B-Spline Explicit Active Surfaces: An efficient framework for real-time 3D region-based segmentation

Daniel Barbosa, T. Dietenbeck, Joel Schaerer, Jan d'Hooge, Denis Friboulet, Olivier Bernard

B-Spline Explicit Active Surfaces: An Efficient Framework for Real-Time 3-D Region-Based Segmentation

Daniel Barbosa, *Student Member, IEEE*, Thomas Dietenbeck, *Student Member, IEEE*, Joel Schaerer, Jan D'hooge, *Member, IEEE*, Denis Friboulet, *Member, IEEE*, and Olivier Bernard

***Abstract*—A new formulation of active contours based on explicit functions has been recently suggested. This novel framework allows real-time 3-D segmentation since it reduces the dimensionality of the segmentation problem. In this paper, we propose a B-spline formulation of this approach, which further improves the computational efficiency of the algorithm. We also show that this framework allows evolving the active contour using local region-based terms, thereby overcoming the limitations of the original method while preserving computational speed. The feasibility of real-time 3-D segmentation is demonstrated using simulated and medical data such as liver computer tomography and cardiac ultrasound images.**

***Index Terms*—Active contours, B-splines, image segmentation, real-time image processing, variational method.**

I. Introduction

SINCE their first introduction in [1], active contours have proven to be very powerful in the field of image processing [2]–[5]. In image segmentation, active contour-based methods correspond to a class of deformable models where the shape to be recovered is captured by propagating an evolving interface. The evolution of the interface is generally derived through a variational formulation: the segmentation problem is expressed as the minimization of an energy functional that reflects the properties of the objects to be recovered. In this energy functional, the terms corresponding to image features (usually called data attachment terms) may be broadly classified according two categories: edge-based or region-based terms. Edge-based active contour models use image gradients in order to identify object boundaries [6]. This type of approach is adequate in some situations, but is generally sensitive to image noise. In order to overcome these limitations, Chan and Vese introduced the concept of region-based active contours, where the properties of the regions inside and outside the object under segmentation drive the contour evolution [7]. This work yielded many extensions through different region-based energy functionals (based on mean separation [8], histogram separation [9], and Bayesian model [10]). Nevertheless, these approaches may fail when the region properties are computed in a global way (i.e., using the whole inner or outer regions), as soon as these features are not spatially invariant. In this context, Lankton and Tannenbaum have recently proposed a framework well suited to detect objects having heterogeneous properties by localizing the region-based approach [11]. Their model presents promising results in challenging images and clearly shows the advantages of localizing the energy functional while dealing with nonhomogeneous objects.

Active contours may be also categorized according to the representation of the evolving interface. With this respect, level-set-based methods have become one of the most popular approaches. This approach consists in representing the interface as the zero levet-set of a higher dimensional smooth function, which is usually called the level-set function. The underlying formulation allows working in a Eulerian framework and naturally deals with topology changes, making level-set methods able to segment multiple unconnected regions. However, the fact that levet-sets are built by adding a dimension to the original problem has a drawback: it increases the computational cost, making 3-D real-time applications difficult. In this context, Duan *et al.* have recently proposed to represent the interface using an explicit representation in order to tackle this computation burden problem [12], [13]. This approach was called active geometric functions (AGF). An immediate consequence of using such explicit formulation is the loss of topological flexibility, which is shown as the price to pay for increasing speed. Note, however, that this topological limitation is a mild constraint in many applications, such as medical imaging (i.e., when the goal is to segment one simply connected object), and thus where the topological flexibility of levet-sets is not desired or at least not always needed. The initial AGF framework has nevertheless left open some questions. First, localizing region-based data attachment terms is difficult to handle without *ad hoc* approximations. Furthermore, there is no explicit control on the

D. Barbosa is with the Laboratory on Cardiovascular Imaging and Dynamics, Katholieke Universiteit Leuven, 3000 Leuven, Belgium, and also with the CREATIS, INSA, UCB, CNRS UMR 5220, Inserm U630, 69621 Villeurbanne Cedex, France (e-mail: daniel.barbosa@uzleuven.be).

T. Dietenbeck, D. Friboulet, and O. Bernard are with the CREATIS, INSA, UCB, CNRS UMR 5220, Inserm U630s, 69621 Villeurbanne Cedex, France (e-mail: bernard@creatis.insa-lyon.fr).

J. Schaerer was with the CREATIS, INSA, UCB, CNRS UMR 5220, Inserm U630s, 69621 Villeurbanne Cedex, France. He is now with BioClinica, 69008 Lyon, France.

J. D'hooge is with the Laboratory on Cardiovascular Imaging and Dynamics, Katholieke Universiteit Leuven, 3000 Leuven, Belgium (e-mail: jan.dhooge@uzleuven.be).

interface smoothness, which is simply done through undersampling of the surface function.

Starting from the framework developed by Duan *et al.* [12], [13], we have expanded it to a B-spline formulation that we will denote as B-spline explicit active surfaces (BEAS) in the sequel. The derived model has the following conceptual advantages.

1) By formally relating the explicit and implicit formulation of the region-based evolution terms, it expands the framework of Duan *et al.* and allows using local or global region-based energy initially designed in the level-set framework.
2) It allows explicitly controlling the smoothness of the interface through the scale factor of the underlying B-spline formulation.
3) These properties are obtained while preserving real-time performance for 3-D data due to the fact that the active contour evolution can be expressed as a succession of simple 1-D convolutions.

Moreover, thanks to the continuous B-spline formulation, the analytic computation of differential properties-related quantities of the interface (i.e., defining the normal direction and local curvature) is straightforward.

This paper is structured as follows. In Section II, we focus on the general formulation of image segmentation problems using B-splines and derive the minimization of the resulting functional in terms of the B-spline coefficients. In particular, we show that both global and localized region-based approaches can be used within the proposed framework. In Section III, we discuss the implementation issues of our method. In particular, we present an efficient implementation, which allows performing region-based segmentation in real-time. In Section IV, we evaluate the performance of the method using both simulated and real images. We give the main conclusions and perspectives of this paper in Section V.

II. Methodology

A. General Setting

The key idea of the framework is to model the interface Γ as an explicit function. Geometrically, this implies that one of the coordinates of the points of the interface is expressed as a function of the remaining coordinates. Let us note $\mathbf{x} \in \mathbb{R}^n$ a point of coordinates $\{x_1, \ldots, x_n\}$ in an n-dimensional space and $\mathbf{x}^* \in \mathbb{R}^{n-1}$ a point of coordinates $\{x_2, \ldots, x_n\}$ in the associated $(n-1)$-dimensional subspace. Formally, an explicit function can then be defined as

$$\psi : \mathbb{R}^{n-1} \mapsto \mathbb{R}, \quad \psi(\mathbf{x}^*) = x_1. \tag{1}$$

Let us define the associated function in $\mathbb{R}^n$ as

$$\phi(\mathbf{x}) = \psi(\mathbf{x}^*) - x_1. \tag{2}$$

Using these definitions, the interface can either be shown as an explicit surface or as the zero level of the higher dimensional implicit function ϕ, i.e.,

$$\Gamma = \begin{cases} \mathbf{x} \in \mathbb{R}^n | \psi(\mathbf{x}^*) = x_1, & (3) \\ \mathbf{x} \in \mathbb{R}^n | \phi(x_1, \cdots, x_n) = 0. & (4) \end{cases}$$

Obviously, the explicit form of Γ (3) intrinsically limits the topology, which does not happen in the original level-set framework. However, this can be also shown as introducing prior information in the model since the segmented object will always have the topology induced by the chosen coordinate system.

