Segmentation with active contours: 
apparative study of B-Spline and Level Set techniques

DEMIAN WASSERMAN\textsuperscript{1}, MARTA E. MEJAIL\textsuperscript{1}, JULIANA GAMBINI\textsuperscript{1}, MARÍA E. BUEMI\textsuperscript{1}

\textsuperscript{1}UBA–Universidad de Buenos Aires, FCEyN, Departamento de Computación, Ciudad Universitaria, Pab. I, 1428 Buenos Aires, República Argentina
\{dwasser, marta, jgambini, mebuemi\}@dc.uba.ar

Abstract. Active contours approach is a very used family of techniques in image analysis. This work presents 
a comparative study between two active contour approaches for image segmentation. Level sets method and de-
formable contours under B-spline representation are compared. These image segmentation methods have different 
features and are difficult to compare in terms of performance, accuracy and initialization conditions. Both are 
implemented and a way to calculate the approximation error is developed. As a conclusion of this work a theoretical 
comparison and a comparative characterization of the approximation error for each method are presented.

1 Introduction
Image feature extraction is required for several applications such as medical image analysis, audio-visual speech analy-
sis, computer assisted design and robotics.

Active contours approach applied to image segmentation was introduced by Kass [4] and it was a new approach in feature extraction. Active contours are curves that evolve within the image until they find the boundaries of the interest object. Based on the original work, different flavors have been developed in literature in order to improve stability, flexibility and accuracy.

Level sets’ active contour approach, originally presented in [6], is based on a curve representing an interface between two inmisible fluids (see [10]). The evolution of the interface is modeled in time by means of partial differential equations. This technique is used due to its flexibility and accuracy. On the other hand B-splines deformable models approach, originally developed by Blake (see [1]), segments an image finding object boundaries by B-spline curve fitting. This technique main advantages rely on the balance between accuracy and low computational cost.

The methods of level set and B-spline deformable contours have different features and are difficult to compare in terms of performance, accuracy and initialization conditions. In this work a comparative study between both active contour methods is presented. In the first place a theoretical introduction for these techniques is presented. Secondly both methods are compared over a set of random generated images measuring the approximation error, hence each method advantages and drawbacks are exhibited.

The organization of this paper is as follows: section 2 describes the level set model for contour detection, characterizes the curve evolution and presents the numerical implementation. Section 3 gives an introduction to B-Spline curve fitting, specifies the criterion used to determine the transition points and explains the region fitting algorithm. Section 4 shows the obtained results and finally, section 5 presents a comparative study of both techniques and the advantages and disadvantages of each method.

2 The level set approach for contour detection
In this section we present level set methods and their numerical representation (see [7]).

The mathematical front evolution model proposed by [7] and [9] is the following:

\[ C : [0, S] \times \mathbb{R}^+ \rightarrow \mathbb{R}^2 \]

\[ C(s, t) = (x(s, t), y(s, t)) \]

where \( s \) and \( t \) are the spatial and temporal parameters, satisfying the differential equation:

\[ C_t(s, t) = \vec{F} \]

(1)

where \( C_t \) is the time derivative and \( \vec{F} \) is a vectorial function. Consider the tangential \( \vec{C}_T \) and normal \( \vec{C}_N \) components of the curve, the equation (1) is rewritten as:

\[ C_t(s, t) = \vec{C}_T(s, t)F_T(s) + \vec{C}_N(s, t)F_N(s) \]

and in turn, according to the parameterized curve evolution theorem (see [9]), it becomes into the equation:

\[ C_t(s, t) = \vec{C}_N(s, t)F(s) \]

where \( F : \mathbb{R} \rightarrow \mathbb{R} \) is the projection of \( \vec{F} \) along the normal direction of the curve \( \vec{C}_N \).
An example of front propagation under normal curve evolution is shown in Figure 1 where curve parameterization is taken that the normals lines are external.

Given an initial curve $C_0(s)$, front propagation is modeled by following the equations:

$$C_t(s, t) = C_N(s, t)F(s)$$
$$C(s, 0) = C_0(s)$$

2.1 Curve evolution characterization

The speed function $F$ of any point in the evolving curve, can be characterized as depending on three kinds of properties (see [9]), local curve properties (normal direction, curvature), global curve properties (area) and properties that are external to the curve.

