FACE: Fast Active-Contour Curvature-based Evolution

Daniele D. Giusto, Francesco Massidda, Cristian Perra(*)

Department of Electrical and Electronic Engineering, University of Cagliari, Piazza D'Armi, Cagliari 09123, Italy

Abstract

This paper presents an active contour model for fast object segmentation called FACE. A novel energy term that takes into account the computational complexity of the active contour is introduced together with related constraints and minimization procedure. The described process is based on the regularization and optimization of the active contour control points position. The trade-off between computational complexity and final contour accuracy is based on curvature estimation. The result is a fast active contour convergence towards desired object boundaries. This method can be combined with most of the other active contour approaches presented in literature, thanks to the independence between the computational minimization process and the classical active contour minimization process. The object segmentation procedure can be automatic or semiautomatic depending on the original image complexity. Several tests and experiments have been realized. Results show improvements in terms of computational time reduction when compared with other similar active contour models.

Keywords: Image segmentation; Active contour

(*)Corresponding author
Cristian Perra
DIEE - *Department of Electrical and Electronic Engineering,*
University of Cagliari
Piazza D'Armi, Cagliari 09123,
Tel. +39 070 6755866
Fax. +39 070 6755890
E-mail address: cperra@diee.unica.it

1. Introduction

Automatic object boundaries location (object shape identification) is a challenging field in computer vision. Approaches based on several techniques have attracted many researchers in the last two decades. In particular, a family of methods called "deformable models" was introduced by Terzopoulos et al. [1] in 1988. "*The mathematical foundations of deformable models represent the confluence of geometry, physics, and approximation theory.*" [1,2,3,10]. In fact, the object shape is represented by geometry, shape variation over time and space is imposed by physical constraints, and a formal support is provided by the theory of optimal approximation.

Active contours or *snakes* are deformable models (deformable contours or curves) confined to the plane. The active contour model algorithm, proposed by Kass, Witkin, Terzopoulos [4], locks a deformable contour onto features of interest (e.g. object boundaries) within an image. This deformable model moves towards contours guided by external energies, forces depending from image boundaries, and it is guided by internal energies, forces that tend to regularize the related parametric curves over time and space. Such algorithm, starting from a rough approximation of an object boundary in an image, can provide a closed contour representing the expected object boundary.

The objective is to identify the contour that is solution of an energy minimization problem. An appropriate choice of the energy functions should provide a contour solution corresponding to the object boundary.

Nevertheless, the original model proposed by Kass et al. [4] has some drawbacks. The noise within an image brings to false local minima tending to trap the snake before the desired solution (object boundary) is achieved. The snake is attracted by almost any edge within the image plane, whether or not it belongs to the desired object. Furthermore, external forces are localized close to the object boundaries and, in order to obtain an acceptable solution, the initial contour must be very close to the desired object contour otherwise it is difficult to make the active contour move towards the object. Another drawback is the difficulty encountered in making active contours move into boundary concavities or indentation. Moreover, internal forces tend to regularize the shape using an elastic force between control points, and then the active contour can collapse into a single point. An accurate parameters setting is needed for avoiding such problem and often it drives to ad-hoc models able to segment only particular set of images.

Many researchers have proposed different methods in order to overcome the limitations of the original approach [23,24,25,26,27,28,29,30,31]. The solution for the parametric curve model [4], uses a minimization method based on discrete partial derivatives in space and time evaluated with finite differences approximation methods. The intrinsic difficulty, of this model, in reaching the stability was pointed out and faced by an alternative model proposed in [7], based on dynamic programming that allows the inclusion of hard constraints in the original model. Furthermore, using the dynamic addition and subtraction of control points this method achieves superior results. Since the addition or deletion of points requires an inversion of an n x n matrix, this class of methods have a computational load proportional to $O(nm^3)$ where n is the number of control points in the active contour that can move in a *m-size* neighborhood at each iteration [12].

An accurate final contour requires a high number of control points. Managing lot of points in the numerical solution of the energy minimization process drives to very slow iterative algorithms.