Let us now consider a function $f(\mathbf{x})$ in $\mathbb{R}^n$, and let us note $\bar{f}$ as the restriction of f over interface Γ, i.e.,

$$\bar{f}(\mathbf{x}^*) = f\left(\psi(\mathbf{x}^*), \mathbf{x}^*\right). \tag{5}$$

Let us moreover consider a functional depending on $f(\mathbf{x})$ and evaluated over the interface Γ. Using the above definition and the properties of the 1-D delta Dirac function $\delta(\cdot)$, we have

$$\int_{\Omega} f(\mathbf{x}) \delta_\phi(\mathbf{x}) d\mathbf{x} = \int_{\Gamma} \bar{f}(\mathbf{x}^*) d\mathbf{x}^* \tag{6}$$

where Γ is an open bounded subset of $\mathbb{R}^n$, $\delta_\phi(\mathbf{x}) = \delta(\phi(\mathbf{x}))$, $d\mathbf{x} = dx_1 \cdots dx_n$, and $d\mathbf{x}^* = dx_2 \cdots dx_n$.

This expression thus relates the explicit and implicit formulations of the functional. As shown in the sequel, this property will allow using the region-based terms initially designed for levet-sets in the explicit framework.

B. B-Spline Representation of the Explicit Function

Inspired by the recent work proposed by Bernard *et al.* in [14], we propose to express the explicit functions as the linear combination of B-spline basis functions [15], i.e.,

$$x_1 = \psi(x_2, \cdots, x_n) = \sum_{\mathbf{k} \in \mathbb{Z}^{n-1}} c[\mathbf{k}] \beta^d \left(\frac{\mathbf{x}^*}{h} - \mathbf{k} \right) \tag{7}$$

where $\mathbf{x}^* \in \mathbb{R}^{n-1}$ is a point of coordinates $\{x_2, \ldots, x_n\}$ and $\beta^d(\cdot)$ is the uniform symmetric $n-1$-dimensional B-spline of degree d. This function is separable and is built as the product of $n-1$ 1-D B-splines so that $\beta^d(\mathbf{x}^*) = \prod_{j=2}^{n} \beta^d(x_j)$. The knots of the B-splines are located on a rectangular grid defined on the chosen coordinate system, with a regular spacing given by h. The coefficients of the B-spline representation are gathered in $c[\mathbf{k}]$.

C. Global Region-Based Energy Formulation

We now turn back to the problem of segmenting one object from the background of image $I(\mathbf{x})$ using region-based terms. In the level-set framework, a general expression of the energy functional to be minimized can be formulated as [16]

$$E_G = \int_{\Omega} F\left(\mathbf{x}, H_\phi(\mathbf{x})\right) d\mathbf{x} \tag{8}$$

where F is given by

$$F\left(\mathbf{x}, H_\phi(\mathbf{x})\right) = f_{\text{in}}\left(I(\mathbf{x})\right) H_\phi(\mathbf{x}) + f_{\text{out}}\left(I(\mathbf{x})\right)\left(1 - H_\phi(\mathbf{x})\right) \tag{9}$$

and $H_\phi(\mathbf{x}) = H(\phi(\mathbf{x}))$, where H is the Heaviside function.

In (9), f_{in} and f_{out} provide energy criteria attached to the inside and outside regions delimited by interface Γ, respectively. In the sequel, the smoothness of the interface will be implicitly enforced through the underlying B-spline representation, and therefore, the energy functional given in (9) does not include

the usual regularization term. As an example, for the classical Chan–Vese functional [7], we have

$$\begin{cases} f_{\text{in}}\left(I(\mathbf{x})\right) = \left(I(\mathbf{x}) - u\right)^2, \\ f_{\text{out}}\left(I(\mathbf{x})\right) = \left(I(\mathbf{x}) - v\right)^2 \end{cases} \tag{10}$$

where u and v are the mean image value inside and outside the interface.

In a classical active contour framework, calculus of variations is employed to minimize the energy criterion. In contrast with these approaches, we make use of our formulation and perform the minimization of the energy with respect to B-spline coefficients $c[\mathbf{k}]$. Such minimization implies computing the derivatives of (8) with respect to each B-spline coefficient $c[\mathbf{k_i}]$. We show in Appendix A that these derivatives may be simply expressed as

$$\frac{\partial E_G}{\partial c[\mathbf{k_i}]} = \int_{\Gamma} \bar{g}(\mathbf{x}^*)\beta^d\left(\frac{\mathbf{x}^*}{h} - \mathbf{k}_i\right) d\mathbf{x}^* \tag{11}$$

where

$$\bar{g}(\mathbf{x}^*) = \bar{f}_{\text{in}}\left(\bar{I}(\mathbf{x}^*)\right) - \bar{f}_{\text{out}}\left(\bar{I}(\mathbf{x}^*)\right). \tag{12}$$

$\bar{g}(\mathbf{x}^*)$ thus reflects the features of the object to be segmented and is evaluated over interface Γ. $\bar{g}$ will be called the feature function in the sequel. For clarity sake, $\bar{I}(\mathbf{x}^*)$ corresponds to the image value at position $\mathbf{x} = \{\psi(\mathbf{x}^*), x_2, \ldots, x_n\}$. Equation (11) yields an interesting interpretation of the minimization process. Let us define $\beta_h^d(\mathbf{x}^*) = \beta^d(\mathbf{x}^*/h)$, which is the B-spline of degree d upscaled by factor h. The expression of the energy gradient is then given by

$$\boldsymbol{\nabla}_c E_G = \frac{\partial E_G}{\partial c[\mathbf{k}]} = \int_{\Gamma} \bar{g}(\mathbf{x}^*)\beta_h^d(\mathbf{x}^* - h\mathbf{k})d\mathbf{x}^*. \tag{13}$$

This last equation shows that the computation of the gradient of E_G with respect to a set of B-spline coefficients may be interpreted as convolving feature function $\bar{g}(\mathbf{x}^*)$ with B-spline $\beta_h^d(\mathbf{x}^*)$ and sampling the result with period h.

D. Localized Region-Based Energy Formulation

As previously mentioned, Lankton and Tannenbaum have recently introduced a mathematical framework for localizing region-based energies in level-set segmentation methods, overcoming the problems associated with global segmentation functionals. These localization strategies can be adapted to our framework. First, we consider the general localized region-based energy functional proposed in [11] as

$$E_L = \int_{\Omega} \delta_\phi(\mathbf{x}) \int_{\Omega} B(\mathbf{x},\mathbf{y}) F_L\left(\mathbf{y}, H_\phi(\mathbf{y})\right) d\mathbf{y} d\mathbf{x} \tag{14}$$

where $F_L(\mathbf{y}, H_\phi(\mathbf{y}))$ is an image criteria and $B(\mathbf{x},\mathbf{y})$ corresponds to a mask function in which the local parameters that drive the evolution of the interface are estimated. Energy criteria F_L is still given by (9) above; however, its parameters now have to be locally estimated at each $\mathbf{x}$ in neighborhood B. Using again the Chan–Vese functional as an example, the energy criteria is still given through (10), where parameters u and v are now replaced by their localized version, i.e., u_x and v_x, which can be written as

$$\begin{cases} u_x = \frac{\int_\Omega B(\mathbf{x},\mathbf{y}) H_\phi(\mathbf{y}) I(\mathbf{y}) d\mathbf{y}}{\int_\Omega B(\mathbf{x},\mathbf{y}) H_\phi(\mathbf{y}) d\mathbf{y}}, \\ v_x = \frac{\int_\Omega B(\mathbf{x},\mathbf{y}) (1 - H_\phi(\mathbf{y})) I(\mathbf{y}) d\mathbf{y}}{\int_\Omega B(\mathbf{x},\mathbf{y}) (1 - H_\phi(\mathbf{y})) d\mathbf{y}}. \end{cases} \tag{15}$$

The original method of Lankton and Tannenbaum suggests the use of a radial mask around interface point $\mathbf{x}$ under evaluation. However, in order to maintain low computation costs, we propose in this paper to restrict the region defined by B to the set of points belonging to the normal direction of the interface at point $\mathbf{x}$ and whose distance to $\mathbf{x}$ is smaller than a parameter ρ. Let us note that this framework has been recently proposed and studied by Mille [17]. His results show that this does not negatively affect the segmentation results, as compared with the original formulation.