Curve evolution in time is formulated using the *fundamental theorem of plane curves* ([3]), which states that two plane curves which have the same curvature differ only by an Euclidean motion, i.e. translation and rotation. The equation (2) can be written as:

$$C_t(s, t) = F(k(s, t))C_N(s, t)$$

where $k(s, t)$ is the curvature at any point, in a moment $t$.

2.2 Numerical implementation

A natural approach is to consider a Lagrangian formulation of the problem, obtaining the motion equations for $x(s, t)$ and $y(s, t)$. In this approach the parameterization is discretized into a set of marker particles over the domain parameterization. These particles are updated in time approximating spatial derivatives and thus changing their locations. Finally, propagating front will then be reconstructed using these particles. Nevertheless, this approach has several flaws discussed in [9]. Firstly small errors in the computed location of the particles are deeply amplified in the curvature estimation thus leading the model to numerical instability. Secondly if there is no smoothing term for the curvature (viscosity) in the differential equation, singularities will develop on the propagating front and an entropy condition must be applied in order to obtain a weak solution of the differential equation. Finally topology changes on the curve are hard to handle.

As an alternative, the level set approach proposed by Sethian and Osher in [10] and [7] represents the front $C(s, t)$ as a level set of a function $\phi$ which is one dimension higher than $C(s, t)$. Thus, given a closed curve $C(s, t)$ which evolves in time, the given front in a moment $t$ will be given by the set $\Gamma(t) = \{ \vec{x} \in \mathbb{R}^2 : \phi(\vec{x}, t) = c \}$.

Therefore an Eulerian mathematical model for front evolution is stated: let $\gamma(t) = C(s, t)$ be the front evolving along its normals with speed $F$, and let $\phi(\vec{x}, t = 0), \vec{x} \in \mathbb{R}^2$, satisfying

$$\phi(\vec{x}, t = 0) = \pm d$$

where $d$ is the signed distance from $s$ to $\gamma(t = 0)$ and the sign is chosen according to the point position inside or outside of the curve.

Thus, the evolution equation for $\phi(\vec{x}, t)$ is obtained, ([6]):

$$\phi_t + F|\nabla \phi| = 0$$

where $F$ can have several parameters including, among others, curvature or normal direction.

An advantage to this formulation is that $\phi$ will remain a function as long as $F$ is smooth, hence the level set $\Gamma(t)$ can change its topology, merge, split and develop sharp edges as $\phi$ evolves, see [7].

A second advantage concerns numerical approximations. Because $\phi$ remains a function as it evolves, a discrete grid in the domain and finite differences numerical approximations to the temporal and spatial derivatives can be used.

However, there are cases where the solution to the differential equation can lose its differentiability in a moment $t_p$ and the solution curves for $t > t_p$ lack interpretation as a propagating front. An example of this problem is presented (see [9]) for the initial curve with a constant evolution speed $F(k(s, t))$.

$$\begin{align*}
C(s, 0) &= \left( 1 - s, \frac{1 + \cos(2\pi s)}{2} \right) \\
F(k(s, t)) &= 1
\end{align*}$$

The evolution of this curve for $t = 0; 0.05; \ldots; 0.20$ is shown in Figure 2. There a “sharp corner” can be devised in $t = 0.05$, the curvature and the normal cease to exist at $(s, t) = (0.5, 0.05)$. Beyond, for $t > 0.05$ the curve passes through itself and its interpretation as an interface becomes uncertain.

Figure 1: Front propagation according to normal velocities
In order to solve this problem an entropy condition is invoked, i.e. a property based in the knowledge of the “order” that the particles in the front should have:

**Definition 1 (Entropy condition [5])** In the process of inward deformation, once a point is dislodged from the inner part of the curve, it remains disjoint from it forever. Similarly, in the process of outward deformation, once a point becomes part of the interior of the curve, it remains a part of it forever.