Some variations to this approach have been proposed in literature in order to reduce the computational load. For example in [8], authors obtain an algorithm having computational complexity $O(nm)$. The problems of the original Kass model was faced also in [9] by Berger that introduce the idea of "*Snake Growing*". A single initial active contour starts to deform and then it is divided into multiple snakes. Next an energy evaluation is performed. Low-energy snakes continue their evolution, while high-energy snakes are deleted. Remaining low-energy contours are used to initialize the snake in the following time step. With the Berger approach the initialization problems

were overcome, the iterative algorithm converges and numerical solution is stable but the algorithm increases the computational load when compared with previous methods.
A different approach was proposed by Cohen et al. in [5,11]. In this model, a pressure force is added to the contour interior considering the snake as an inflated balloon. The drawback of false local minima caused by random edges is partially solved by this method. They tried to overcome the collapse towards a single point of the active contour introducing an internal pressure energy that tends to inflate the balloon and allows the expansion of the model, fighting the contracting force. Unfortunately, a constant pressure term introduced new problems, for instance, the initial placement of the snake had to be within the target. Some solutions were proposed in order to solve this problem, like a dynamic pressure model [13].
External forces defined as the negative gradient of a Euclidean distance map were widely used [5]. Xu and Price [6] proposed a new class of external forces for active contour models that face the problems related to the image noise and the short capture range of the standard external forces. They called this field *gradient vector flow* (GVF). The GVF is computed as the gradient vectors diffusion of a gray level image. The methods described require an appropriate contour initialization and, for this reason, they are not suited for automatic boundary location application.
The Level Set approach [14,15,16,17,32,33] provides a powerful alternative with respect to snake models. In literature, they are also called dynamic contours, while active contours based on original Kass et al. procedure are known as parametric active contours. The idea is the use of a front propagating and a convergence speed that depends on the characteristics of images and on front curvature. As previously mentioned, Osher [16] shows the limitation of Kass model. In particular, Osher proved that parametric contours need a periodic regridding. Furthermore, the parametric models cannot face topology changes without ad-hoc solutions [22]. The Level Set methods are alternative useful models for image segmentation. However, in spite of an elegant solution, Level Sets increase the computational load. Researchers propose different solutions in order to overcome speed limitations [41,42,43,44,45] but some problems still remain, in particular when Level Set are applied to low level gradient images [34,35,36,37,38,39,40].
This paper presents a fast active-contour evolution scheme (FACE) for image segmentation, based on curvature analysis and GVF field. The proposed method can be combined with almost all the other active contour approaches, thanks to the independence between the computational minimization process and the classical active contour model concerning the external energy, the internal energy and the iterative active contour minimization phase. Active contour segmentation can be very helpful in locating object boundaries in several fields of application like, for example, medical analysis, video tracking applications, object recognition.
Traditional active contours do not have large capture range because the external forces decrease rapidly as the distance from the object contour increases. The convergence to boundary concavities can be achieved by GVF and the proposed FACE but not by the traditional or the distance map forces. GVF field represents a precise method for boundaries extraction but it needs more iterations than the other methods to expand the field in the entire image and to be independent from the boundary initialization. The fastest computation can be achieved by the traditional field, with the limitations previously mentioned, and by the FACE field, with a more accurate final contour. The FACE external forces can be viewed as a combination between the GFV external forces [6] and the potential forces derived from Euclidean distance map [5].
As it will be described in detail in the next sections, the GVF field is computed close to the desired boundaries while for the rest of the image the field is computed with a distance map functional. Active contour computational complexity is reduced using as external forces a composition of GVF and distance map fields. Therefore, the concavities-convergence characteristic of GVF is locally preserved. The iterative evolution of the snake is performed like in classical methods. Finally, during this evolution, the proposed curvature analysis is computed in order to minimize the number of control points needed to have the differences between the optimized and non-optimized parametric curve under a given threshold.

Several tests show improvements in terms of convergence speed and final boundaries accuracy when compared to similar methods or to the same method without computational energy regularization/minimization. Differences between optimized and non-optimized standard active contour can be controlled and minimized at each iteration step. Furthermore, the application field is quite large. Appropriate setting of the curvature evaluation parameters allows the possibility of achieving minimal differences between optimized and original snake configuration, required for example in medical applications or to speed up the process for real-time applications like automatic object tracking. Finally, the segmentation process can be automatic or semiautomatic depending on the original image complexity.
The paper is organized as follows. Section 2 discusses the related work and the methods and algorithms that are the starting point for the proposed model, presented in Section 3. The experimental results are presented in section 4. Section 5 concludes this paper.

