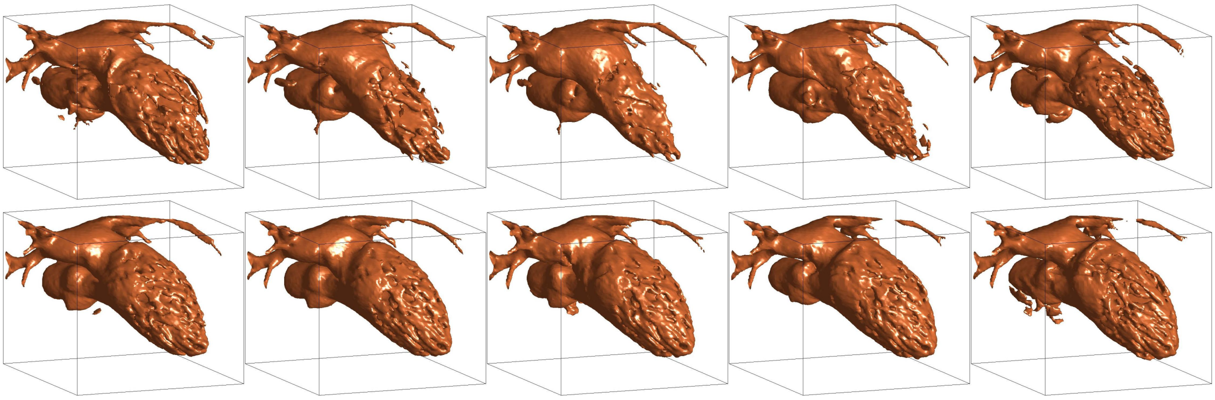


Figure 13. Final result of the 4D segmentation of a beating heart.

Conclusion

In this paper a set of substantial improvements to the GPF method ((Yeo et al. 2011)) is proposed. It is shown that upgraded version of GPF method does not decrease the quality of the segmentation but improves efficiency mainly via reduced memory requirement and CPU time. The improved version of the GPF method allows automatic segmentation of large 3D and 4D scans on standard personal computers. The Riesz transform provides a methods of rapidly computing the geometrical potential but leads to oscillatory solution. The oscillations are removed via effective correction methods. The final results save substantial amount of memory, up to four times, and CPU time, up to three times.

References

Aprovitola A, Gallo L. 2014. Edge and junction detection improvement using the Canny algorithm with a fourth order accurate derivative filter. In: Proc. The 10th Int. Conf. on Signal Image Technology & Internet Based Systems. SITIS'14, pp. 104–111, IEEE Computer Society

Arfken GB, Weber HJ. 1995. Mathematical methods for physicists. 4th ed. San Diego : Academic Press

Boykov Y, Funka-Lea G. 2006. Graph cuts and efficient N-D image segmentation, International Journal of Computer Vision (IJCV), 70(2):109–131.

Bremermann H. 1965. Distributions, Complex Variables and Fourier Transforms. Addison-Wesley Publiching Company, Inc., Reading, MA, U.S.A.

Caselles V, Kimmel R, Sapiro G. 1997 Geodesic active contour. Int. J. Comput. Vis. 22(1):61-79

Chan T., Vese L. 2001. Active contours without edges. IEEE Trans. Image Process. 10(2):266-277

Felsberg M, Sommer G. 2004. The monogenic scale-space: a unifying approach to phase-based image processing in scale-space. Journal of Mathematical Imaging and Vision, 21:5–26.

Fleureau J, Garreau M, Hernández AI, Simon A, Boulmier D. 2006. Multi-object and N-D segmentation of cardiac MSCT data using SVM classifiers and a connectivity algorithm. Computers in Cardiology 33:817–820.

Friedlander FG, Joshi M. 1998. Introduction to the Theory of Distributions, Cambridge Univeristy Press.

Gel'fand IM, Shilov GE. (Eugene Saletan, Transl.) 1964. Generalized Functions. Volume 1 Properties and Operations. Academic Press.

Hewitt E, Hewitt RE. 1979. The Gibbs-Wilbraham phenomenon: An episode in fourier analysis, in Archive for History of Exact Sciences 21(2):129–160.

Jones DS. 1982. The Theory of Generalised Functions. Second ed., Cambridge Univ. Press.

Kress R. 1989. Linear Integral Equations, Springer-Verlag, Berlin.

Lighthill MJ. 1964. Introduction to Fourier Analysis and Generalized Functions. Cambridge Univ. Press.

Malladi R, Sethian JA, Vemuri BC. 1995. Shape modelling withfront propagation: A level set approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(2):158-175

Noo F, Clackdoyle R, Pack JD. 2004. A two-step Hilbert transform method for 2D image reconstruction. Phys. Med. Biol. 49:3903-3923.