The corresponding mask function $B(\mathbf{x},\mathbf{y})$ can be defined as

$$B(\mathbf{x},\mathbf{y}) = \begin{cases} 1, & \text{if } \mathbf{y} = \mathbf{x} + k \times \hat{\mathbf{N}}, k \in [-\rho, \rho] \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

where $\hat{\mathbf{N}}$ is the normal vector to the interface at position $\mathbf{x}$ and ρ the radius of the local neighborhood.

We show in Appendix B that the minimization of (14) with respect to B-spline coefficients $c[\mathbf{k_i}]$ leads to the following expression:

$$\frac{\partial E_L}{\partial c[\mathbf{k_i}]} = \int_{\Gamma} \bar{g}_L(\mathbf{x}^*)\beta^d\left(\frac{\mathbf{x}^*}{h} - \mathbf{k}_i\right) d\mathbf{x}^* \tag{17}$$

where $\bar{g}_L(\mathbf{x}^*)$ reflects the features of the object to be segmented. Finally, the expression of the localized region-based energy gradient is thus given by

$$\boldsymbol{\nabla}_c E_L = \frac{\partial E_L}{\partial c[\mathbf{k}]} = \int_{\Gamma} \bar{g}_L(\mathbf{x}^*)\beta_h^d\left(\mathbf{x}^* - h\mathbf{k}\right) d\mathbf{x}^*. \tag{18}$$

As for the global region-based case, the computation of the gradient of E_L with respect to a set of B-spline coefficients may be thus interpreted as convolving feature function $\bar{g}_L(\mathbf{x}^*)$ with B-spline $\beta_h^d(\mathbf{x}^*)$ and sampling the result with period h.

In this paper, we will use a localized formulation of the Chan–Vese functional to produce the results given in Section IV. In this case, $\bar{g}_L(\mathbf{x}^*) = (\bar{I}(\mathbf{x}) - u_x)^2 - (\bar{I}(\mathbf{x}) - v_x)^2$, and expression (17) may be written as

$$\frac{\partial E_L}{\partial c[\mathbf{k_i}]} = \int_{\Gamma} \left(\left(\bar{I}(\mathbf{x}^*) - u_x\right)^2 - \left(\bar{I}(\mathbf{x}^*) - v_x\right)^2\right)\beta^d\left(\frac{\mathbf{x}^*}{h} - \mathbf{k}_i\right) d\mathbf{x}^* \tag{19}$$

where u_x and v_x now represent the intensity means in the interior and exterior of the evolving interface localized by B at point $\mathbf{x}$.

In the same way, other classical energy functional such as the one proposed by Yezzi *et al.* in [8] or Michailovich *et al.* in [9] can be easily formulated into localized versions and integrated within our B-spline explicit framework.

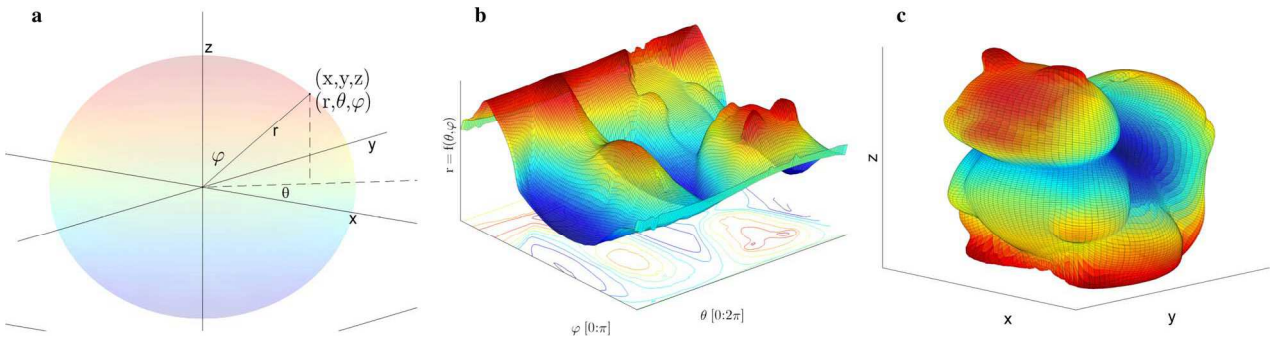

Fig. 1. Illustrative 3-D surface compact representation through an explicit function in the spherical domain. (a) Correspondence between the Cartesian and spherical domains. (b) Explicit function in the spherical domain. (c) Explicit function converted to the Cartesian space.

III. Implementation

A. Coordinate System

The choice of the coordinate system should be done according to the topology of the object to be segmented. For instance, taking the simple example of a closed 3-D object, a possible choice could be the use of an explicit surface defined in spherical coordinates. In such case, the radius of the points within the boundary surface would be given as a function of both the azimuth and zenith angles. Obviously, this formulation implies that all the boundary points can be "seen" from the origin of the coordinate system. This limits the degree of complexity of the shapes that can be accurately represented by such an explicit function. On the other hand, the dimensionality reduction is quite obvious since a 3-D surface can be compactly represented by a 2-D function defined on a rectangular grid in the spherical domain. This is shown in the example presented in Fig. 1. Naturally, the choice of an appropriate coordinate system should be accompanied with the proper definition of the origin since the representation of some shapes might be impossible otherwise.

B. Problem Discretization

In order to make it computationally tractable, the segmentation problem has to be discretized. Indeed, the image information is available only on a discrete grid, which implies the discretization of the chosen feature function $\bar{g}(\mathbf{x}^*)$. Furthermore, the number of points within the interface needs to be defined in order to discretize convolution integrals (11) and (18) driving the interface evolution. We assume here that the explicit surface is defined over an $n-1$-dimensional rectangular grid in the chosen coordinate system. This grid is then discretized in a uniform way using N_b points. Two main factors drive the choice of number of discretization points:

1) The complexity of shape to be recovered: N_b should be indeed large enough to capture the level of shape detail required for the application;
2) The complexity of the image properties: whenever the images features are spatially inhomogeneous, it is desirable to use a discretization grid which guarantees that the image information is extracted in a dense manner for the interface evolution.

Taking the examples shown in the sequel, 1024 points (32×32 grid) were used to segment the relatively simple shapes corresponding to 3-D liver tumors and left ventricles (see Figs. 6 and 7), whereas 16,384 points (128×128 grid) were needed to segment the more complex shape corresponding to the 3-D squirrel (see Fig. 5).

C. Surface Evolution

In order to minimize the energy criterion, a modified version of the gradient descent with feedback step adjustment algorithm was used [18]. The B-spline coefficients will evolve as

$$c[\mathbf{k}]^{(t+1)} = c[\mathbf{k}]^{(t)} + \lambda \frac{\partial E}{\partial c[\mathbf{k}]^{(t)}}. \tag{20}$$

At each step, the energy associated with the update of the B-spline coefficients to $c[\mathbf{k}]^{(t+1)}$ is calculated. If this leads to a decrease in the energy criterion, the update is considered successful and B-spline coefficients $c[\mathbf{k}]^{(t+1)}$ are kept to the next iteration and step size λ is multiplied by factor $\alpha_f \geq 1$. If not, $c[\mathbf{k}]^{(t+1)}$ are still kept, but the step size of the next iteration will be more conservative by dividing the step size by $\eta_f \geq 1$. If a new energy minimum is not found in a few iterations, the algorithm exits, and the last energy minimum is taken as the segmentation result. This way, the algorithm is still able to escape small local minima and better converge to the optimal solution.

D. Algorithm Implementation

In order to provide an overview of the key processing blocks of the proposed method and to potentiate its reproducibility, the different steps of the algorithm are shown in Table I.