2. Related work

In Kass et al. [4], the active contour, also called snake, is defined in the (x, y) plane, through the function $\Omega = [0,1] \longrightarrow R^2$, and is represented by the parametric curve $V(s) = (x(s), y(s))$ where $s = [0,1]$. The related discrete deformable model, the active contour, is changed, in subsequent iterative steps, by deformations guided and limited in order to minimize the following functional:

$$E_{snake} = \int_0^1 E_{snake}(V(s))ds = \int_0^1 \left(E_{\text{int}}(V(s)) + E_{image}(V(s)) + E_{con}(V(s))\right)ds \,. \tag{1}$$

This is defined as a sum of energy terms. These energies can be divided into two main groups, internal energies, function of the same $V(s)$ contour at a certain time step, and external, functions that take into account characteristics of the processed images such as edges and luminance peaks. The external energy drives the active contour towards the desired points or boundaries within the image plane. The internal energy tries to keep the snake connected and consistent, preserving characteristics like steadiness, smoothness, tension and stiffness.
The iterative minimization process of Eq. (1) deforms dynamically the parametric curve until a minimum is found that corresponds to the final active contour $V(s)$ that better matches the desired boundaries within the image:
Generally, the internal energy is the sum of tension and stiffness (curvature) terms:

$$E_{\text{int}}(V(s)) = \frac{1}{2}\alpha(s)\left|V_s(s)\right|^2 + \frac{1}{2}\beta(s)\left|V_{ss}(s)\right|^2 \,. \tag{2}$$

The more general expression for the image energy is shown in Eq. (3), and represents, the external forces that drive the contour towards, respectively, high luminance image points, boundaries and high curvature image areas, that are the ends of a smoothed version of the image.

$$E_{image}(V(s)) = w_{line}E_{line} + w_{edge}E_{edge} + w_{term}E_{term} \tag{3}$$

The last term in Eq. (1) comes from user-defined constraints. Those external forces tend to keep cohesion between the control points of the snake and particular points in the image plane.
Frequently, the second and the third terms in Eq. (1) are joined together in a single external energy term in Eq. (4):

$$E_{ext}(V(s)) = E_{image}(V(s)) + E_{con}(V(s)). \quad (4)$$

Finally, replacing Eq. (2) and Eq. (4) in Eq. (1), the snake energy becomes

$$E_{snake} = \int_0^1 E_{snake}(V(s))ds = \int_0^1 F(V, V_s, V_{ss})ds \quad (5)$$

where

$$F(V, V_s, V_{ss}) = E_{ext}(V(s)) + (\frac{1}{2}\alpha(s)\left|V_s(s)\right|^2 + \frac{1}{2}\beta(s)\left|V_{ss}(s)\right|^2). \quad (6)$$

The solution of Eq. (6) is given by the Euler-Lagrange equation:

$$F_V - \frac{\partial F_{V_s}}{\partial s} + \frac{\partial^2 F_{V_{ss}}}{\partial s^2} = 0 \quad (7)$$

where

$$F_V = \frac{\partial}{\partial v}E_{ext}(v(s)),\ F_{V_s} = \alpha(s)V_s(s),\ F_{V_{ss}} = \beta(s)V_{ss}(s). \quad (8)$$

Eq. (7) becomes

$$\frac{\partial}{\partial v}[E_{ext}(V(s))] - \frac{\partial}{\partial s}[(\alpha(s)V_s(s))] + \frac{\partial^2}{\partial s^2}[\beta(s)V_{ss}(s)] = 0. \quad (9)$$

Eq. (9) can be written as

$$\nabla E_{ext} - \alpha V_{ss} + \beta V_{ssss} = 0 \quad (10)$$

assuming α and β parameters independent from *s*.
Eq. (10) can be solved using numerical methods and approximations.
The temporal snake evolution is given by

$$x^t = (A)^{-1}(\gamma \cdot x^{t-1} - f_x(x^{t-1}, y^{t-1})). \quad (11)$$

as explained in [3], where $A = A' + \gamma$ is a cyclic symmetric pentadiagonal banded matrix of order N^t, number of control points at *t*th iteration. Eq. (11) becomes

$$x^t = A^{-1}(\gamma \cdot x^{t-1} - f_x(x, y)) \quad (12)$$

for time invariant fields.
The solution to the minimization problem is reached when the equilibrium for Eq. (11) or Eq. (12) is obtained.