Following [10] choosing a speed function \( F(k) = 1 - \epsilon k, \epsilon \in \mathbb{R}, \epsilon > 0 \) (viscosity solution) makes the curve evolution respect the entropy condition, thus the curve will remain smooth and the constant speed solution is obtained when \( \epsilon \to 0 \). Replacing this function in the equation (2):

\[
\phi_t + (1 - \epsilon k) |\nabla \phi| = 0
\]

then

\[
\phi_t + H_0(\phi_x, \phi_y, t) = \epsilon H_1(\phi_{xx}, \phi_{yy}, \phi_{xy}, \phi_x, \phi_y, t)
\]

where

\[
H_0(\phi_x, \phi_y, t) = \sqrt{\phi_x^2 + \phi_y^2}
\]

\[
H_1(\phi_{xx}, \phi_{yy}, \phi_{xy}, \phi_x, \phi_y, t) = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}} \sqrt{\phi_x^2 + \phi_y^2}
\]

This is a Hamilton-Jacobi second order equation whose numerical solution can be found in [7].

Taking a curvature linear dependent velocity \( F(k) = F_0 - F_1 k, F_0, F_1 \in \mathbb{R} \), the equation 2 can be rewritten:

\[
\phi_t + (F_0 - F_1 k) |\nabla \phi| = 0
\]

\[
\phi_t + F_0 |\nabla \phi| - F_1 k |\nabla \phi| = 0
\]  \hspace{1cm} (3)

The second term of the equation (3) is known as advection or constant movement term, whereas the thirds is known as diffusion or curve smoothing term (see [10]).

Conversely a discretization, where \( \Delta t \) is the temporal step, \( \phi^n_{ij} \) is the value of the function \( \phi \) at \((i, j)\), in the moment \( n \), is taken, namely.

\[
\frac{\phi^{n+1}_{ij} - \phi^n_{ij}}{\Delta t} + (F_0 \nabla_{ij} \phi^n_{ij}) - F_1 k (\nabla_{ij} \phi^n_{ij}) = 0.
\]

The discretized advection term, \( F_0 (\nabla_{ij} \phi^n_{ij}) \) is computed with an *upwind scheme* \n
\[
\{(max(F_0,0)\nabla_{ij} \phi^n_{ij})^2 + (min(F_0,0)\nabla_{ij} \phi^n_{ij})^2\}^{1/2}
\]

where

\[
\nabla_{ij}^x = [max(D_{ij}^{x-1},0) + min(D_{ij}^{x+1},0)]
\]

\[
\nabla_{ij}^y = [max(D_{ij}^{y-1},0) + min(D_{ij}^{y+1},0)]
\]

The forward and backward finite difference approximation to the spatial derivatives over \( x \), \( D_{ij}^{x+} \) and \( D_{ij}^{x-} \) are stated as:

\[
D_{ij}^{x+} = \frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x}
\]

\[
D_{ij}^{x-} = \frac{\phi_{i-1,j} - \phi_{i,j}}{\Delta x}
\]

where \( \Delta x \) is the spatial step over \( x \), the derivatives over \( y \) analogously defined.

The diffusion term \( F_1 k |\nabla \phi| \) is discretized using central differences

\[
F_1 k \sqrt{(D_{ij}^x)^2 + (D_{ij}^y)^2}
\]

where the approximation to the derivative of \( \phi \) over \( x \) is stated as:

\[
D_{ij}^x = \frac{\phi_{i+1,j} - \phi_{i-1,j}}{2\Delta x}
\]

and the derivative over \( y \) is analogously defined. Finally the curvature \( k \) defined using the mean curvature:

\[
k = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}}
\]

is discretized using central finite differences for first and second derivatives:

\[
k_{ij} = \frac{D_{ij}^{xx}(D_{ij}^x)^2 - D_{ij}^{xy}D_{ij}^x + D_{ij}^{yy}(D_{ij}^y)^2}{((D_{ij}^x)^2 + (D_{ij}^y)^2)^{3/2}}.
\]
where the central approximation to the second derivative over $x$ is defined as

$$D_{ij}^{xx} = \frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2}$$

and the approximations to the second derivatives over $y$ and over $x$ and $y$ are defined analogously.

The third advantage of this formulation is that the intrinsic geometric properties of the front may be easily determined from the level set function $\phi$ (i.e., curvature, enclosed area, etc.).

Finally the fourth advantage is the easiness to generalize this approach to higher dimensional models by changing $\phi$’s domain.

