Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2007, Article ID 76241, 12 pages
doi:10.1155/2007/76241

Research Article

Locally Regularized Smoothing B-Snake

Jérôme Velut, Hugues Benoit-Cattin, and Christophe Odet

CREATIS, CNRS UMR 5220, Inserm U 630, INSA, Bâtiment Blaise Pascal, 69621 Villeurbanne, France

Received 22 July 2005; Revised 25 July 2006; Accepted 17 December 2006

Recommended by Jiri Jan

We propose a locally regularized snake based on smoothing-spline filtering. The proposed algorithm associates a regularization process with a force equilibrium scheme leading the snake's deformation. In this algorithm, the regularization is implemented with a smoothing of the deformation forces. The regularization level is controlled through a unique parameter that can vary along the contour. It provides a locally regularized smoothing B-snake that offers a powerful framework to introduce prior knowledge. We illustrate the snake behavior on synthetic and real images, with global and local regularization.

1. INTRODUCTION

Active contour models (or snakes) are well adapted for edge detection and segmentation. Since snakes were introduced by Kass et al. [1], they have been widely used in many domains and improved using different contour representations and deformation algorithms. Menet et al. [2] proposed the B-snakes that take advantages of the B-spline representation. A local control of the curve continuity and a limited number of processed points increase the convergence speed and the segmentation reliability. At the same time, L. Cohen and I. Cohen [3] focused on external forces that drive the snake toward the features of interest in the image and proposed the balloon force that increases considerably the attainability zone. Then, Xu and Prince [4] defined another external force called gradient vector flow (GVF) that brings a better control on the deformation directions: they proposed to diffuse the gradient over the image according to optical flow theory. Beside these works, the multiresolution frameworks were integrated within the active contours. Wang et al. [5] used a B-spline representation that allows a coarse-to-fine evolution of the snake. Brigger et al. [6, 7] extended Wang's technique with a multiscale approach in both the image and the parametric contour domain. Precioso et al. [8] proposed a region-based active contour that achieves real-time computation adapted to video segmentation. They extended their model by applying a smoothing B-spline filter [9, 10] on the contour. It increases considerably the robustness to noise without additional computation. Recently, new energies have been proposed by Jacob et al. [11] who unify the edge-based scheme with the region-based one.

Existing snakes suffer from several limitations when a local regularization is wanted. With the original snake [1], a local regularization involves a matrix inversion step at each iteration. Although B-snakes [6] avoid this inversion step by implicitizing the internal energy, the proposed solutions induce a varying sampling step when a local regularization of the snake is needed. Consequently, prior knowledge would be difficult to integrate in the varying sampling step. The smoothing B-spline filtering method of [8] does not deal with local regularization and has a strong initialization-dependent minimization process linked to the regularization algorithm proposed.

In this paper, we propose to regularize locally a snake while keeping a uniform sampling step. The presented approach is based on smoothing-spline filtering that is controlled through a unique parameter λ. The next section reminds the snake concepts and their interaction with B-splines. Section 3 details the proposed algorithm named LRSB-snake that stands for locally regularized smoothing B-snake. Section 4 presents experimental results on real and synthetic images.

2. SNAKES AND B-SPLINES

First, we remind the original active contour of Kass et al. [1] and its minimization procedure. Then, we present the B-spline snake method and its evolutions. Afterward, we

describe the smoothing B-spline filtering strategy. Finally, we analyze its usage within the smoothing-spline snake-based algorithm of Precioso et al. [8].

2.1. Snakes: active contour model

Basically, a snake is a parametric curve $g(s) = (x(s), y(s))$ placed on an image [1]. The final snake, that represents the segmentation result, will be the curve that minimizes the energy E_{snake} given by

$$E_{\text{snake}} = \int_s E_{\text{int}}(g(s)) + E_{\text{ext}}(g(s))\,ds, \tag{1}$$

where

$$E_{\text{int}} = \frac{1}{2}\left(\alpha(s)\cdot\left|\frac{dg(s)}{ds}\right|^2 + \beta(s)\cdot\left|\frac{d^2g(s)}{ds^2}\right|^2\right), \tag{2}$$

$$E_{\text{ext}} = -\left|\nabla I(x,y)\right|^2. \tag{3}$$

E_{int} given by (2) is the internal energy that traduces shape constraints on the curve. The $\alpha(s)$ and $\beta(s)$ functions tune the regularization. The function $\beta(s)$ gives more or less importance to the curvature by weighting the second derivative of the curve. In the same way, the function $\alpha(s)$ weights the elasticity through a tuning of the first derivative. In order to attract the snake to the contours of an image I, the external energy E_{ext} is defined in (3). Such energy is the most current one. One can use any external energy expressions coherent with the image features to detect.

In [1], the authors complete the minimization of the E_{snake} energy using the discretized version of (1):

$$E_{\text{snake}} = \sum_k E_{\text{int}}(g(k)) + E_{\text{ext}}(g(k)), \tag{4}$$

where k is the discretized version of the parameter s. This leads to a force balance

$$A\cdot\mathbf{x} + \mathbf{f}_x = 0, \qquad A\cdot\mathbf{y} + \mathbf{f}_y = 0, \tag{5}$$

where A is a pentadiagonal banded matrix built from $\alpha(k)$ and $\beta(k)$ function values, where vectors $\mathbf{x}$ and $\mathbf{y}$ contain the point coordinates of the discrete version $g(k)$ of the curve $g(s)$, and where vectors $\mathbf{f}_x$ and $\mathbf{f}_y$ constitute the external forces computed at the kth point of the snake as follows:

$$f(k) = (f_x(k), f_y(k)) = \left(\frac{\partial E_{\text{ext}}(k)}{\partial x}, \frac{\partial E_{\text{ext}}(k)}{\partial y}\right), \tag{6}$$

where $(x(k), y(k)) = g(k)$.

The equilibrium state is reached using the gradient descent method that ensures the convergence toward a local minimum of energy:

$$\begin{aligned} A\cdot\mathbf{x}_i + \mathbf{f}_{x,i-1} &= -\gamma(\mathbf{x}_i - \mathbf{x}_{i-1}),\\ A\cdot\mathbf{y}_i + \mathbf{f}_{y,i-1} &= -\gamma(\mathbf{y}_i - \mathbf{y}_{i-1}), \end{aligned} \tag{7}$$

where γ is a step-size parameter and i is the iteration index of the gradient descent evolution.

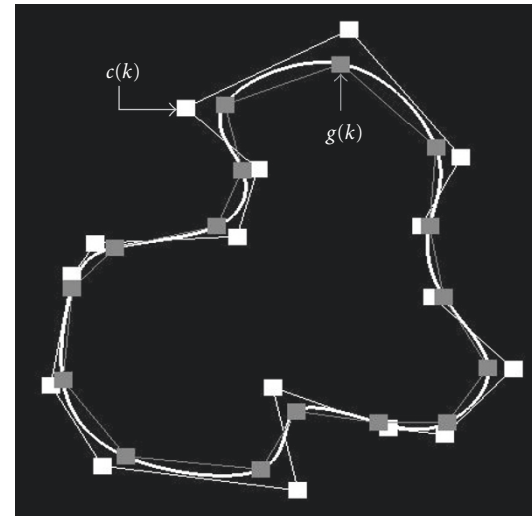


Figure 1: B-spline interpolation. $c(k)$ are the coefficients, $g(k)$ are the B-spline points, and $g(s)$ is the curve that interpolates $g(k)$.