Sazonov I, Yeo SY; Bevan RLT, Xie X; van Loon R, Nithiarasu P. 2011. Modelling pipeline for subject-specific arterial blood flow-A review. Int. J. Numer. Meth. Biomed. Engng., 27(8):1167–1184.

Sazonov I, Xie X, Nithiarasu P. 2012. Efficient Geometrical Potential Force Computation for Deformable Model Segmentation. MCV:104-113

Schwartz L. 1954. Sur l'impossibilité de la multiplications des distributions. C. R. AcadṠci. Paris. 239: 847-848

Stein EM. 1970. Singular Integrals and Differentiability Properties of Functions. Princeton University Press, Princeton, New Jersey.

Stein EM, Weiss GL. 1971. Introduction to Fourier Analysis on Euclidean Spaces. Princeton University Press, Princeton, New Jersey.

Schneider CB. 1998. Inversion formulas for the discretized Hilbert transform on the unit circle. SIAM Journal on Numerical Analysis 35(1):71–77.

Vladimirov VS. 2002. Methods of the Theory of Generalized Functions. Taylor & Francis.

Wietzke L, Sommer G, Schmaltz C, Weickert J. 2008. Differential geometry of monogenic signal representations. In Robot Vision Volume 4931 of LNCS. ed: Sommer, G., Springer-Verlag, Berlin, Heidelberg, New York pp. 454-465.

Whitaker R. 2004. Modeling deformable surfaces with level sets. IEEE Computer Graphics and Applications. 24(5):6-9

Xiang Y, Chung A., Ye J. 2005. A new active contour method based on elastic interaction. In: Proc. IEEE Conf. Comput. Vis. Pattern. Recognit., pp. 452-457, IEEE Press, New York

Xie X, Mirmehdi M. 2008. MAC: Magnetostatic active contour model. IEEE Transactions on Pattern Analysis and Machine Intelligence. 30:632-646

Xu C, Prince JL. 1998. Snakes, shapes, and gradient vector flow. IEEE Trans. Image Process., 7(3):359369

Yang R, Mirmehdi M, Xie X, Hall D. 2013. Shape and appearance priors for Level Set-based LV segmentation, IET Journal of Computer Vision, 7(3):170–183.

Yeo SY, Xie X, Sazonov I, Nithiarasu P. 2011. Geometrically induced force interaction for three-dimensional deformable models. IEEE Transactions of Image Processing 20(5):1373–1387

Appendix A. Proof of proposition 2.1

Assume that the image occupy the total space $\mathbb{R}^n$, belongs to $\mathcal{D}$. We make the substitution $\boldsymbol{x} - \boldsymbol{x}' = \boldsymbol{x}''$ in integral (15):

$$\begin{aligned} G(\boldsymbol{x}) &= \lim_{\varepsilon \to 0^+} G_\varepsilon(\boldsymbol{x}) \quad \text{where} \\ G_\varepsilon(\boldsymbol{x}) &= c_n \int_{|\boldsymbol{x}''|>\varepsilon} \nabla I(\boldsymbol{x} - \boldsymbol{x}'') \cdot \frac{\boldsymbol{x}''}{|\boldsymbol{x}''|^{n+1}} \, \mathrm{d}^n \boldsymbol{x}''. \end{aligned} \tag{A1}$$

Now we introduce the n-spherical co-ordinates $\boldsymbol{x} = r\boldsymbol{\xi}(\boldsymbol{\theta})$ where $r = |\boldsymbol{x}|$, $\boldsymbol{\xi} = \boldsymbol{x}/r$. Angular coordinates θ_k form a $(n-1)$-vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{n-1}]$ where $\theta_k \in [0, \pi)$ for $k = 1, \ldots, n-2$ and $\theta_{n-1} \in [0, 2\pi)$. They are defined such that Cartesian components of vector $\boldsymbol{\xi}$ are

$$\begin{aligned} \xi_1 &= \cos\theta_1 \\ \xi_2 &= \sin\theta_1 \cos\theta_2 \\ \xi_3 &= \sin\theta_1 \sin\theta_2 \cos\theta_3 \\ &\cdots \\ \xi_{n-1} &= \sin\theta_1 \sin\theta_2 \cdots \sin\theta_{n-2} \cos\theta_{n-1} \\ \xi_n &= \sin\theta_1 \sin\theta_2 \cdots \sin\theta_{n-2} \sin\theta_{n-1}. \end{aligned}$$

We write the n-spherical volume element in the form

$$\begin{aligned} \mathrm{d}^n \boldsymbol{x} &= r^{n-1} \, \mathrm{d}r \; \mathrm{d}^{n-1}\boldsymbol{\theta} \\ \mathrm{d}^{n-1}\boldsymbol{\theta} &= \sin^{n-2}\theta_1 \sin^{n-3}\theta_2 \cdots \sin\theta_{n-2} \; \mathrm{d}\theta_1 \cdots \mathrm{d}\theta_{n-1}. \end{aligned}$$

Here $\mathrm{d}^{n-1}\boldsymbol{\theta}$ is the solid angle element, i.e. the area element of the unit n-sphere.