### 2.3 Boundary detection using front propagation

In this section a description of the level set technique application for boundary detection is given. In the first place the interface is taken as representing the boundary of a shape that evolves in time. Given that the goal is to detect several objects’ boundary in an image, the evolution of the interface must be forced to a halt when it reaches the boundary. Finally the contour of the object is defined as the set of points that belong to the interface when all of them have come to a halt.

The definition of a curve evolution speed function $F$ such that the evolution stops when the curve reaches the boundary is the objective now. Thus the equation (3) is restated:

$$\phi_t(\vec{x}, t) + g(\vec{x}) (F_0 |\nabla \phi(\vec{x}, t)| - F_1 k(\vec{x}, t) |\nabla \phi(\vec{x}, t)|) = 0$$

where $g(\vec{x})$ is a function that becomes zero as long as the interface approaches a border and is at the most one at any other point of the domain. If $I$ is the image a classical choice for $g$ is (16)

$$g(\vec{x}) = \frac{1}{1 + |\nabla (G_\sigma * I(\vec{x}))|^2}$$

where $G_\sigma * I$ is the convolution of the image $I$ with a filter $G_\sigma$ which removes the image noise.

Following the revision stated in [11], the equation (4), has two segmentation problems: firstly $g$ does not suffice to stop completely the interface at the boundary, and secondly if the interface crosses the border there is no term which can pull it back with enough strength for it to came back to the boundary. In [2] a new term is added which makes the algorithm more robust and solves this problems:

$$\phi_t(\vec{x}, t) + g(\vec{x}) (F_0 |\nabla \phi(\vec{x}, t)| - F_1 k(\vec{x}, t) |\nabla \phi(\vec{x}, t)|) + \nabla g(\vec{x}) \cdot \nabla \phi(\vec{x}, t) = 0$$

### 3 Boundary detection under B-spline Deformable Contours

In this section a contour extraction B-spline curve based method is developed. A brief theoretical review of B-spline representation and a boundary detection method are presented.

#### 3.1 B-spline Representation

A brief theoretical review of B-spline representation of contours is presented; for more details see [1, 8].

Let $\{Q_0, ..., Q_{N_B-1}\}$ be a set of control points, where $Q_n = (x_n, y_n)^T \in \mathbb{R}^2$, $0 \leq n \leq N_B - 1$, and let $\{s_0 < s_1 < s_2 < \cdots < s_{L-1}\} \subset \mathbb{R}$ be a set of $L$ knots. A B-spline curve of order $d$ is defined as a weighted sum of $N_B$ polynomial basis functions $B_{n,d}(s)$ of degree $d - 1$, within each interval $[s_i, s_{i+1}]$ with $0 \leq i \leq L - 1$. The constructed spline function is $r(s) = (x(s), y(s))^T$, $0 \leq s \leq L - 1$, being

$$r(s) = \sum_{n=0}^{N_B-1} B_{n,d}(s) Q_n,$$

and

$$x(s) = B^1(s) Q^x \quad (5)$$

$$y(s) = B^1(s) Q^y \quad (6)$$

where the basis functions vector $B(s)$ of $N_B$ components is given by $B(s) = (B_{0,d}(s), ..., B_{N_B-1,d}(s))^T$. The weight vectors $Q^x$ and $Q^y$ give the first and second components of the $Q_n$, respectively.

The curves used in this work are closed of order $d = 3$ or $d = 4$ specified by periodic B-spline basis functions.

The problem of determining a polygon that generates a fitting B-spline curve with known number of control points, $N_B$, was studied by [8]. We now present a brief review of this subject.

A set of $k$ data points in the image plane is given by $\{D_0, D_1, ..., D_{k-1}\}$, where $D_i = (x_i, y_i)^T$, $i = 0, ..., k - 1$, and the spline curve that best-fits them is sought. Then, by equations 5 and 6, the components $D_i$ must satisfy

$$x_i = B^1(t_i) Q^x,$$

$$y_i = B^1(t_i) Q^y,$$

for certain values of $t_i$, where $i = 0, 0, ..., k - 1$, and $N_B \leq k$.