We can then solve for $\mathbf{x}_i$ and $\mathbf{y}_i$ with $\mathbf{x}_{i-1}$ and $\mathbf{y}_{i-1}$:

$$\begin{aligned} \mathbf{x}_i &= (A+\gamma I)^{-1}(\gamma\cdot\mathbf{x}_{i-1} - \mathbf{f}_{x,i-1}),\\ \mathbf{y}_i &= (A+\gamma I)^{-1}(\gamma\cdot\mathbf{y}_{i-1} - \mathbf{f}_{y,i-1}). \end{aligned} \tag{8}$$

Solving this system iteratively leads to an equilibrium state that is the minimum of the snake energy. Local minima traps may be avoided with sufficient regularization, but the need to initialize the snake close to the solution remains.

2.2. B-splines for B-snakes

The B-splines are continuous functions used to build parametric curves. The continuity is dependent on the degree of the B-spline. Cubic B-splines are often used, because this is a C^2 continuous function that provides an implicit smoothness to adjacent points of the corresponding parametric curve. A cubic uniform B-spline curve $g(s) = (x(s), y(s))$ (Figure 1) is built from a finite set of coefficients $c(k)$ with

$$x(s) = \sum_{k=0}^{n-1} c_x(k)B_k(s), \qquad y(s) = \sum_{k=0}^{n-1} c_y(k)B_k(s), \tag{9}$$

where n is the number of coefficients, $B_k(s)$ is the basis function centered on the kth coefficient.

The coefficients are represented through their plane coordinates $c(k) = (c_x(k), c_y(k))$.

Menet et al. [2] showed that (1) can be minimized through an iterative process that is equivalent to (8) except that the matrix A becomes A^b and external forces $\mathbf{f}$ become $\mathbf{f}^b$. The matrix A^b integrates the α and β coefficients of matrix A and the first and second derivatives of the B-spline function. The new external forces $f^b(k)$ are derived from the external forces $f(k)$ and the B-spline function B_k. This iterative

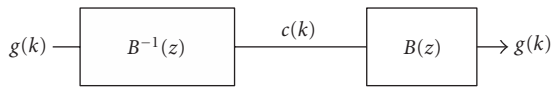


Figure 2: Block diagram of the direct and inverse B-spline filters. $g(k)$ are the curve points. $B^{-1}(z)$ is the direct B-spline filter that computes the B-spline coefficients $c(k)$ from $g(k)$. $B(z)$ is the inverse filter of $B^{-1}(z)$ and is called indirect B-spline filter.

process is given by

$$\begin{aligned}\mathbf{c}_{x,i} &= \left(A^b + \gamma I\right)^{-1}\left(\gamma \cdot c_{x,i-1} - \mathbf{f^b}_{x,i-1}\right)\\ \mathbf{c}_{y,i} &= \left(A^b + \gamma I\right)^{-1}\left(\gamma \cdot c_{y,i-1} - \mathbf{f^b}_{y,i-1}\right),\end{aligned} \tag{10}$$

where $\mathbf{c}_{x,i}$ and $\mathbf{c}_{y,i}$ are vectors containing the coefficient values at iteration i. Such a snake is called B-snake in [2] and its evolution is conducted using the coefficients values. Moreover, Flickner et al. [12] and Brigger et al. [6] use the built-in smoothness property of the cubic B-splines to take the internal energy out of (1). The regularization is then controlled by varying the distance between adjacent coefficients. The more distant two coefficients are, the smoother the spline is. The iterative system in (10) is then simplified by setting α and β to 0 in A, and the iterative minimization process becomes

$$\begin{aligned}c_{x,i}(k) &= c_{x,i-1}(k) - \gamma^{-1} \cdot f^b_{x,i-1}(k),\\ c_{y,i}(k) &= c_{y,i-1}(k) - \gamma^{-1} \cdot f^b_{y,i-1}(k).\end{aligned} \tag{11}$$

In [6], the authors prefer to interact with the snake via real curve points instead of the coefficients. They implement a digital filter $B(z)$ described in [13] that links the B-spline coefficients $c(k)$ to the curve points $g(k)$ by

$$g(k) = b(k) * c(k) \xrightarrow{z} G(z) = B(z) \cdot C(z), \tag{12}$$

where $G(z)$, $B(z)$, and $C(z)$ are the Z-transforms of $g(k)$, $b(k)$, and $c(k)$ and where

$$B(z) = \frac{z + 4 + z^{-1}}{6}. \tag{13}$$

Using the inverse filter of $B(z)$ allows one to obtain the B-spline coefficients from the curve points (Figure 2). As coefficients and curve points are linked together, there is no need to work with B-spline coefficients in the deformation process of the snake. Then (11) can be rewritten using the snake's point $g(k) = (x(k), y(k))$:

$$\begin{aligned}x_i(k) &= x_{i-1}(k) - \gamma^{-1} \cdot f_{x,i-1}(k),\\ y_i(k) &= y_{i-1}(k) - \gamma^{-1} \cdot f_{y,i-1}(k),\end{aligned} \tag{14}$$

where $x_i(k)$ and $y_i(k)$ are the coordinates of the kth curve point at iteration i.

2.3. Smoothing B-splines

B-splines interpolation is a mean of building continuous curves from a finite set of coefficients. A B-spline interpolates exactly a set of points $g(k)$. However, an exact interpolation is often not a reliable method of reconstruction. Reinsch [9] illustrates this issue with a 1D signal taken from an imperfect measuring tool. He proposed to approximate the set of measured points by spline functions. Given a set of points $g(k)$, the smoothing spline is the one that minimizes

$$\varepsilon_s^2 = \sum_{k=-\infty}^{+\infty} \left(g(k) - \hat{g}(k)\right)^2 + \lambda \int_{-\infty}^{+\infty} \left(\frac{\partial^2 \hat{g}(s)}{\partial^2 s}\right)^2 ds, \tag{15}$$

where $g(k)$ is the point set, $\hat{g}(k)$ is the approximating point set of $g(k)$, and $\hat{g}(s)$ is the continuous function that interpolates $\hat{g}(k)$.

The first term of (15) represents the error between the original data set $g(k)$ and the approximating one $\hat{g}(k)$. The second term, weighted by λ, represents the global curvature of the curve. A large λ gives more importance to the smoothing aspect of $\hat{g}(k)$ whereas a small λ imposes $\hat{g}(k)$ to be closer to $g(k)$. It is proven [9, 10] that a cubic B-spline is the solution of (15).

In [10], Unser et al. apply the B-spline filtering approach to the smoothing spline formulation. They show that the coefficients of the approximating B-spline could be computed through an IIR filter S_λ. Consequently, from these approximating coefficients $\hat{c}(k)$, one can find the approximating B-spline points $\hat{g}(k)$ through a B-spline filter B (Figures 3 and 4).

The smoothing B-spline filter transfer function is given by

$$SB_\lambda(z) = S_\lambda(z) \cdot B(z) = \frac{z + 4 + z^{-1}}{a + b \cdot \left(z + z^{-1}\right) + c \cdot \left(z^2 + z^{-2}\right)}, \tag{16}$$

where

$$S_\lambda(z) = \frac{6}{z + 4 + z^{-1} + 6 \cdot \lambda\left(z^2 - 4z + 6 - 4z^{-1} + z^{-2}\right)}, \tag{17}$$

$B(z)$ is given in (13), and $a = 4+36\lambda$, $b = 1-24\lambda$, and $c = 6\lambda$.