Passing to the spherical coordinates in integral (A1) we obtain

$$G_\varepsilon(\boldsymbol{x}) = c_n \oint \int_\varepsilon^{+\infty} \nabla I(\boldsymbol{x} - r''\boldsymbol{\xi}'') \cdot \frac{\boldsymbol{\xi}''}{r''} \mathrm{d}r'' \mathrm{d}^{n-1}\boldsymbol{\theta}''.$$

Observe that $\boldsymbol{\xi}''$ is the unit vector normal to the boundary of the domain of integration: $|\boldsymbol{x}''| > \varepsilon$. Therefore the dot product with this vector gives the radial component only, and we can substitute ∇I by $I_{r''} = (\partial/\partial r'')\, I$:

$$G_\varepsilon(\boldsymbol{x}) = c_n \oint \int_\varepsilon^{+\infty} I_{r''}(\boldsymbol{x} - r''\boldsymbol{\xi}'') \frac{1}{r''} \, \mathrm{d}r'' \mathrm{d}^{n-1}\boldsymbol{\theta}''.$$

Now we apply the integration by parts in the integral with respect to r'' representing the primitive as $\int I_{r''}(\boldsymbol{x} - r''\boldsymbol{\xi}'') \, \mathrm{d}r'' = I(\boldsymbol{x} - r''\boldsymbol{\xi}'') - I(\boldsymbol{x})$:

$$\begin{aligned} G_\varepsilon(\boldsymbol{x}) = c_n \frac{1}{\varepsilon} &\oint \left[I(\boldsymbol{x}) - I(\boldsymbol{x} - \varepsilon\boldsymbol{\xi}'') \right] \, \mathrm{d}^{n-1}\boldsymbol{\theta}'' \\ &+ c_n \oint \int_\varepsilon^{+\infty} \left[I(\boldsymbol{x} - r''\boldsymbol{\xi}'') - I(\boldsymbol{x}) \right] \frac{\mathrm{d}r''}{(r'')^2} \, \mathrm{d}^{n-1}\boldsymbol{\theta}''. \end{aligned}$$

For ε small enough we can expand $I(\boldsymbol{x} - \varepsilon\boldsymbol{\xi}'') - I(\boldsymbol{x})$ into Taylor's series in the vicinity of point $\boldsymbol{x}$:

$$I(\boldsymbol{x} - \varepsilon\boldsymbol{\xi}'') - I(\boldsymbol{x}) = \nabla I(\boldsymbol{x}) \cdot \varepsilon\boldsymbol{\xi}'' + O(\varepsilon^2).$$

Substituting the series into the first integral and integrating term-by-term we can write

$$G_\varepsilon(\boldsymbol{x}) = c_n\Big[-\nabla I(\boldsymbol{x})\cdot\oint \boldsymbol{\xi}''\,\mathrm{d}^{n-1}\boldsymbol{\theta}'' + O(\varepsilon)\Big] + c_n\oint\int_{\varepsilon}^{+\infty}\big[I(\boldsymbol{x}-r''\boldsymbol{\xi}'')-I(\boldsymbol{x})\big]\,\frac{\mathrm{d}r''}{(r'')^2}\;\mathrm{d}^{n-1}\boldsymbol{\theta}''.$$

Using the property of symmetry we can show that $\oint\,\boldsymbol{\xi}''\mathrm{d}^{n-1}\boldsymbol{\theta}''=0$. Returning to the Cartesian coordinates we have

$$G_\varepsilon(\boldsymbol{x}) = c_n\int_{|\boldsymbol{x}''|>\varepsilon}\frac{I(\boldsymbol{x}-\boldsymbol{x}'')-I(\boldsymbol{x})}{|\boldsymbol{x}''|^{n+1}}\;\mathrm{d}^n\boldsymbol{x}''+O(\varepsilon).$$

Passing to the limit $\varepsilon\to 0^+$ we have

$$G(\boldsymbol{x}) = c_n\text{p.v.}\int_{\boldsymbol{x}''\in\mathbb{R}^n}\frac{I(\boldsymbol{x}-\boldsymbol{x}'')-I(\boldsymbol{x})}{|\boldsymbol{x}''|^{n+1}}\;\mathrm{d}^n\boldsymbol{x}''. \qquad \text{(A2)}$$

■