This linear system is more compactly written in matrix form as $D = K(Q^x, Q^y)$, where the $k \times N_B$ elements of the real matrix $K$ are given by $K_{ij} = B_{j,d}(t_i)$, with $i = 0, ..., k - 1$, $j = 0, ..., N_B - 1$, and $D = (D_0, D_1, ..., D_k)^T$. In the most general case $N_B < k$ and, therefore, $K$ is not a square matrix. In this case, the pseudo-inverse matrix form $(Q^x, Q^y)^T = K^+ D$ is used to find the
Once the initial search zone is determined the centroid is
of which are control points that generate a B-spline curve.
search area, which is specified by polygons, the vertexes
with the form $a_\theta$ its respective boundaries

In this section we describe an algorithm developed for bound-
ary detection

3.2 Boundary detection

In this section we describe an algorithm developed for bound-
ary detection using B-spline deformable contours. Let $E$ be a
scene made up by the background $B$ and a region $R$ with
its respective boundaries $\partial R$. We want to find the curve
$C_B$ that fits boundary $\partial R$ in the image. We define an initial
search area, which is specified by polygons, the vertexes
of which are control points that generate a B-spline curve.
Once the initial search zone is determined the centroid is
calculated.

If a point belongs to the object boundary, then a sam-
ple taken from the neighborhood of that point exhibits a
change in the intensity level of image and it is considered to
be a transition point. Then $N$ segments $a_i$, $i \in \{1, ..., N\}$
with the form $a_i = OP_i$ are considered. Here $O$ is the
centroid of the initial region, the extreme $P_i$ is a point out-
side of the region and $\theta_i = \text{ang} (a_i, a_{i+1})$ $\forall i$ is the angle
between two consecutive segments as it can be seen in Fig-
ure 3. The segment $a_i$ is an array of $m$ elements coming from
a discretization of the straight line on the image. In this work all the experimental results are binary images, so the border point on each segment is found convolving the
data of the segment with a mask given by $[-1, 1]$. Once the
set of border points $A = \{b_1, ..., b_N\}$ is found, the method
builds the interpolating B-spline curve as its explained in
section 3.1. This method can generate redundant contour
points, and generally it does not need to use all of them
for obtaining the interpolating curve. An algorithm to find
significant boundary points in the description of the same

Figure 3: Radial straight lines $a_i$ separated an angle $\theta_i$.

B-spline fitting curve. A useful set of values for the param-
eters $\{t_0, \ldots, t_{k-1}\}$ is given by

$$t_0 = 0, \quad t_\ell = \frac{\sum_{i=1}^{\ell} \|D_i - D_{i-1}\|}{\sum_{i=1}^{\ell} \|D_i - D_{i-1}\|}, \ell \geq 1.$$ 

The knot set to build the B-spline basis functions is arbitrar-
ily chosen.

We consider the polar coordinates of each point $b_i \in A$:

$$b_i = (\rho_i, \alpha_i)$$

where $\rho_i$ and $\alpha_i$ are the length and the orientation of $b_i$, respectively. $A$ is an ordering sequence, then each point $b_i$
has only two geometrical neighbors $b_{i-1}$ and $b_{i+1}$.

If $b_{i-1}$, $b_i$ and $b_{i+1}$ are consecutive points then there
is a critical point if

$$\text{sign}(\rho_i - \rho_{i-1}) \neq \text{sign}(\rho_{i+1} - \rho_i)$$

or

$$\text{sign}(\alpha_i - \alpha_{i-1}) \neq \text{sign}(\alpha_{i+1} - \alpha_i)$$

hold. In this manner, a subset

$$M = \{b_{c_1}, \ldots, b_{c_m}\}$$

of critical points is obtained. Then, for each pair of adja-
cent points $b_{c_j}, b_{c_{j+1}}$ of $M$, an interpolating straight line $S_j$
is considered. Then the distances $d_i$ between $b_j \in A - M$
and $S_j$ are computed. The points with maximum and mini-
mum distances that verify the following condition:

$$\text{sign}(d_i - d_{i-1}) \neq \text{sign}(d_{i+1} - d_i)$$

for $c_j < i < c_{j+1}$, are selected.

We define $\Gamma = \{b_0, b_1, \ldots, b_{c-1}\}$ the set of $r$ points
that verify equation (3.2).