For the sake of clarity, λ is the update step, as defined in (20), whereas $\mathrm{Its}_{\mathrm{Threshold}}$ corresponds to the number of iterations that the algorithm is allowed to continue without finding a new energy optimum.

E. Computational Cost

The complexity of our algorithm depends on the support of the discrete B-spline kernel. More specifically, it essentially depends on its chosen degree d. The support of a 1-D discrete B-spline of degree d expanded by factor h is given by $(d+1)h$. Calling n the dimensionality of the image data and considering that the object contour can be effectively represented by an $n-1$-dimensional explicit surface discretized over N^{n-1} points, the number of spline knots is then $(N/h)^{n-1}$. The cost of

TABLE I
BEAS ALGORITHM IMPLEMENTATION

Initialize	• Define an appropriate coordinate system (e.g.: a polar for a 2D closed contour) • Define the origin of the chosen coordinate system • Initialize $\psi(\mathbf{x}^*)$ and sample the surface at the chosen discretization grid • Initialize final segmentation result $\psi_{final}(\mathbf{x}^*)$=$\psi(\mathbf{x}^*)$ • Set the gradient descent parameters:$\lambda_{Initial}$, η_f and $Its_{Threshold}$
Run	**While** (Its<Its_{Max} and $counter$<$Its_{Threshold}$) • Estimate image feature for the current interface position, $\bar{g}(\mathbf{x}^*)$, from the set of discretized points • Compute energy gradient, $\nabla_c E$, with respect to the B-spline coefficients $c[\mathbf{k_i}]$, as shown in (13) • Update B-spline coefficients $c[\mathbf{k_i}]$, as shown in (20) • Re-sample $\psi(\mathbf{x}^*)$ to get the new discretized interface points • **If** the update leads to a decrease in the energy criteria ◦ $counter = 0$ ◦ $\lambda = \lambda_{Initial}$ ◦ $\psi_{final}(\mathbf{x}^*) = \psi(\mathbf{x}^*)$ • **Else** ◦ $counter = counter + 1$ ◦ $\lambda = \lambda \div \eta_f$ • **End If** • $Its = Its + 1$ **End While**

computing one update (20) of the B-spline coefficients $c[\mathbf{k}]^{(t+1)}$ is dominated by convolution (13). Because of separability, the downsampling can be applied on the fly, and the overall cost for updating all coefficients in (20) is

$$K_{\text{exp}} = (d+1)N^{n-1}\frac{(1-h^{n-1})}{(1-h^{-1})}. \tag{21}$$

IV. EXPERIMENTS

A. Segmentation Parameters

In order to evaluate the proposed algorithm, several experiments were carried out on both simulated and real data. The following settings are applied to all experiments.

1) We use a cubic B-spline function as basis for the BEAS representation. This function provides a good tradeoff between smoothing properties and computational cost.
2) Except otherwise mentioned, the segmentation functional is the localized Chan–Vese functional.
3) The parameters that adjust the steps of the gradient descent are fixed as $\lambda = \lambda_{\text{initial}} = 1$, $\alpha_f = 1$, and $\eta_f = 1.1$.

The number of points used to discretize the interface, in polar (2-D experiments) or spherical coordinates (3-D experiments), as well as the scale parameters, which are denoted by N_b and h, respectively, are specified for each result. The size of neighborhood $B(x, y)$, i.e., ρ, is also indicated when the local version of the algorithm is used.

We also give the CPU time of the computation, performed on a 2.8-GHz Core Duo laptop, with 3-GB random access memory running Ubuntu. Except otherwise mentioned, these CPU times refer to a C++ implementation.

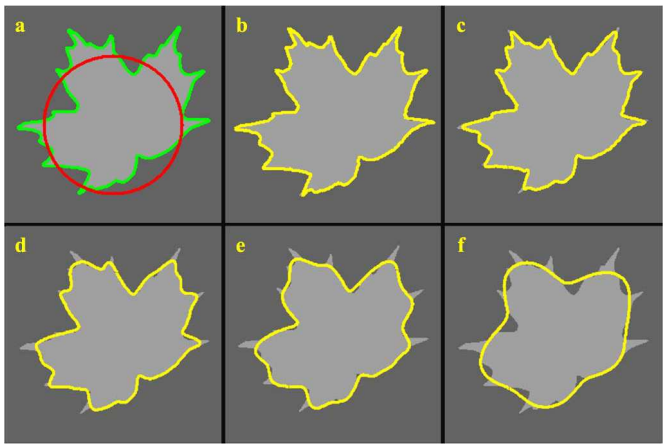

Fig. 2. Influence of the scale factor on the smoothness of the final segmentation result. (a) Ground truth (contour in green and initialization in red). (b)–(f) Segmentation results using the (yellow contour) proposed BEAS approach for varying scales (1, 2, 4, 8, and 16, respectively). N_b is set to 256 for this experiment.

B. Interface Smoothness Control

The first experiment aims at illustrating the intrinsic smoothness control through the scale factor in the B-spline kernel. To this end, the proposed method was applied to a binary leaf image containing various levels of details using the global version of the approach, as given in (13), and the Chan–Vese functional. As shown in Fig. 2, the increase in the scale parameter clearly controls the degree of smoothness of the curve, acting, therefore, in a similar way as curvature-related penalties.

C. BEAS Versus Original AGF Formulation

In order to demonstrate the computational advantages of the use of B-spline basis, instead of the ones used in the original AGF framework, we used the same example image as Duan *et al.* [12]. This test image is made of two regions having different standard deviation separated by a sinusoidal boundary. The segmentation energy used by Duan *et al.* is based on this standard deviation difference and is given as

$$\begin{cases} f_{\text{in}} = (\sigma(\mathbf{x}) - s)^2 \\ f_{\text{out}} = (\sigma(\mathbf{x}) - t)^2 \end{cases} \tag{22}$$

where $\sigma(\mathbf{x})$ is the local standard deviation computed in a 7×7 neighborhood around $\mathbf{x}$ and s and t are the average standard deviation of the inside and outside regions. The original AGF were tested using two different expression of the explicit contour (i.e., a sine function or a cubic Hermite polynomial) and two different initializations (i.e., close and far from the solution).

In order to provide a consistent comparison with the original AGF framework, we performed the segmentation using the global version of our approach using the same functional given in (22). The segmentation results are shown in Fig. 3. Table II provides the related quantitative results in terms of accuracy (measured as the root mean square error) and computation times. Note that, in this particular experiment, all computation times correspond to a simple MATLAB implementation of our approach in order to provide a consistent comparison with

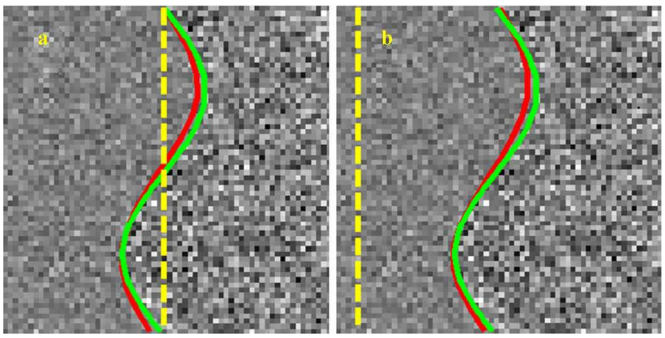

Fig. 3. Segmentation of a simulated image using the BEAS algorithm. (a) Result using a close initialization. (b) Result using a far initialization. (Green) The true boundary, (yellow) the initial contour, and (red) the segmentation result ($N_b = 16, h = 4$).

TABLE II
COMPARISON BETWEEN BEAS AND ORIGINAL AGF FRAMEWORK

Surface basis	**Cubic B-Spline**		**Sine Function [12]**		**Cubic Hermite [12]**
Initialization	Close	Far	Close	Far	Not referred
Number of iterations	25	58	10	25	Not referred
Total CPU time (ms)	32.7	104.0	210	425	3590
RMSE (px.)	1.22	1.56	1.26	1.40	1.66

the MATLAB implementation used in [12]. Note that no C++ routines are called in our MATLAB code.