$SB_\lambda(z)$ represents a fourth order symmetric filter with coefficients depending on λ. It is a low-pass filter with a cut-off frequency controlled by λ (Figure 5). The link between λ and the cut-off frequency f_c with an attenuation of -3 dB is given by

$$\begin{aligned}\lambda(f_c) &= \frac{-\cos\left(2\pi f_c\right) + \cos\left(2\pi f_c\right)\sqrt{2} - 2 + 2\sqrt{2}}{12\left(\cos\left(2\pi f_c\right)^2 - 2\cos\left(2\pi f_c\right) + 1\right)},\\ f_c(\lambda) &= \frac{\arccos\left(\left(-1+\sqrt{2}+24\cdot\lambda+\sqrt{3-2\sqrt{2}-144\cdot\lambda+144\sqrt{2}\cdot\lambda}\right)/24\cdot\lambda\right)}{2\pi}.\end{aligned} \tag{18}$$

When λ equals 0, (34) shows that $SB_\lambda(z)$ equals 1. It means that $\hat{g}(k) = g(k)$, that is, there is no approximation (Figure 3).

In the time domain, the filtering is represented by a convolution equation involving the input signal $g(k)$ and the

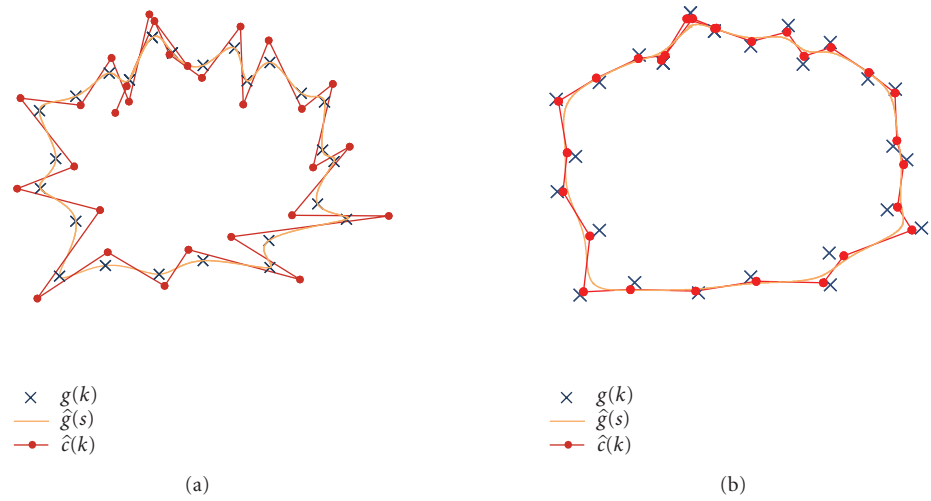


FIGURE 3: A smoothing B-spline curve. The points to approximate are the $g(k)$ represented by the cross symbols. The $\hat{c}(k)$ are the coefficients of the smoothing B-spline and are represented by the dot symbols. The curve $\hat{g}(s)$ is the approximating curve according to λ. (a) Approximation of a point set $g(k)$ with $\lambda = 0$. Note that in this case, we obtain a B-spline interpolation. (b) Approximation of the same point set with $\lambda = 0.1$.

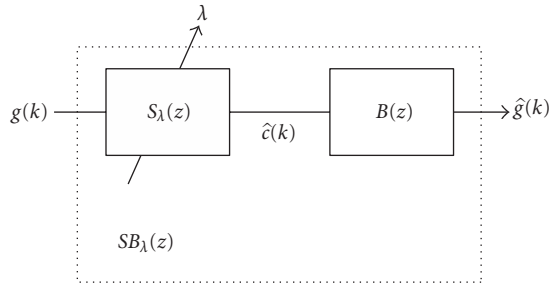


FIGURE 4: Block diagram of the smoothing B-spline filter SB_λ. $g(k)$ are the curve points. $S_\lambda(z)$ is a filter that computes the smoothing B-spline coefficients $\hat{c}(k)$. $B(z)$ is a filter that computes the B-spline curve points from a set of coefficients. $\hat{g}(k)$ are the approximating points of $g(k)$.

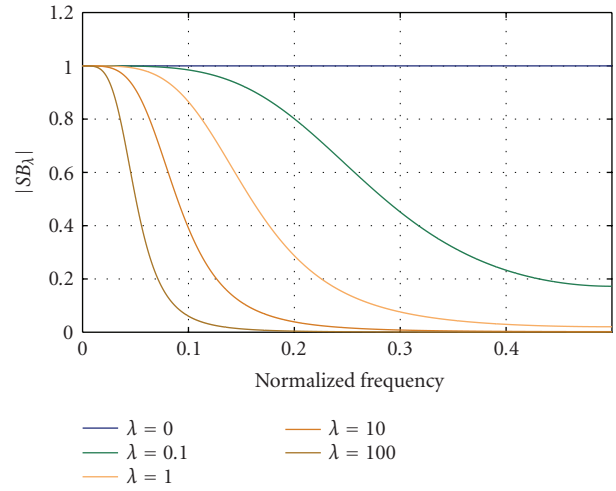


FIGURE 5: Frequency response of the smoothing-spline filter SB_λ for different values of λ.

smoothed output $\hat{g}(k)$:

$$\hat{g}(k) = g(k) * b(k) * s_\lambda(k) = g(k) * sb_\lambda(k), \tag{19}$$

where $sb_\lambda(k)$ is the impulse response of the approximation filter SB_λ.

The implementation of the smoothing B-spline filter is not straightforward. An efficient implementation was proposed by Unser et al. [13]. It implements a causal/anticausal filtering technique (see the appendix), resulting in $O(n)$ complexity of the filtering process, n being the number of points of the input signal.

2.4. Smoothing B-splines and snakes

The smoothing-spline snake-based algorithm proposed by Precioso et al. [8] takes advantage of the smoothness control allowed by a smoothing B-spline filter in the regularization of an active contour. The segmentation results obtained with this algorithm show a good robustness to noise for λ varying in the range $[0.1, 1]$. The authors underlined that the computational cost of the contour smoothing is negligible compared to the statistical evaluation involved in the region-based active contour. This algorithm contains an initialization step, an evolution step, and a convergence test. Let i be the iteration index, $g(k)$ the sampling points, $\hat{c}(k)$ the smoothing spline coefficients.

At initialization step, we get $g_{i=0}(k)$ from an interface. The smoothing spline coefficients are

$$\hat{c}_{i,0}(k) = s_\lambda(k) * g_{i,0}(k), \tag{20}$$

where $s_\lambda(k)$ is the impulse response of $S_\lambda(z)$.

The snake evolution is conducted through the following steps.

(a) Computation of the smoothing spline points:

$$\hat{g}_i(k) = b(k) * \hat{c}_i(k). \tag{21}$$

(b) Computation of evolution forces ($d_i(k)$) according to a region-based scheme detailed in [8]:

$$d_i(k) = \nu \cdot N(k), \tag{22}$$

where ν is defined as the velocity of the contour, N is the normal vector to the contour.

(c) Displacement of $\hat{g}_i(k)$ by $d_i(k)$ to get the next points $g_{i+1}(k)$:

$$g_{i+1}(k) = \hat{g}_i(k) + d_i(k). \tag{23}$$

(d) Computation of smoothing spline coefficients:

$$\hat{c}_{i+1}(k) = s_\lambda(k) * g_{i+1}(k). \tag{24}$$

(e) Return to step (a) until convergence.

The convergence test is based on some variation measures of the contour characteristics [8]. If an equilibrium criterion is reached, the snake stops. We note that each iteration involves a smoothing part represented by the convolution equations (21) and (24), where $b(k)$ is the impulse response of the filter $B(z)$, respectively. Introducing the iteration index i and the deformation process through the evolution forces ($\nu \cdot N_i(k)$) with respect to the algorithm, we obtain, from (21) and (23),

$$g_{i+1}(k) = \hat{g}_i(k) + d_i(k) = \hat{c}_i(k) * b(k) + d_i(k). \tag{25}$$

From (25), we find

$$\hat{c}_{i+1}(k) = \hat{c}_i(k) * sb_\lambda(k) + d_i(k) * s_\lambda(k), \tag{26}$$

where $d_i(k) = \nu \cdot N_i(k)$.