Let $C_0 = \text{sort}(M \cup \Gamma)$ be the ordering set of pseudo
critical points. In order to select the critical points from $C_0$,
we define a critical level $l$ for each point $b_i$:

$$l(b_i) = \Delta \text{area}(b_{i-1}, b_i, b_{i+1}).$$

Finally, taking the critical point set

$$D = \{b_i \in C_0 / l(b_i) \geq L\},$$

where $L$ is a threshold defined as $L = \min \left\{ \frac{l(b_i)}{A}, b_i \in A \right\}$,
a new interpolating B-spline curve is obtained.
curve are considered. Let boundary of the region of interest and where 
\( \alpha \) of petals and \( 2 \) and \( \beta \) are given by:

\[
\rho(s; \alpha, \beta, \gamma) = \alpha - \gamma \cos(\theta(s))
\]

\[
\theta(s) = \frac{2\pi}{S}
\]

Figured 4: Parameters \( \alpha \) and \( \gamma \) as seen on the flower equation (7)

4 Results

This section presents the obtained results of the segmentation methods applied to images which are randomly generated. The approximation error for both methods, B-spline and level set approach is characterized and compared.

The algorithms are applied to 500 synthetic images (flowers) generated by means of the parametric curve in polar coordinates:

\[
f(s; \alpha, \beta, \gamma) = (\theta(s), \rho(s; \alpha, \beta, \gamma)), s \in [0, S]
\]

where \( \alpha \) is the mean radius of the flower, \( \beta \) is the number of petals and \( 2 \gamma \) is the height of the petals, see Figure 4. \( \alpha \), \( \beta \) and \( \gamma \) are considered independent random variables with uniform distribution. The variation ranges for each variable are given by: \( \alpha \in [5, 20], \beta \in [15, 50], \gamma \in [2, 10] \).

In order to compute the approximation error, \( \partial R \) the boundary of the region of interest and \( C \) the approximating curve are considered. Let \( a_j \) be the \( j \)th radial straight line, see Figure 3, which equation is given by:

\[
a_j : \lambda \vec{u}_j + \vec{O} \quad j = 1, \ldots, N
\]

where \( \vec{u}_j \) is the straight line unit vector and \( \vec{O} \) is the centroid of the region \( R \). Let \( \tilde{V}_j \) the intersection point between the curve \( C \) and the straight line \( a_j \), and \( \tilde{W}_j \) the intersection point between the boundary \( \partial R \) and the straight line \( a_j \):

\[
a_j \cap C = \tilde{V}_j
\]

\[
a_j \cap \partial R = \tilde{W}_j
\]

Then, a distance between \( C \) and \( \partial R \) is defined as

\[
d(\partial R, C) = \frac{1}{N} \sqrt{\sum_{j=1}^{N} \| \tilde{V}_j - \tilde{W}_j \|^2} \tag{8}
\]

Figure 5 (bs1) to (bs4) shows boundary detection with B-spline method and (ls1) to (ls4) boundary detection with level set method, in both cases as a white curve. The two methods applied to four synthetic images generated from the equation (7). Shapes from (bs1) and (ls1) generated with \( \alpha = 10, \beta = 38 \) and \( \gamma = 5 \), (bs2) and (ls2) with \( \alpha = 6, \beta = 50 \) and \( \gamma = 10 \), (bs3) and (ls3) with \( \alpha = 8, \beta = 38 \) and \( \gamma = 9 \) and (bs4) and (ls4) with \( \alpha = 12, \beta = 15 \) and \( \gamma = 3 \). Figure 6 shows the histograms of the B-spline and level set approximation error.

5 Conclusions and Further Work

Comparing the exhibited results in Section 4 from a visual perspective the segmentation achieved with level sets method is more accurate than the one achieved with B-splines. Furthermore, using the measure defined in (8) over 500 generated samples, we noticed that more than 50% of the segmented shapes using the level sets method, presented an error less or equal than 0.05 and more than 85% an error less or equal than 0.1. On the other hand, less than 20% of the samples segmented using the B-spline technique presented an error lower than 0.1 and less than 84% were segmented with an error lower than 0.3. These results can be seen in table 5, corresponding to the accumulated frequencies of the approximation errors of the B-spline and level set methods, respectively. This fact can also be noticed on the error histograms shown in Figure 6.