Standard approach sets $w_{line} = 0$ and $w_{term} = 0$ for the image energy (3). Furthermore, in Eq. (4) $E_{con} = 0$ is often imposed. In this case, the final external energy expression is

$$E_{ext}(V(s)) = E_{image}(V(s)) = f(V(s)) = -w_{edge}\left|\nabla I(x, y)\right|^2 . \tag{13}$$

2.1 *Gradient Vector Field*

The energy field obtained from Eq. (13) is not diffuse over the entire image plane but it is localized near to the object contours. The starting active contour must be close to the desired object boundary otherwise the control points cannot be guided by the energy field. Furthermore, the local nature of this energy field does not allow the active contour entering into boundary concavities.
Gradient Vector Flow (GVF) implementation [6] use a more complex but more effective approach. An edge map of the image *I(x,y)* is computed:

$$f = f(I(x, y)) . \tag{14}$$

The GVF uses a vector field depending on the image gradient, with single vectors pointing towards object boundary:

$$f = -E_{ext}(V(s)) \tag{15}$$

that is

$$f = -[-\left|\nabla I(x, y)\right|^2] . \tag{16}$$

The edge map can be obtained also from other contour extraction algorithms such as the well-known Sobel, Roberts or Canny filters.
An iterative approach is used to spread the local field in the entire image plane. The GVF is defined as the vector field $\mathbf{V}(x, y) = [u(x, y), v(x, y)]$ that minimizes the energetic functional

$$\varepsilon = \iint \left(\mu \cdot (u_x{}^2 + u_y{}^2 + v_x{}^2 + v_y{}^2) + \left|\nabla f\right|^2 \left|\mathbf{V} - \nabla f\right|^2 \right) dxdy . \tag{17}$$

The expression is composed by two terms. When the functional is evaluated in smoothed zones of the image, where $\left|\nabla f\right|$ is small, it is controlled by the first term that is the sums of the square of partial derivative of vector field $\mathbf{V}(x, y)$. When, vice versa, $\left|\nabla f\right|$ is large, the second term is greater, and the minimization is reached for $\mathbf{V} = \left|\nabla f\right|$. With this approach, the final vector field is close to the image gradient, where the same gradient is great and a slow variation of the same field is reached in smoothed areas.
The equation is solved using variational methods and solving the Euler equation

$$\begin{aligned} \mu \cdot \nabla^2 u - (u - f_x)(f_x{}^2 + f_y{}^2) = 0 \\ \mu \cdot \nabla^2 v - (v - f_y)(f_x{}^2 + f_y{}^2) = 0 \end{aligned} \tag{18}$$

The main advantage of the GVF field is its diffusion over the entire image, allowing an initialization of the snake far from the desired boundaries. Moreover, such diffusion allows the active contour entering into boundary concavities.

2.2 Extended Vector Field

In order to reduce the great computational load needed by GVF, a variation to the original implementation has been introduced in [18]. The purpose was to increase the segmentation process speed, preserving the GVF convergence characteristics. The procedure was originally proposed in connection with the *Interframe Moving Masks* [19]. The basic idea is the construction of a global extended external energy field (EVF) that allows a fast diffusion over the entire image and locally preserves the GVF strengths: the possibility to initialize the snake far from the desired boundary and the convergence within the concavities. In order to minimize Eq. (17), authors use the standard GVF with a limited number of iterations. The external field diffusion is completed introducing an additional field that is function of the starting control point positions. An appropriate definition of the additional field can increase the convergence speed of the process far from desired contour.
EVF implementation uses a Sobel operator instead of the gradient of the image (Eq. (16)), in order to produce better edge map.
It should be observed that a minimum number of iterations, related to the size and nature of images, is needed in order to allow that active contour enters into concavities. The greater the iteration number is, the greater the computational load and time is. The last section will show how a fast convergence speed is possible, even increasing GVF iteration number.

2.3 Curvature Estimation

Three active contour control points i, i-1, and i-2 are typically used to estimate the curvature of the active contour for a given point *i* at each *k*th iterative step [4,6,7]:

$$\frac{d^2V_i}{ds^2} = v_{i-1} - 2v_i + v_{i+1} = (x_{i-1} - 2x_i + x_{i+1}) + (y_{i-1} - 2y_i + y_{i+1}) \tag{19}$$

where control points are evenly spaced with unitary intervals. In the literature, several different approaches have been presented in order to obtain a better approximation. In [7], the minimization of the active contour energy is viewed as a discrete multistage decision process in a dynamic programming framework. For each stage the total energy of the active contour is computed as

$$\begin{aligned} E_t(i,j,k) = \min_{0\le m\le N}\{\ & E_t(i-1,k,m) + E_{ext}(v_{i-1}\oplus k) + \\ & \frac{1}{2}\alpha_i\left|v_{i-1}\oplus k - v_{i-2}\oplus m\right|^2 + \\ & \frac{1}{2}\beta_i\left|v_i\oplus j - 2v_{i-1}\oplus k + v_{i-2}\oplus m\right|^2\ \} \end{aligned} \tag{20}$$

where

$$\frac{1}{2}\beta_i\left|v_i\oplus j - 2v_{i-1}\oplus k + v_{i-2}\oplus m\right|^2 \tag{21}$$

is the curvature of active contour for point *i*.