Table II shows that the proposed B-spline explicit framework yields a computational speedup when compared with the original formulation while presenting comparable accuracy. This is related with the convolution-based evolution of the segmentation energy in the B-spline framework. In particular, even when the AGF uses a sine function as the explicit contour, which implies a strong *a priori*, the B-spline formulation presents smaller computational burden.

D. B-Spline Explicit Active Surface Versus Fast Level-Sets

In order to show the computation speed of the proposed approach, the BEAS algorithm was compared with a state-of-the-art approximation of level-set-based segmentation method for real-time segmentation, i.e., the so-called FTC algorithm recently proposed by Shi and Karl [19]. In the FTC, the level-set evolution is separated into two cycles: 1) one cycle for the data-dependent term and 2) a second cycle for the smoothness regularization. The speed of the FTC algorithm then results from two main features: 1) It uses a discrete approximation of a narrow-band levet-set that yields evolution without the need of solving partial differential equations, and 2) the smoothing curvature term is approximated by Gaussian filtering of the interface. The FTC algorithm is controlled via four parameters: 1) the number of iterations of the data-dependent cycle N_a; 2) the number of iterations of the smoothing cycle N_s; 3) the variance of the Gaussian filter σ_g^2; and 4) the number of points used to numerically approximate filter N_g.

In order to evaluate the methods, two simulated data sets corresponding to the 3-D squirrel shape were generated. The first data set is simply a 3-D binary image of the squirrel, whereas the second corresponds to an inhomogeneous version of the

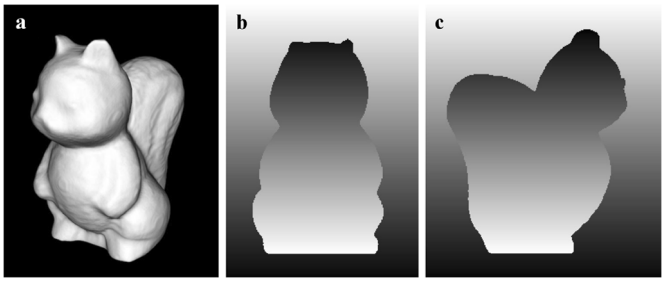

Fig. 4. Segmentation of the 3-D squirrel data set. (a) Three-dimensional rendering of the binary data. (b) XZ and (c) YZ slices through the inhomogeneous 3-D data set.

squirrel, obtained by linearly varying the gray levels of the object and the background in the z-direction, as shown in Fig. 4. The dimensions of these two data sets are $296 \times 215 \times 296$.

The parameters of the two approaches were set as follows. Concerning the FTC, parameters N_a, N_s, σ_g^2, and N_g were set to 30, 3, 1, and 3, respectively, as in the original paper [19]. The proposed BEAS approach was applied using the global energy criterion and the scale set to one ($h = 2$), and the interface was evolved through two strategies: 1) direct evolution, i.e., the explicit surface was modeled using a 128×128 points grid and the segmentation was performed by directly evolving this surface, and 2) two-steps evolution, i.e., a fast and approximate solution was first obtained using a 32×32 grid. This solution was then upsampled and used as the initialization to the evolution of the final 128×128 grid.

Both methods were first applied to the binary squirrel using the same threshold-based functional described in [19], and the same initialization (a box around the squirrel) was used. Fig. 5(c)–(d) display the obtained segmentations and qualitatively show a very close agreement between the FTC and the proposed approach. Table III (first three columns) shows the performances of the two methods, in terms of accuracy (measured as the Dice coefficient) and computation times. The Dice coefficients confirm the agreement between FTC and BEAS (0.9878 for the direct evolution and 0.9863 for the two-step evolution). Table III also indicates that the proposed BEAS approach yields significantly reduced computation times, as compared with FTC (speed up factor of about 25 and 46 for the direct evolution and two-step evolution, respectively), allowing real-time 3-D segmentation.

In order to test the influence of the region term localization in terms of accuracy and speed, the BEAS approach, equipped with the local Chan–Vese functional (see Section II-D), was then applied to the inhomogeneous squirrel ($\rho = 7$). Only the two-step evolution was tested since it provides more robustness regarding the initialization and reduces the computational burden of the method. Fig. 5(e) displays the obtained segmentation, and Table III (last column) shows the resulting performances. These results show that the agreement with the reference is in the same order as in the above global case. The inclusion of the local Chan–Vese functional has a significant impact on the computational times (0.636 s for the two-step evolution). Nevertheless, it allows segmenting this challenging data while still being almost five times faster than the FTC

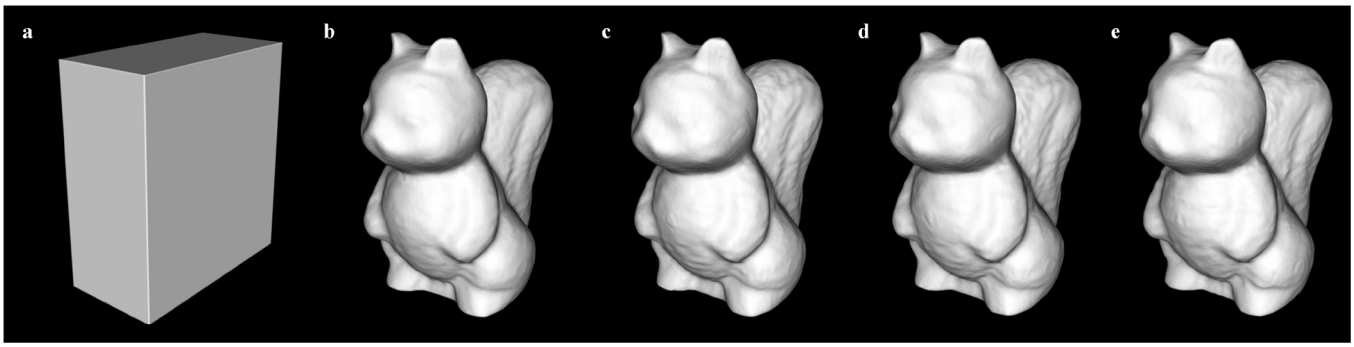

Fig. 5. Segmentation results of the 3-D squirrel data. (a) Initialization. (b) FTC [binary + threshold functional]. (c) BEAS [binary + threshold functional + direct evolution]. (d) BEAS [binary data+threshold functional+two-step evolution]. (e) BEAS [inhomogeneous+local ChanVese functional+two-step evolution]. $h = 2$ for all the BEAS results.

TABLE III
COMPARISON OF SEGMENTATION PERFORMANCE BETWEEN BEAS AND FTC

Algorithm	**BEAS Direct Evolution**	**BEAS 2 steps Evolution**	**FTC [19]**	**BEAS 2 steps Evolution**
Functional	Threshold	Threshold	Threshold	Local Chan-Vese
Data	Reference	Reference	Reference	Inhomogeneous
Dice	0.9878	0.9863	1 (Reference)	0.9875
CPU time (s)	0.118	0.067	3.1	0.636

method applied on homogeneous data and using a much simpler threshold-based functional. It is important to stress that for this case, the segmentation is not truly real-time. Nonetheless, this is a very challenging case that requires a large sampling grid to capture all the shape details. In real-life applications, as the ones presented in the next two examples, the shape of the objects to recover are simpler, and thus, smaller sampling grids may be used, which allows us to achieve accurate 3-D segmentation results in real-time.

Overall, these results thus show that the proposed approach has a significantly lower computation burden than the fastest approximation of the level-set-based method while having simultaneously accurate segmentation results. Note, however, that these features are obtained in part by limiting the admissible surfaces to starlike shapes.