In this work we established a comparing method for segmentation techniques. Finally, as a conclusion of this work we present in a comparative manner the advantages and drawbacks of each segmentation method compared:

- Level set method

1. Boundary fitting accuracy depends only on the ratio between the advection and diffusion parameters. Furthermore, Eulerian evolution instead of Lagrangian implies higher accuracy on the differential equation resolution, hence better fitting results are achieved.
2. Front topology can change during the evolution, thus supervised initialization is not required.
3. The result is a discretization over the domain of a function whose zero level set is the boundary. Therefore, in order to obtain a continuous representation of the boundary it is necessary to extract the isocontour and interpolate it.
Figure 5: The B-spline approximation curve (bs1) to (bs4). Level set approximation curve (ls1) to (ls4). (bs1) and (ls1) generated with $\alpha = 10$, $\beta = 38$ and $\gamma = 5$, (bs2) and (ls2) with $\alpha = 6$, $\beta = 50$ and $\gamma = 10$, (bs3) and (ls3) with $\alpha = 8$, $\beta = 38$ and $\gamma = 9$ and (bs4) and (ls4) with $\alpha = 12$, $\beta = 15$ and $\gamma = 3$.

4. The smoothing level of corners depends on the diffusion parameter and can be omitted. Then, this approach can handle sharp edges effortlessly.

Figure 6: Histogram of the approximation errors for the B-spline and level set methods.

5. High computational cost.

6. Easy to generalize to 3D and more dimensional models. The mathematical formulation has a direct implementation.

- B-spline

1. The number of control point and the B-spline order have to be known a priori and determines the accuracy of the results.

2. It is mandatory to specify an initial area of interest, often in a supervised manner, for each contour in the image. In addition, the topology of the solution must be known beforehand.

3. The result is a parametric equation of the boundary determined by a set of control points.

4. Smoothes out corners.

5. Low computational cost.

Finally, as it was exposed previously in this section, the level set method is more accurate than the B-spline method.

As further work we propose to improve the error characterization of both methods by performing this analysis on samples of synthetic and real textured images, hence partitioning the sample sets according to the geometrical properties of the shapes. In order to decide which method is more convenient for a particular problem, a statistical description of the approximation error for each method will be studied.

References

Table 1: algo

<table>
<thead>
<tr>
<th>d</th>
<th>B-spline</th>
<th>Level set</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
<td>0</td>
<td>0.592</td>
</tr>
<tr>
<td>0.1</td>
<td>0.197</td>
<td>0.853</td>
</tr>
<tr>
<td>0.15</td>
<td>0.495</td>
<td>0.890</td>
</tr>
<tr>
<td>0.2</td>
<td>0.683</td>
<td>0.895</td>
</tr>
<tr>
<td>0.25</td>
<td>0.770</td>
<td>0.908</td>
</tr>
<tr>
<td>0.3</td>
<td>0.839</td>
<td>0.913</td>
</tr>
<tr>
<td>0.35</td>
<td>0.858</td>
<td>0.922</td>
</tr>
<tr>
<td>0.4</td>
<td>0.872</td>
<td>0.936</td>
</tr>
<tr>
<td>0.45</td>
<td>0.881</td>
<td>0.945</td>
</tr>
<tr>
<td>0.5</td>
<td>0.885</td>
<td>0.950</td>
</tr>
<tr>
<td>0.55</td>
<td>0.885</td>
<td>0.954</td>
</tr>
<tr>
<td>0.6</td>
<td>0.904</td>
<td>0.959</td>
</tr>
<tr>
<td>0.65</td>
<td>0.931</td>
<td>0.968</td>
</tr>
<tr>
<td>0.7</td>
<td>0.950</td>
<td>0.968</td>
</tr>
<tr>
<td>0.75</td>
<td>0.963</td>
<td>0.977</td>
</tr>
<tr>
<td>0.8</td>
<td>0.972</td>
<td>0.977</td>
</tr>
<tr>
<td>0.85</td>
<td>0.982</td>
<td>0.977</td>
</tr>
<tr>
<td>0.9</td>
<td>0.991</td>
<td>0.977</td>
</tr>
<tr>
<td>0.95</td>
<td>1</td>
<td>0.977</td>
</tr>
<tr>
<td>≥ 1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>