Finally, from (26) and using the Z transform, we obtain:

$$\hat{C}_{i+1}(z) = \hat{C}_0(z) \cdot \prod^{i+1} SB_\lambda(z) + \sum_{j=0}^{i} \left(D_j(z) \cdot S_\lambda(z) \prod^{i-j} SB_\lambda(z) \right). \tag{27}$$

Equation (27) presents one term linked to the initial smoothing-spline coefficients $\hat{C}_0(z)$ which is multiplied by a product of the $SB_\lambda(z)$ approximation filter. As this filter is a low-pass filter, the initial position of the snake tends to have less importance when the number of iterations increases. A similar behavior is observed within the second term of (26) where the oldest deformation forces tend to be canceled. The product terms involving the $SB_\lambda(z)$ approximation filter in (27) explains the restricted range $[0.1, 1]$ of the λ parameter values. For a given number of iterations, greater values of λ make the snake to shrink. The reproducibility of the results is hard to obtain because the SB_λ regularization effect is strongly linked to the number of iterations that is itself linked to the initial position of the snake.

In this paper, we propose another approach that uses the smoothing-spline filter in an active contour scheme where λ values are not limited and where the regularization is not iteration-dependent and can be locally defined.

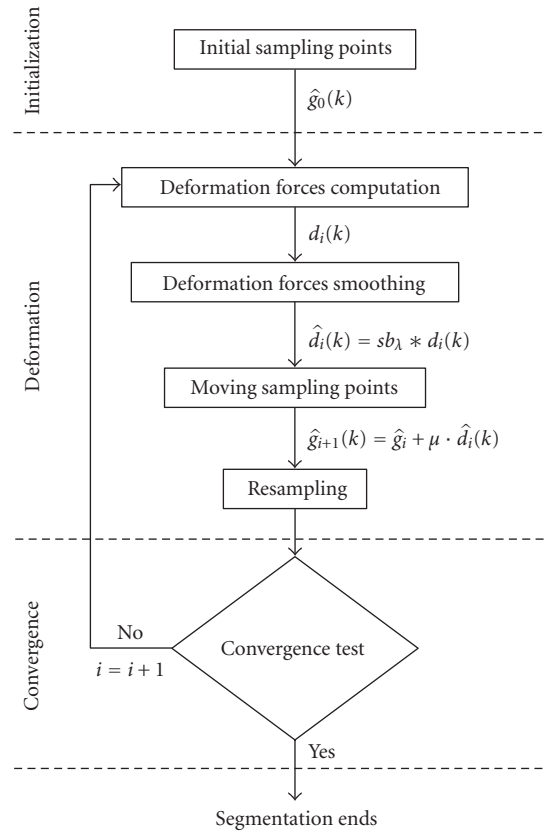


Figure 6: Flow-chart of the locally regularized smoothing B-snake algorithm.

3. LOCALLY REGULARIZED SMOOTHING B-SNAKE

An overview of the proposed locally regularized Smoothing B-Snake (LRSB-snake) is given in Figure 6. From an initial contour $\hat{g}_0(k)$ and the image to segment, we compute the deformation forces (see Section 3.1). The regularization is done by smoothing the deformation forces. We can apply a global regularization (Section 3.2) or a local one (Section 3.3). The contour is then moved by applying the regularized deformation forces. To enforce a similar behavior of the smoothing process at each iteration, the contour is resampled as discussed in Section 3.4. If the contour is stabilized, the iterative process is stopped.

3.1. Deformation forces computation

The deformation of parametric models is performed by an energy minimization [1]. The variational method used to complete the minimization leads to the force balance given by (5). This equation involves internal and external forces. Internal forces that have a regularization role will be discussed in Section 3.2. This section focuses on external forces and their usage as deformation vectors. The external forces are directly derived from the image to segment. They guide the snake to the desired features. Within the LRSB-snake algorithm any type of external forces may be used. Basically, a

Laplacian vector field (6) computed from a Gaussian-blurred version of the image gives suitable deformation forces [1]. Xu and Prince proposed the gradient vector flow (GVF) [4] to diffuse the gradient over the image in order to give a greater range to the attraction forces. The balloon force [3] is another external force. It is a vector directed along the normal of the contour. It creates a pressure force at each point of the snake that makes it swelling.

The sum of every considered forces gives the deformation vector $d_i(k)$ at each point k and at each iteration i. As the basic idea of a deformation process is to move every point according to a deformation vector, we define the following deformation equation:

$$g_{i+1}(k) = g_i(k) + \mu \cdot d_i(k), \tag{28}$$

where i is the iteration index, $g(k)$ are the snake points, $d_i(k)$ are the deformation vectors, that is, the external forces, and μ is a step-size parameter involved in the speed of convergence. Equation (10) corresponds to (28) if we set $\mu = -\gamma^{-1}$ and $d(k) = f(k)$. Note that (28) does not imply any regularization process.

3.2. Global regularization process through deformation forces smoothing

Regularization of the deformable model is essential to ensure a good robustness to noise of such segmentation approach. Usually, the regularization is assumed through an internal energy term in the snake energy formulation [1] or implicitly through a variation of the contour sampling [6]. Such regularization prevents incoherent deformation of the contour by introducing a curvature-based penalty. We choose to control the curvature of the contour with a smoothing B-spline filter that minimizes the curvature optimally [9] according to a single parameter λ.

Equation (29) gives the snake regularized by the SB_λ approximation filter at iteration i,

$$\hat{g}_i(k) = sb_\lambda(k) * g_i(k). \tag{29}$$

From (28) and (29), we obtain (30) that yields the deformation and the motion steps of the algorithm (Figure 6),

$$\hat{g}_{i+1}(k) = \hat{g}_i(k) + (\mu \cdot d_i(k)) * sb_\lambda(k). \tag{30}$$

Finally, from (30) and using the Z-transform, we obtain

$$\hat{G}_{i+1}(k) = \hat{G}_0(k) + \mu \cdot \sum_{j=0}^{i} D_j(z) \cdot SB_\lambda(z). \tag{31}$$

It is clear in (30) that we bring into effect the snake regularization by a smoothing filtering of the deformation forces (Figure 6). Consequently (31), an infinite iterative process does not lead to an infinite successive convolution of d_i. Compared to [8], the regularization does not depend on the number of iterations. It allows λ to control the cut-off frequency of the SB_λ approximation filter and thus the regularization level by taking any real positive values. As the regularization is done by a digital recursive filter, it preserves the processing speed mentioned in [8].

In practice, the 1D smoothing B-spline filter is used to filter a parametric curve in the plane. It is applied on each parametric component of the curve as in (32). Figure 7 illustrates the filtering of such a parametric curve,

$$\hat{g}(k) = sb_\lambda(k) * g(k) \Longrightarrow \begin{pmatrix} \hat{g}_x(k) \\ \hat{g}_y(k) \end{pmatrix} = \begin{pmatrix} sb_\lambda(k) * g_x(k) \\ sb_\lambda(k) * g_y(k) \end{pmatrix}. \tag{32}$$

Note that a 1-D filter is usually applied on a uniformly sampled signal. As we want to keep the same filter frequency response which is λ-dependent, we enforce a uniform sampling of the contour in our algorithm (see Section 3.4).

3.3. Local regularization process

From the regularization term of (30), one can write the following equation:

$$\hat{D}_i(z) = SB_\lambda(z) \cdot D_i(z), \tag{33}$$

where

$$SB_\lambda(z) = S_\lambda(z) \cdot B(z) = \frac{z + 4 + z^{-1}}{a + b \cdot (z + z^{-1}) + c \cdot (z^2 + z^{-2})} \tag{34}$$

with $a = 4 + 36\lambda$, $b = 1 - 24\lambda$, and $c = 6\lambda$.