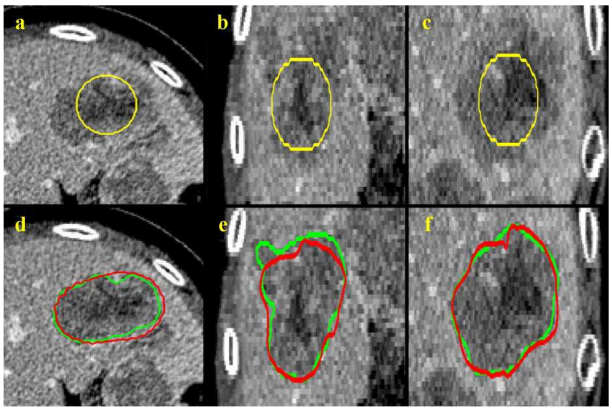

Fig. 6. Example of a liver tumor segmented with the proposed method. Initialization in the (a) axial, (b) coronal, and (c) sagital planes. (d)–(f) Segmentation results. (Red) BEAS. (Green) Manual reference.

E. Segmentation Performance in Clinical Data

Here, the segmentation of liver tumors in computer tomography (CT) data is used as a clinical example of another possible application of the proposed framework. Because these tumors generally have different shapes and intensities, the segmentation is not straightforward [20]. The gray values of a tumor depend on several properties, such as the type of tumor, the delay between the contrast injection and the image acquisition, the contrast dose, and the patient physiology. Therefore, a simplified energy functional as the one originally proposed in the work of Duan *et al.* will not be suited for this segmentation task. The energy functional used for this example was the localized Chan–Vese functional given in (19).

The proposed algorithm was evaluated in ten tumors from four patients, used as training data of the "3-D Segmentation in the Clinic: A Grand Challenge II" in MICCAI2008 [21]. The liver tumor data of the MICCAI2008 challenge were acquired on one 64-slice and two 40-slice CT scanners using a standard four-phase contrast-enhanced imaging protocol with slice thickness of 1–1.5 mm and an in-plane resolution of 0.6–0.9 mm. The initialization was taken as a sphere centered in the tumor, with a user-defined radius. This information was extracted from two clicks from the user. Given the significant variations in tumor size, radius ρ for the local parameter estimation was defined as 1/3 of the radius of the initial sphere. Regarding the surface evolution strategy, it was chosen to simply evolve it with the aforementioned direct evolution approach, using a 32×32 grid. B-spline scale parameter h was set to 4.

The proposed BEAS approach is compared with the state-of-the-art liver tumor segmentation method by Smeets *et al.* [20]. Note that this algorithm was the one that performed best on the MICCAI2008 challenge. It is important to stress that the amount of user input is similar to the proposed method (two user clicks).

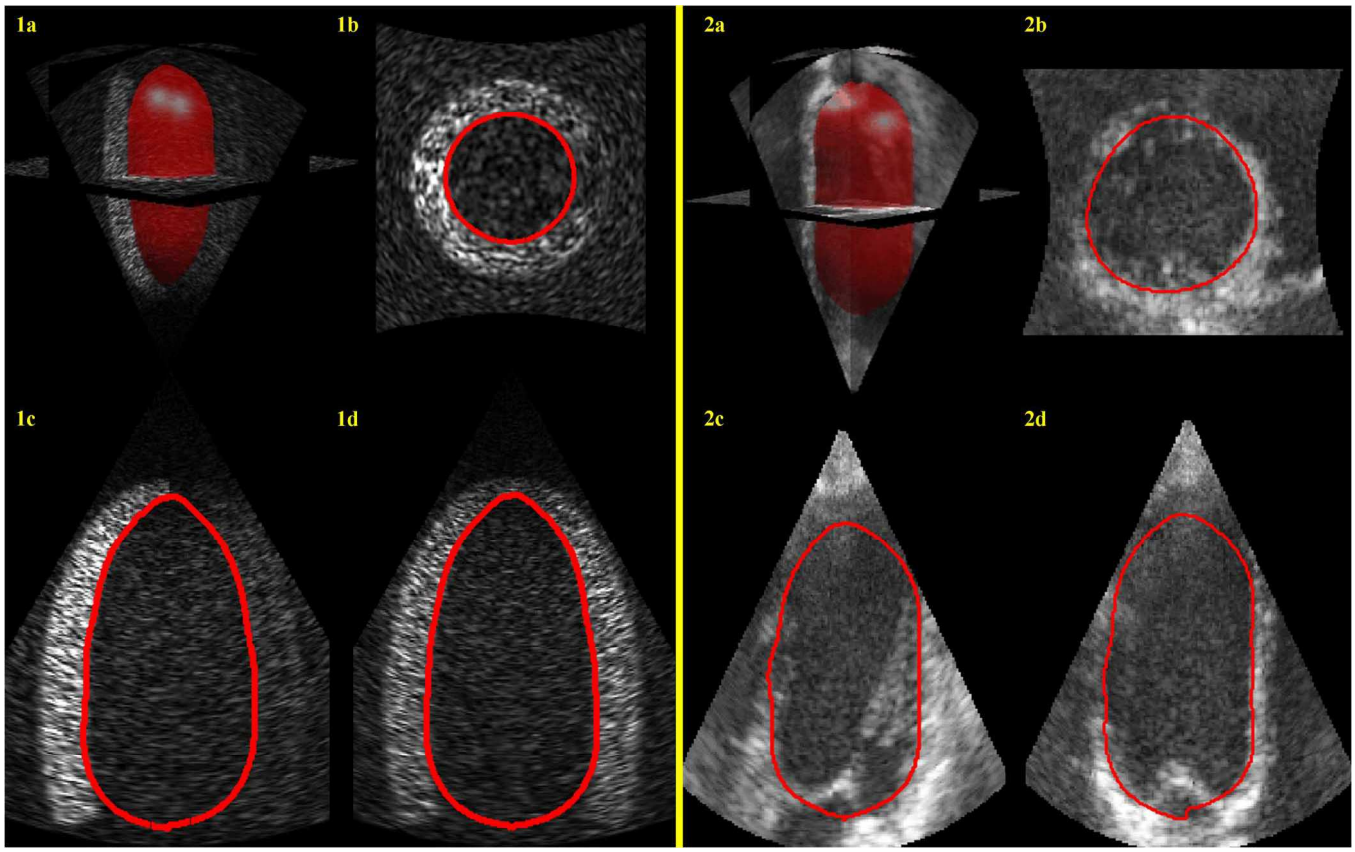

Fig. 7. Segmentation results for (1) simulated and (2) real 4-D ultrasound data [end-diastolic phase: (a) triplane view + 3-D rendering, (b) short-axis view, and (c)–(d) long axes views].

TABLE IV
BEAS SEGMENTATION PERFORMANCE ON LIVER TUMORS

	BEAS	**Method proposed in [20]**
Overlap error (%)	23.9	23.6
Volume Difference (%)	4.4	7.4
Mean Distance (mm)	1.6	1.2
RMS Distance (mm)	2.1	2.4
Maximum Distance (mm)	6.4	7.3
CPU time (min-max, s)	0.025-0.06	20-120

One example of a tumor segmented with the proposed algorithm is shown in Fig. 6. The quantitative performances of the two algorithms are presented in Table IV. It may be observed that the accuracy of the proposed algorithm is comparable to the algorithm proposed by Smeets *et al.* [20]. However, the BEAS framework has a significantly smaller computational burden since it took, on average, 0.035 s to converge to the segmentation solution, whereas the algorithm proposed in [20] yields computation times ranging from 20 s up to 2 minutes. Note that both algorithms use C++ implementations.

F. Tracking Performance in Clinical Data

One of the main advantages of the low computational burden of the proposed method is the possibility to employ it to segment temporal sequences. The original framework of Duan *et al.* [12], [13] focused on the segmentation of 4-D echocardiographic data. Here, we thus perform a feasibility study on the performance of the proposed method in a simulated inhomogeneous 4-D volume and a clinically recorded data set of regular image quality.