In (Appendix A.2), we show that (33) conducts to two recurrence equations (A.10) and (A.11) where the filters coefficients a, b, and c appears. We propose to make those two equations space-varying by making λ dependent of the kth contour point. Thus, a, b, and c become

$$a_k = 4 + 36 \cdot \lambda_k, \qquad b_k = 1 - 24 \cdot \lambda_k, \qquad c_k = 6 \cdot \lambda_k \tag{35}$$

and the space-varying recurrence equations are

$$\begin{aligned} \hat{d}_1(k) &= c_k \cdot d(k) + a_k \cdot \hat{d}_1(k-1) - b_k \cdot \hat{d}_1(k-2), \\ \hat{d}_2(k) &= c_k \cdot \hat{d}_1(k) + a_k \cdot \hat{d}_2(k+1) - b_k \cdot \hat{d}_2(k+2). \end{aligned} \tag{36}$$

Consequently, each point of the snake has its own regularization rate. We are able to affect different values of λ along the contour (Figure 8) according to several strategies like local image information (to adapt λ to noise level or to pertinent image features under the contour), or prior knowledge introduced in the initial model (to keep the contour in a controlled deformation range). These strategies may have different impacts on snake evolution that are beyond the scope of this paper.

3.4. Resampling

For a given λ the contour sampling rate has to be constant to keep the same cut-off frequency of the smoothing filter (see Section 2.3) and consequently the same regularization effect. Our algorithm implements a resampling step at each iteration (Figure 6) to get a constant distance between the contour points. We do not implement a subdivision scheme, but

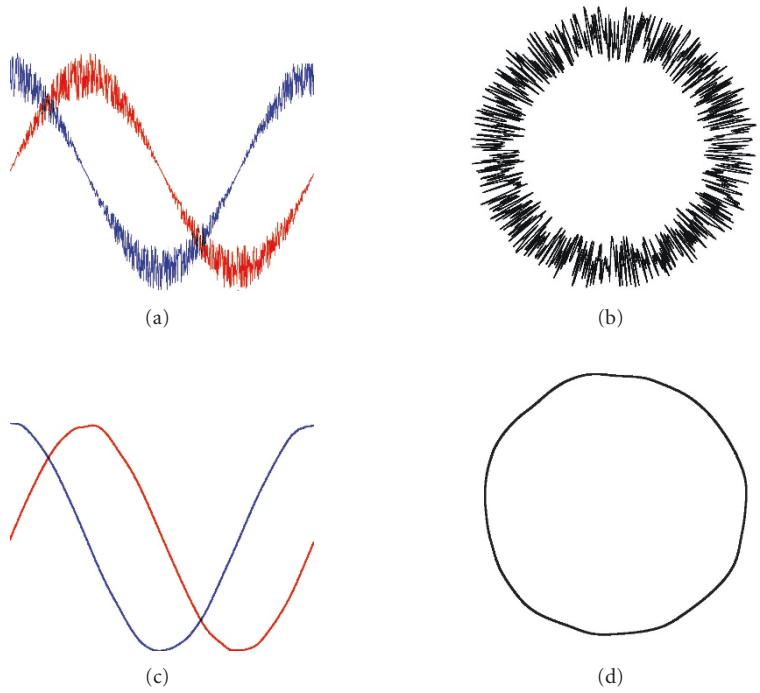


Figure 7: Smoothing B-spline filtering of a parametric circle with parameter s. Noisy circle in (b) is described by $x(s)$ and $y(s)$ (a). The circle in (b) has a constant radius disturbed by a uniformly distributed additive noise. $x(s)$ and $y(s)$ are smoothed separately by the smoothing B-spline filter (c). (d) shows the smoothed parametric circle.

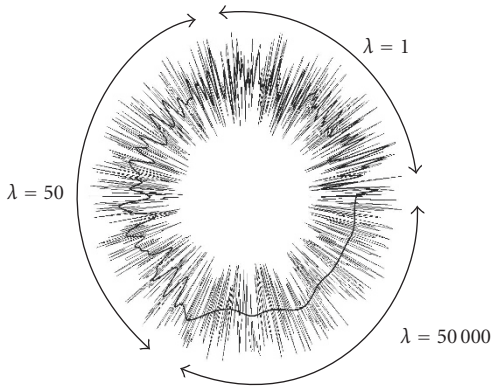


Figure 8: Spatially variant smoothing-spline filtering. The input signal is the noisy circle. This signal is filtered through a smoothing-spline filter where the smoothing parameter λ varies along the contour. The resulting signal is represented in bold.

a resampling one. Starting with one existing point, each next point is set at a fixed distance from the previous one as described in the following algorithm.

Let $g(s)$ be the original contour, $g'(i)$ the resampled contour, i the new point index, s the continuous parameter, e the wanted constant sampling distance, and N the number of points of the original contour. $N-1$ represents also the total curve length.

(1) Initialization: $g'(0) = g(0)$, $i = 0$.
(2) Incrementation of i.
(3) Find s such that $\|g(s)g'(i-1)\| = e$, then set $g'(i) = g(s)$.
(4) Repeat steps (2) and (3) while $s < N-1$.

As $g'(i) = (x'(i), y'(i))$, we have $\|g'(i)g'(i-1)\| = \sqrt{(x'(i)-x'(i-1))^2 + (y'(i)-y'(i-1))^2}$.

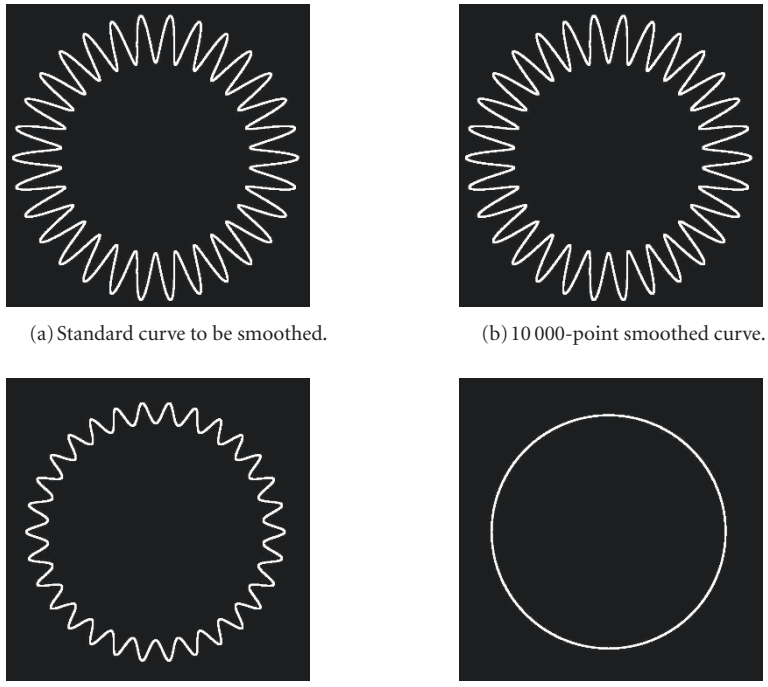
The resampling impact is illustrated in Figure 9 where a curve with different sampling rates is smoothed via a smoothing B-spline filter with the same λ. It appears that the greatest interpoint distance (Figure 9(d)) leads to the smoothest curve.

When λ is locally variant, the λ-values under the new points are obtained by linear interpolation between the λ-values of the previous and the next old points.

3.5. Open and closed contours

The smoothing-spline filter has an infinite impulse response. To implement the corresponding difference equation (33), Unser et al. [13] proposed to initialize the filtering process with an approximation of the impulse response. Boundary conditions have to be clearly defined in order to choose a correct extrapolation of the signal.

If we consider a closed contour, the signal $g(k)$ is made periodic. Thus the extrapolation may be seen as a modulo function that is well adapted to closed contour. The smoothed circle in Figure 9 is obtained in such a way.



(a) Standard curve to be smoothed.

(b) 10 000-point smoothed curve.

(c) 1000-point smoothed curve.

(d) 100-point smoothed curve.

Figure 9: Illustration of the influence of sampling over the λ value. Each curve is smoothed by the same smoothing B-spline filter with $\lambda = 1000$.

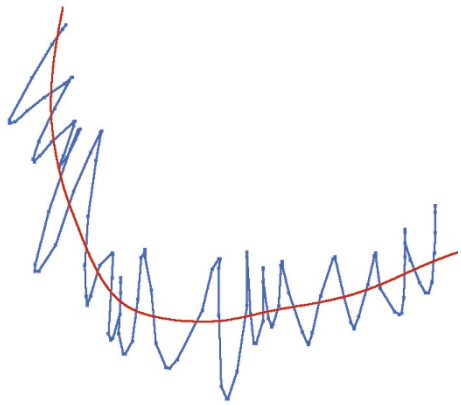


Figure 10: Illustration of an opened contour smoothing using an antimirror with pivot point extension. A hand-drawn line and its smoothed version are given.

The open contour case is different because we need to extrapolate the parametric signal while keeping a suitable continuity at endpoints. Several extrapolation techniques have been proposed leading to different behaviors [6, 14]. We choose to implement an antisymmetric mirror with pivot point extension [6] in order to conserve the continuity at extreme points. A smoothed open contour including such an extension is illustrated in Figure 10.

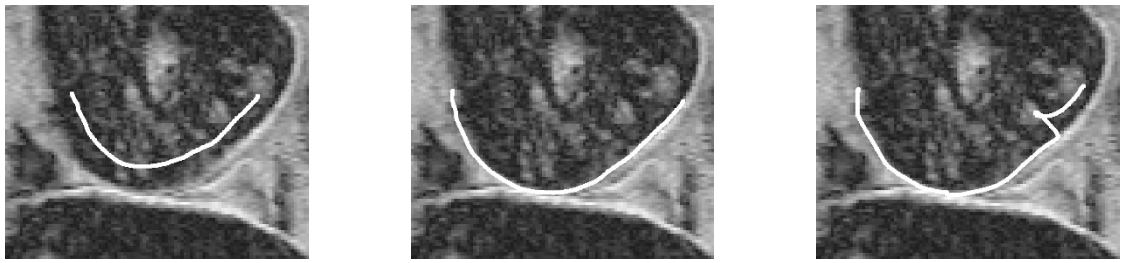
4. RESULTS

This section gives results obtained with the proposed LRSB-snake algorithm. First, the global regularization mode is illustrated using real MRI images with opened and closed contours. Then, the LRSB-snake is applied on synthetic image and real MRI image to illustrate the advantage of such a local regularization. All these results have been obtained with the following external forces: Laplacian vectors of a Gaussian-blurred version of the image combined with a balloon force to increase the convergence speed.

4.1. Global regularization

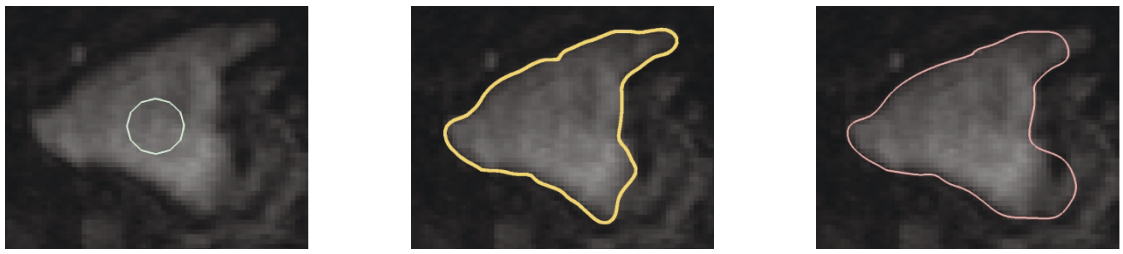
Figure 11(a) shows an MR image of a guinea-pig knee and an initial opened smoothing B-snake. The feature to detect is the femoral border. With a too low λ value ($\lambda = 71$), the final result obtained after 1280 iterations is corrupted by a local minimum (Figure 11(c)). We set a larger λ value ($\lambda = 1000$) to avoid this artifact (Figure 11(c)). The final result obtained after 630 iterations is close to the wanted femoral border.

Figure 12 shows an anatomic structure in an MRI angiography. This structure presents an upper-right protuberance and a lack of gray-level gradient in the bottom right place. Figure 12 illustrates the algorithm behavior with a global regularization and a closed contour. With $\lambda = 200$ (Figure 12(c)), the right upper part of the anatomic structure is missing and the final position is not correct at the bottom right, after 550 iterations. A correct segmentation is obtained



(a) Initial snake. (b) $\lambda = 1000, 630$ iterations. (c) $\lambda = 71, 1280$ iterations.

FIGURE 11: MRI image of a guinea-pig knee and an initial 67 points snake.



(a) Initial snake. (b) $\lambda = 12, 550$ iterations. (c) $\lambda = 220, 480$ iterations.

FIGURE 12: Segmentation of an MRI image through a close snake. Final snakes are 150 points long.

(Figure 12(b)) with a lower value of λ ($\lambda = 12$) and 480 iterations.

4.2. *Local regularization*

To illustrate the limitation of the global regularization, we use a synthetic image obtained from a circle (Figure 13(a)). The circle contour is modulated to introduce as many ranges of curvature variation as there are different modulation frequencies. A Gaussian noise is then added to the image. Global regularization demonstrates its limits on Figures 13(b) and 13(c). A low global λ value leads to an evolution of the contour very sensitive to noise. The snake may be stuck in local minima as in Figure 13(b). A large global λ value induces too much constraint on the snake curvature. The final contour could only outline a mean circle (Figure 13(c)).

On Figure 13(d), we use the locally regularized scheme where λ was made spatially variant. The highest frequency part (I) was attached to a low λ ($\lambda = 1$). We then increase λ to 10 in order to outline successfully the second part (II) of the disk. Last part (III) being perfectly circular, λ was set to 100.

We note with these tests that the LRSB-snake provides a segmentation result (Figure 13(d)) closest to the object to outline (white contour in Figure 13(a)).

Figure 14 illustrates the same behavior on a real MRI image that presents interesting features: an unsharp gradient area ("ghost gradient") at the top of the shape that is not an edge to outline and a lack of gradient at middle right. On Figure 14(b), balloon forces induce a leak of the snake which is not sufficiently regularized. The ghost gradient corrupts the final result also by introducing a curve that does not exist. A larger λ value (Figure 14(c)) prevents the leak but gives too much importance to the unsharp gradient area.

The LRSB-snake is then applied on this image (Figure 14(d)). A λ map gives the λ values at each image position. In this example, the λ map is manually defined, with values empirically determined as follows. Positions where the contour is well visible take a small λ value ($\lambda = 1$), and positions where the contour tends to disappear take a high λ value ($\lambda = 300$). One can observe on Figure 14(d) that such a local regularization prevents the leak, manages correctly the ghost gradient, and stops the swell at the top. A strategy to automatically define λ variations is beyond the scope of this paper.

5. CONCLUSIONS

In this paper, we propose a locally regularized smoothing B-snake algorithm. The regularization process uses an approximating smoothing-spline filter applied directly on the snake point displacement. This algorithm conserves the advantages of snake algorithms and offers a local control of the regularization through the λ value defined at each snake points. As the regularization is implemented through a recursive implementation of a digital filter, this algorithm is fast.