The BEAS method was initialized with a small ellipsoid, which was scaled and centered in the left ventricular cavity with user input. Note that this simple initialization can be easily substituted by an automatic approach, as the one proposed in [22]. The proposed algorithm segmented each frame. Moreover, the segmentation obtained for the current frame is used as the initialization for the next frame. The parameters used for all experiments were $N_b = 32 \times 32$, $h = 4$, and $\rho = 15$.

The first example is a 4-D realistic data set of the left ventricle, simulated using COLE, a research ultrasound simulator available at the Medical Imaging Research Center of the Katholieke Universiteit Leuven. The size of the simulated volumetric data is $282 \times 282 \times 282$, with an isotropic voxel size of 0.46 mm. Further details on the data simulation parameters and on the simulator algorithm can be found in [23] and [24]. In the present simulation, the contrast between the myocardium and the blood pool varies within the wall, introducing therefore inhomogeneities in the data. Such type of artifacts are very common in 3-D chocardiographic data in clinical practice [25].

The second example consists in a clinically recorded data set of regular image quality. This data set was acquired from a healthy volunteer using a GE Vivid 7 system equipped with a 3-V probe, using ECG gating over six cardiac cycles during a single breath hold, resulting in an acquisition rate of 31.5 frames per second.

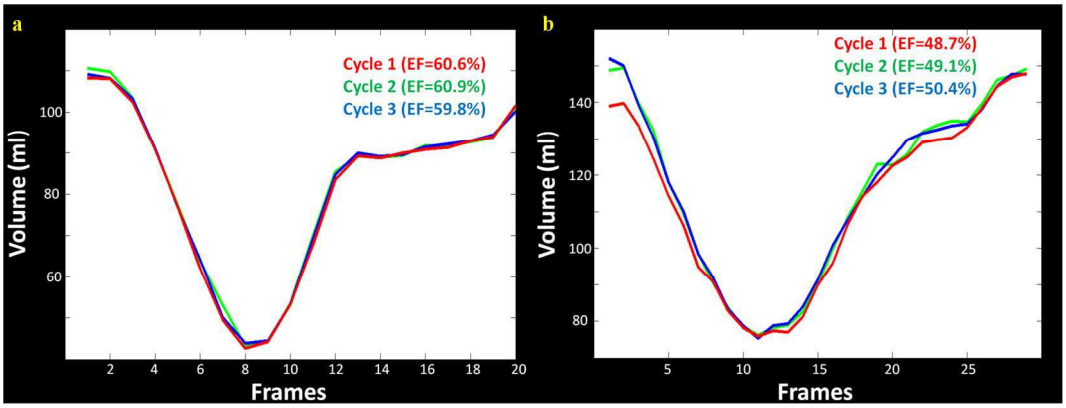

Fig. 8. Volume curves extracted from the tracking using the proposed method. (a) Simulated data. (b) Real data.

The segmentation results obtained for the simulated and clinical data at end-diastole are shown in Fig. 7.

On average, the proposed method was able to converge to the final result in 18.2 iterations for the simulated data and in 9.8 iterations for the real data, for each frame. The corresponding average computation times were 25 ms for the simulated data and 12.5 ms for the clinical data, thus surpassing the available 4-D echocardiography image acquisition rate, which was 31.5 ms per frame using ECG gating. The videos with the tracking are available in the multimedia materials available with the online version of this paper.

In order to assess the reproducibility of the segmentation, the method was applied to three consecutive cardiac cycles and the left ventricular volumes computed for each frame. The resulting temporal volume evolution is shown in Fig. 8. In both cases, there is good agreement in the volume curves of the tracking in the subsequent cycles. Furthermore, all volume traces present a correct physiologic shape, where the different phases of the cardiac cycle can be easily identified.

As a quantitative measure of the quality of the tracking, the ejection fraction (EF) values (EF $=$ (EDV$-$ESV)/EDV, where ESV is the end-sytolic volume and EDV is the end-diastolic volume) was used since it is the main cardiac index extracted from volume curves. The reference EF value for the simulated data set is 60%. As it is shown in Fig. 8, the EF values extracted from the volume curves for the simulated data set are in the range 59.8%–60.9% and are thus very close to the reference value. For the clinical case, the extracted EF is within the expected range for a healthy young subject.

V. Conclusion

We have introduced in the present paper an extension to the original AGF framework introduced by Duan *et al.*, further expanding the computational efficiency due to a B-spline formulation, which allows expressing the interface evolution as fast separable 1-D convolutions. The time saved in the interface evolution can be then used to include more advanced segmentation energies, which can be analytically integrated into the explicit framework, given its intrinsic mathematical link with level-set methods. It is to be noted that these features are obtained in part by limiting the topology and shape of the interface. In particular, this limitation makes the proposed approach adapted to medical image segmentation tasks, where the objects to be recovered have often only one connected component. This is the case of the examples presented in this paper (3-D echocardiographic data and CT liver tumor images) and could be easily extended to other image modalities and organs. Moreover, the ability of the proposed framework to efficiently deal with challenging 3-D segmentation in real-time makes it particularly relevant in clinical segmentation tasks, where such feature is often desirable to provide a smooth workflow for the physician.

Appendix A
Energy Derivation for General Region-Based Forces

We consider here the differentiation of global region-based energy criterion E_G (8) with respect to a given B-spline coefficient $c[\mathbf{k_i}]$. For brevity sake, let us skip the function arguments and assume that f_{in} and f_{out} do not explicitly depend on ϕ. Using differentiation with respect to parameter $c[\mathbf{k_i}]$, we have

$$\frac{\partial E_G}{\partial c[\mathbf{k_i}]} = \int_{\Omega} \frac{\partial F\left(\mathbf{x}, H_\phi(\mathbf{x})\right)}{\partial \phi} \frac{\partial \phi}{\partial c[\mathbf{k_i}]} d\mathbf{x}. \tag{23}$$

The derivative of region term $F(\mathbf{x}, H_\phi(\mathbf{x}))$ with respect to ϕ may be expressed as

$$\frac{\partial F\left(\mathbf{x}, H_\phi(\mathbf{x})\right)}{\partial \phi} = g(\mathbf{x})\delta_\phi(\mathbf{x}) \tag{24}$$

where $g(\mathbf{x}) = f_{\text{in}}(\mathbf{x}) - f_{\text{out}}(\mathbf{x})$. Noting moreover that, from (2) and (7), we have

$$\frac{\partial \phi}{\partial c[\mathbf{k_i}]} = \beta^d \left(\frac{\mathbf{x}^*}{h} - \mathbf{k}_i \right) \tag{25}$$

and using the general relation between levet-set and AGF given in (6), we finally obtain (11) and (12).

Appendix B
Energy Derivation for Localized Region-Based Forces

We consider here the differentiation of general localized region-based energy criterion E_L (14) with respect to a given

B-spline coefficient $c[\mathbf{k_i}]$. Using differentiation with respect to parameter $c[\mathbf{k_i}]$, we have

$$\frac{\partial E_L}{\partial c[\mathbf{k_i}]} = \int_\Omega \delta_\phi(\mathbf{x}) \left[\int_\Omega B(\mathbf{x},\mathbf{y}) \frac{\partial F\left(\mathbf{y}, H_\phi(\mathbf{y})\right)}{\partial \phi} \frac{\partial \phi(\mathbf{y})}{\partial c[\mathbf{k_i}]} d\mathbf{y} \right] d\mathbf{x} + \int_\Omega \delta'_\phi(\mathbf{x}) \frac{\partial \phi(\mathbf{x})}{\partial c[\mathbf{k_i}]} \left[\int_\Omega B(\mathbf{x},\mathbf{y}) F\left(\mathbf{y}, H_\phi(\mathbf{y})\right) d\mathbf{y} \right] d\mathbf{x}. \quad (26)$$