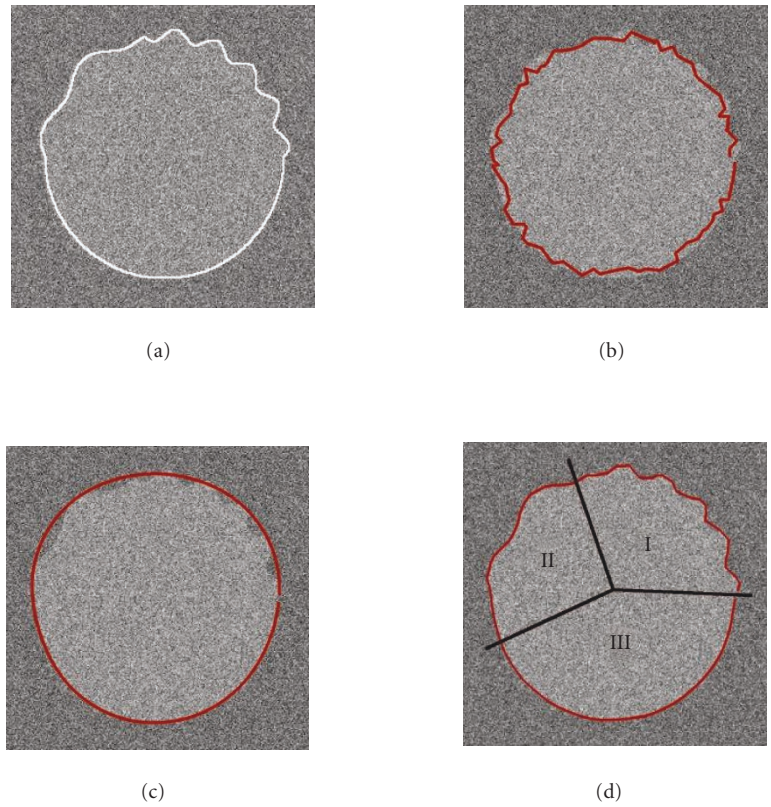


Figure 13: Constant and locally regularized 100-points smoothing B-snake on a synthesis image. (a) Noisy object with the reference contour in white. (b) Final segmentation with global low λ ($\lambda = 1$). (c) Final segmentation with global high λ ($\lambda = 100$). (d) Locally regularized smoothing B-Snake. (I : $\lambda = 1$, II : $\lambda = 10$, III : $\lambda = 100$).

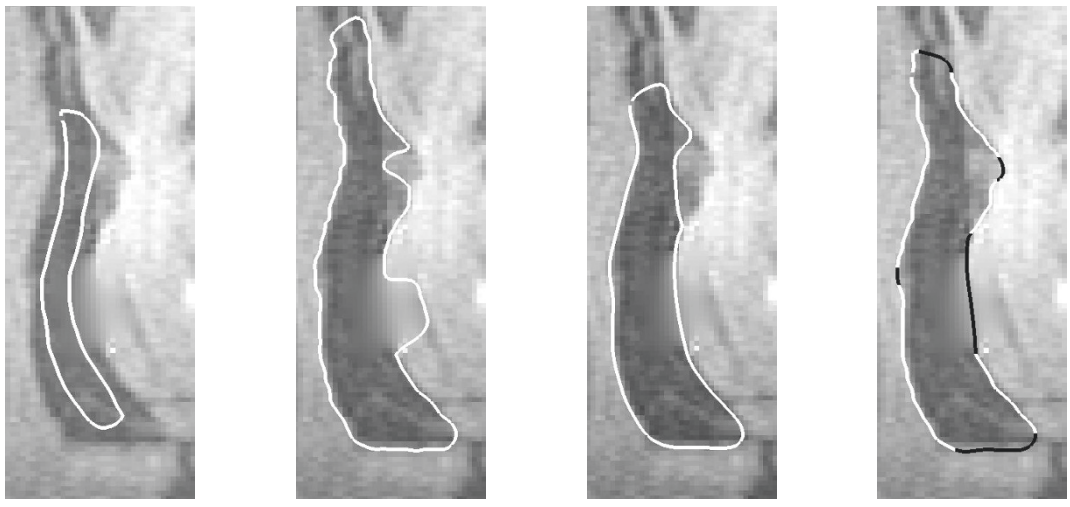


Figure 14: Constant and local regularization on an MRI image. (a) Initial snake and original MRI image. (b) Final segmentation after 310 iterations using $\lambda = 1$. (c) Final segmentation after 250 iterations using $\lambda = 300$. (d) Final segmentation after 330 iterations using local λ values (black for $\lambda = 300$, white for $\lambda = 1$).

This algorithm associated to pertinent local λ definition offers a powerful tool for introducing prior knowledge and consequently makes the segmentation process more robust.

A. APPENDIX

This appendix details the smoothing B-spline filter implementation which is linked to the λ value that impacts the filter pole values.

A.1. Smoothing B-spline filter pole analysis

We implement the smoothing spline-filter as discussed in [13]. The filtering system is the one presented in Figure 4. The $B(z)$ filter is an FIR filter and is implemented through the difference equation associated to its transfer function (13). The $S(z)$ filter implementation is not straightforward because of its transfer function (17). The $S(z)$ filter is equivalent to a cascade of two symmetric second-order filters:

$$S_\lambda(z) = S_\lambda^+(z) \cdot S_\lambda^+(z^{-1}) \tag{A.1}$$

with

$$S_\lambda^+(z) = \frac{1 - 2\rho \cdot \cos(\omega) + \rho^2}{1 - 2\rho \cdot \cos(\omega) \cdot z^{-1} + \rho^2 \cdot z^{-2}}. \tag{A.2}$$

In (A.2), ρ and ω are the magnitude and argument of the two complex conjugate roots ($p_1 = \rho \exp^{j\omega}$, $p_2 = \rho \exp^{-j\omega}$) of the $S_\lambda^+(z)$ denominator, and they are linked to the λ value by

$$\begin{aligned} \rho &= \frac{24\lambda - 1 - \sqrt{\xi}}{24\lambda} \cdot \sqrt{\frac{48\lambda + 24\lambda \cdot \sqrt{3 + 144\lambda}}{\xi}}, \\ \tan(\omega) &= \sqrt{\frac{144\lambda - 1}{\xi}}, \end{aligned} \tag{A.3}$$

$$\xi = 1 - 96\lambda + 24\lambda \cdot \sqrt{3 + 144\lambda}. \tag{A.4}$$

Equations (A.3) are valid for λ value greater than 1/24. For other λ values several cases have to be considered.

(1) When $\lambda = 1/24$, ρ and ω take the specific values given by

$$\begin{aligned} \rho &= \sqrt{11 - \sqrt{120}}, \\ \omega &= \frac{\pi}{2}. \end{aligned} \tag{A.5}$$

(2) When λ value belongs to the interval $]1/144 : 1/24[$, (A.3) are still valid but ρ value is greater than one, and consequently the associated conjugate roots are the roots of $S_\lambda^+(z^{-1})$ instead of being those of $S_\lambda^+(z)$.

(3) When λ value belongs to the interval $]0 : 1/144]$, the roots of $S_\lambda^+(z)$ become real and are given by

$$p_1 = \frac{2 + x_1 + \sqrt{4x_1 + x_1^2}}{2}, \qquad p_2 = \frac{2 + x_2 + \sqrt{4x_2 + x_2^2}}{2} \tag{A.6}$$

with $x_1 = (-1 + \sqrt{1 - 144\lambda})/12\lambda$ and $x_2 = (-1 - \sqrt{1 - 144\lambda})/12\lambda$.