As suggested in [11], $\delta'_\phi(\cdot)$ evaluates to zero near the zero levet-set (i.e., the evolving interface Γ). As such, it does not affect the movement of the interface, and we ignore this term. Equation (26) thus becomes

$$\frac{\partial E_L}{\partial c[\mathbf{k_i}]} = \int_\Omega \delta_\phi(\mathbf{x}) \left[\int_\Omega B(\mathbf{x},\mathbf{y}) \frac{\partial F\left(\mathbf{y}, H_\phi(\mathbf{y})\right)}{\partial \phi} \frac{\partial \phi(\mathbf{y})}{\partial c[\mathbf{k_i}]} d\mathbf{y} \right] d\mathbf{x}. \quad (27)$$

Noting that the derivative of the local term may be written as

$$\frac{\partial F\left(\mathbf{y}, H_\phi(\mathbf{y})\right)}{\partial \phi} = g_L(\mathbf{y}) \delta_\phi(\mathbf{y}) \quad (28)$$

we obtain

$$\frac{\partial E_L}{\partial c[\mathbf{k_i}]} = \int_\Omega \delta_\phi(\mathbf{x}) \left[\int_\Omega B(\mathbf{x},\mathbf{y}) \delta_\phi(\mathbf{y}) g_L(\mathbf{y}) \frac{\partial \phi(\mathbf{y})}{\partial c[\mathbf{k_i}]} d\mathbf{y} \right] d\mathbf{x}. \quad (29)$$

Noting that $B(\mathbf{x},\mathbf{y})\delta_\phi(\mathbf{y})$ is only different than zero if $\mathbf{y} \equiv \mathbf{x}$, we have

$$B(\mathbf{x},\mathbf{y})\delta_\phi(\mathbf{y}) = \delta(\mathbf{y} - \mathbf{x}). \quad (30)$$

As a consequence, the inner integral can be simplified, leading to the following expression:

$$\frac{\partial E_L}{\partial c[\mathbf{k_i}]} = \int_\Omega g_L(\mathbf{x}) \delta_\phi(\mathbf{x}) \frac{\partial \phi(\mathbf{x})}{\partial c[\mathbf{k_i}]} d\mathbf{x}. \quad (31)$$

Now getting use of (6) and (25), the previous equation yields equation (17).

ACKNOWLEDGMENT

The authors would like to thank to D. Smeets for providing access to the challenge data, as well as to his own algorithm implementation. He also provided a common platform to evaluate both algorithms in order to ensure an accurate comparison of the segmentation metrics.

REFERENCES

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vis.*, vol. 1, no. 4, pp. 321–331, Jan. 1988.

[2] A. Blake and M. Isard, *Active Contours*. Cambridge, MA: Springer-Verlag, 1998.

[3] N. Paragios and R. Deriche, "Geodesic active contours and levet-sets for the detection and tracking of moving objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 3, pp. 266–280, Mar. 2000.

[4] T. Zhang and D. Freedman, "Tracking objects using density matching and shape priors," in *Proc. Int. Conf. Comput. Vis.*, 2004, pp. 1950–1954.

[5] N. Paragios, Y. Chen, and O. Faugeras, *Handbook of Mathematical Models in Computer Vision*. New York: Springer-Verlag, 2005.

[6] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *Int. J. Comput. Vis.*, vol. 22, no. 1, pp. 61–79, Feb. 1997.

[7] T. Chan and L. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, Feb. 2001.

[8] J. A. Yezzi, A. Tsai, and A. Willsky, "A fully global approach to image segmentation via coupled curve evolution equations," *J. Vis. Commun. Image Represent.*, vol. 13, no. 1/2, pp. 195–216, Mar. 2002.

[9] O. Michailovich, Y. Rathi, and A. Tannenbaum, "Image segmentation using active contours driven by the Bhattacharyya gradient flow," *IEEE Trans. Image Process.*, vol. 16, no. 11, pp. 2787–2801, Nov. 2007.

[10] M. Rousson and R. Deriche, "A variational framework for active and adaptative segmentation of vector valued images," in *Proc. Motion Video Comput.*, 2002, pp. 56–61.

[11] S. Lankton and A. Tannenbaum, "Localizing region-based active contours," *IEEE Trans. Image Process.*, vol. 17, no. 11, pp. 2029–2039, Nov. 2008.

[12] Q. Duan, E. D. Angelini, and A. F. Laine, "Surface function actives," *J. Vis. Commun. Image Represent.*, vol. 20, no. 7, pp. 478–490, Oct. 2009.

[13] Q. Duan, E. D. Angelini, and A. F. Laine, "Real-time segmentation by active geometric functions," *Comput. Methods Programs Biomed.*, vol. 98, no. 3, pp. 223–230, Jun. 2010.

[14] O. Bernard, D. Friboulet, P. Thévenaz, and M. Unser, "Variational B-spline level-set: A linear filtering approach for fast deformable model evolution," *IEEE Trans. Image Process.*, vol. 18, no. 6, pp. 1179–1191, Jun. 2009.

[15] M. Unser, "Splines: A perfect fit for signal and image processing," *IEEE Signal Process. Mag.*, vol. 16, no. 6, pp. 22–38, Nov. 1999.

[16] S. Jehan-Besson, M. Barlaud, and G. Aubert, "DREAM2S: Deformable regions driven by an Eulerian accurate minimization method for image and video segmentation," *Int. J. Comput. Vis.*, vol. 53, no. 1, pp. 45–70, Jun. 2003.

[17] J. Mille, "Narrow band region-based active contours and surfaces for 2D and 3D segmentation," *Comput. Vis. Image Understand.*, vol. 113, no. 9, pp. 946–965, Sep. 2009.

[18] J. Kybic and M. Unser, "Fast parametric elastic image registration," *IEEE Trans. Image Process.*, vol. 12, no. 11, pp. 1427–1442, Nov. 2003.

[19] Y. Shi and W. C. Karl, "A real-time algorithm for the approximation of level-set-based curve evolution," *IEEE Trans. Image Process.*, vol. 17, no. 5, pp. 645–656, May 2008.

[20] D. Smeets, D. Loeckx, B. Stijnen, B. De Dobbelaer, D. Vandermeulen, and P. Suetens, "Semi-automatic levet-set segmentation of liver tumors combining a spiral scanning technique with supervised fuzzy pixel classification," *Med. Image Anal.*, vol. 14, no. 1, pp. 13–20, Feb. 2010.

[21] X. Deng and G. Du, "Editorial: 3D segmentation in the clinic: A grand challenge II—Liver tumor segmentation," in *Proc. Int. Conf. MICCAI*, 2008.

[22] M. van Stralen, K. Leung, M. Voormolen, N. de Jong, A. van der Steen, J. Reiber, and J. Bosch, "Time continuous detection of the left ventricular long axis and the mitral valve plane in 3-D echocardiography," *Ultrasound Med. Biol.*, vol. 34, no. 2, pp. 196–207, Feb. 2008.

[23] A. Elen, H. F. Choi, D. Loeckx, H. Gao, P. Claus, P. Suetens, F. Maes, and J. D'hooge, "Three-dimensional cardiac strain estimation using spatio-temporal elastic registration of ultrasound images: A feasibility study," *IEEE Trans. Med. Imag.*, vol. 27, no. 11, pp. 1580–1591, Nov. 2008.

[24] H. Gao, H. F. Choi, P. Claus, S. Boonen, S. Jaecques, G. H. van Lenthe, G. Van Der Perre, W. Lauriks, and J. D'hooge, "A fast convolution-based methodology to simulate 2-D/3-D cardiac ultrasound images," *IEEE Trans. Ultrason., Ferroelectr., Freq. Control*, vol. 56, no. 2, pp. 404–409, Feb. 2009.

[25] J. A. Noble and D. Boukerroui, "Ultrasound image segmentation: A survey," *IEEE Trans. Med. Imag.*, vol. 25, no. 8, pp. 987–1010, Aug. 2006.