Then, we can write $S_\lambda^+(z)$ as follows:

$$S^+\lambda(z) = \sqrt{\frac{p_1 p_2}{\lambda}} \frac{1}{1 - (p_1 p_2) z^{-1} + p_1 p_2 z^{-2}}. \tag{A.7}$$

(4) When λ value equals 0, we have

$$\lambda = 0 \Longleftrightarrow S_\lambda(z) = B^{-1}(z) \Longleftrightarrow SB_\lambda(z) = 1. \tag{A.8}$$

It means that the smoothing B-spline filter has no effect: input and output signals are equal.

A.2. Smoothing B-spline filter implementation

The filter $SB(z)$ is implemented as a cascade of two causal/anticausal filters (A.1) followed by the filter $B(z)$ (13),

$$\begin{aligned} \hat{D}_1(z) &= D(z) \cdot S_\lambda^+(z), \\ \hat{D}_2(z) &= \hat{D}_1(z) \cdot S_\lambda^+(z^{-1}), \\ D(z) &= \hat{D}_2(z) \cdot B(z). \end{aligned} \tag{A.9}$$

The corresponding difference equations are given by

$$\hat{d}_1(k) = c \cdot d(k) + a \cdot \hat{d}_1(k-1) - b \cdot \hat{d}_1(k-2), \tag{A.10}$$

$$\hat{d}_2(k) = c \cdot \hat{d}_1(k) + a \cdot \hat{d}_2(k+1) - b \cdot \hat{d}_2(k+2), \tag{A.11}$$

$$\hat{d}(z) = \frac{1}{6}(\hat{d}_2(k-1) + 4 + \hat{d}_2(k+1)), \tag{A.12}$$

where a, b, and c are computed from ρ and ω depending on the λ value as detailed in the previous appendix section. Equation (A.10) represents the causal filtering while (A.11) represents the anticausal one. This is implemented by reversing the signal $\hat{d}_1(k)$, filtering it with S_λ^+ and reversing the output.

As the smoothing B-spline filter is an IIR filter, the computation of the output at instant k requires the output knowledge at instants $k-1$ and $k-2$ ((A.10) and (A.11)). Consequently, the filtering process has to be initialized for the two first points (in the causal recursion) and the two last ones (in the anticausal recursion). For these initializations, we use an approximation of the impulse response of $S^+\lambda(z)$:

$$\hat{d}_1(0) = \sum_{k=0}^{n} d(k) s_\lambda^+(k), \qquad \hat{d}_1(1) = \sum_{k=0}^{n} d(k) s_\lambda^+(k-1), \tag{A.13}$$

where n is the length of the approximation. Note that n should be sufficiently high to correctly approximate the impulse response. Consequently, n value will be increased with the λ value. The anticausal filtering is initialized equivalently. When $\lambda > 1/144$, the impulse response is given by

$$s_\lambda^+(k) = (1 - 2\rho \cdot \cos(\omega) + \rho^2) \frac{\rho^{|k|} \sin((|k| + 1)\omega)}{\sin(\omega)}. \tag{A.14}$$

When $\lambda < 1/144$, the impulse response is given by

$$s_\lambda^+(k) = \sqrt{\frac{p_1 p_2}{\lambda}}\left(\frac{1}{1-p_2/p_1}p_1^{|k|} + \frac{1}{1-p_1/p_2}p_2^{|k|}\right). \tag{A.15}$$

When $\lambda = 1/144$, the impulse response is given by

$$s_\lambda^+(k) = (|k|+1)\cdot p_1^{|k|}. \tag{A.16}$$

ACKNOWLEDGMENTS

This work is in the scope of the scientific topics of the PRC-GdR ISIS Research Group of the French National Center for Scientific Research CNRS. We would like to thank the reviewers of this paper for their comments and suggestions.

REFERENCES

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: active contour models," in *Proceedings of the 1st International Conference on Computer Vision*, pp. 259–268, London, UK, June 1987.

[2] S. Menet, P. Saint-Marc, and G. Medioni, "B-snakes: implementation and application to stereo," in *Proceedings of Image Understanding Workshop*, pp. 720–726, Pittsburgh, Pa, USA, September 1990.

[3] L. D. Cohen and I. Cohen, "Finite-element methods for active contour models and balloons for 2-D and 3-D images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1131–1147, 1993.

[4] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 359–369, 1998.

[5] M. Wang, J. Evans, L. Hassebrook, and C. Knapp, "A multistage, optimal active contour model," *IEEE Transactions on Image Processing*, vol. 5, no. 11, pp. 1586–1591, 1996.

[6] P. Brigger, J. Hoeg, and M. Unser, "B-spline snakes: a flexible tool for parametric contour detection," *IEEE Transactions on Image Processing*, vol. 9, no. 9, pp. 1484–1496, 2000.

[7] P. Brigger and M. Unser, "Multi-scale B-spline snakes for general contour detection," in *Wavelet Applications in Signal and Image Processing VI*, vol. 3458 of *Proceedings of SPIE*, pp. 92–102, San Diego, Calif, USA, July 1998.

[8] F. Precioso, M. Barlaud, T. Blu, and M. Unser, "Robust real-time segmentation of images and videos using a smooth-spline snake-based algorithm," *IEEE Transactions on Image Processing*, vol. 14, no. 7, pp. 910–924, 2005.

[9] C. H. Reinsch, "Smoothing by spline functions," *Numerische Mathematik*, vol. 10, no. 3, pp. 177–183, 1967.

[10] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing—part I. Theory," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 821–833, 1993.

[11] M. Jacob, T. Blu, and M. Unser, "Efficient energies and algorithms for parametric snakes," *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1231–1244, 2004.

[12] M. Flickner, H. Sawhney, D. Pryor, and J. Lotspiech, "Intelligent interactive image outlining using spline snakes," in *Proceedings of the 28th Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 731–735, Pacific Grove, Calif, USA, October-November 1994.

[13] M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing—part II. Efficient design and applications," *IEEE Transactions on Signal Processing*, vol. 41, no. 2, pp. 834–848, 1993.

[14] L. Weruaga, R. Verdú, and J. Morales, "Frequency domain formulation of active parametric deformable models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 12, pp. 1568–1578, 2004.

Jérôme Velut graduated in 2003 as a Computing Engineer from the Université de Provence (France). He then specialized in image processing at the INSA, Lyon, and received the Master degree in image and system in 2004. Since this graduation, he is a Ph.D. student at the Research and Applied Image and Signal Processing Center (CREATIS) at the INSA, Lyon. His research interests include image processing, deformable models, and 3D modeling.

Hugues Benoit-Cattin received in 1992 the Engineer degree (electrical engineering) and in 1995 the Ph.D. degree (wavelet image coding of medical images), both from INSA, Lyon, France. He is Associated Professor at INSA, Lyon, at the Telecommunications Department. His teaching activities mainly concern information theory, signal and image processing. He worked since 1992 at CREATIS Laboratory (CNRS no. 5515 INSERM U630). For six years (1992–1998), he worked on image coding including DCT, wavelet- and fractal-based coding schemes applied to medical images and spatial images. Since 1998, he is member of the Volumic Imaging Research Team of CREATIS. His main research interests include image segmentation as well as MRI image acquisition, correction, and simulation.

Christophe Odet received in 1979 the Engineering degree (electrical engineering) and in 1984 the Ph.D. degree (nondestructive testing), both from the National Institute for Applied Sciences of Lyon (France). He worked since 1984 at CREATIS as Assistant Professor. His primary research interests included image segmentation and classification based on texture analysis and signal and image restoration. He has been involved in the application of the previous methods to linear and 2D image sensors for the control of high-speed moving products. He is now Professor at CREATIS and is mainly involved in image processing and segmentation for medical and biological applications such as bone analysis or small animals imaging.