In [20], the three active contour points *i*, *i-1*, and *i-2* have been used to estimate the curvature of the active contour for point *i*. Using the notation proposed in [7] Eq. (20) becomes

$$E_t(i,j,k) = \min_{0\le m\le N}\{\ E_t(i-1,k,m) + E_{ext}(v_i \oplus k) +$$
$$\alpha\left|v_i \oplus k - v_{i-1+n} \oplus m\right|^2 + \tag{22}$$
$$\beta\left|v_{i+1} \oplus j - 2v_i \oplus k + v_{i-1+n} \oplus m\right|^2\ \}$$

where $\left|v_{i+1} \oplus j - 2v_i \oplus k + v_{i-1+n} \oplus m\right|^2$ is the curvature of the active contour for the *i-th* point estimated using the three consecutive points *i-1*, *i*, *i+1*. The curvature estimation performed with this algorithm is more accurate than the original Amini et al. [7] one.

Several other methods were introduced for changing the classical method for the curvature evaluation, in order to increase the original first order approximation or decrease the computational load needed for curvature estimation. In [21], the authors discuss about classical internal forces implementations for both tension and curvature. Considering original reasons to the introduction of these two forces, authors remark: "*tension exists to evenly space the control points*". In classical implementation however, tension acts like an elastic force that tries to push contours towards a minimum length (a point). In the original theory, *"the curvature force exists to maintain reasonably smooth contour"*, but the curvature minimization implementation forces the snake to assume a straight line characteristic. Several methods use a minimum and maximum distance approach between control points and pressure energies [5,11], in order to struggle these forces. In [21], authors propose an alternative solution. They introduce a new form for the two internal forces, introducing (Fig. 14) a tension force that goes to zero on the perpendicular of the segment AC, so that a point B is able to stay in this perpendicular, evenly spaced with respect to A and C, instead of being pushed toward the AC segment making a straight line. Furthermore, the classical formulation for the curvature defines a smooth curvature as a zero second derivative. Authors re-defined the curvature: a smooth curve is one where the third derivative (the rate of change of curvature) is constant. In this way, using five points instead of three, it is possible to measure three consecutive angles between the five points and consider the cases where these angles remain constants as a smooth curvature (like in the circle case that becomes a maximum smoothed curve, although the original formulation tries to contract the circle into a point).

A known problem is the improvement of the first order approximation, used in previous methods, for the curvature estimation. This brings to an unsatisfactory approximation of the real curvature value. Starting from the original and complete definition for the curvature, a discrete approximation of $\frac{d\theta}{ds}$ has the property that it depends linearly on the angle $\Delta\theta$ between the two vectors, $\mathbf{U}_i = (x_i - x_{i-1}, y_i - y_{i-1})$ and $\mathbf{U}_{i+1} = (x_{i+1} - x_i, y_{i+1} - y_i)$. The formula for $\Delta\theta$ is given by

$$\Delta\theta = \cos^{-1}\frac{(x_i - x_{i-1})(x_{i+1} - x_i) + (y_i - y_{i-1})(y_{i+1} - y_i)}{\sqrt{\left[(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2\right]\left[(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2\right]}}. \tag{23}$$

However, a great computational load is needed to calculate exactly the curvature at a given point with this discrete approximation of the complete definition of curvature.

In [8], strengths and weaknesses of several approaches are shown. The first order approximations and expressions for the exact curvature definition are presented. A further discrete approximation is given, starting from the assumption that the curvature parameter is the arc length. When this assumption is true, for a continuous curve the curvature is given by $|V_{ss}|$. Then the discrete approximation is given by

$$|V_{ss}| = \frac{1}{\Delta s}\sqrt{\left(\frac{\Delta x_i}{\Delta s_i} - \frac{\Delta x_{i+1}}{\Delta s_{i+1}}\right)^2 + \left(\frac{\Delta y_i}{\Delta s_i} - \frac{\Delta y_{i+1}}{\Delta s_{i+1}}\right)^2} \quad . \tag{24}$$

Authors show how this approximation is effective for the curvature estimation, although it is not rotation invariant (also, the complete and the original three-point approximation suffer this problem [8]). Finally a modification to the three point approximation is proposed. Using $\mathbf{U}_i = (x_i - x_{i-1}, y_i - y_{i-1})$ and $\mathbf{U}_{i+1} = (x_{i+1} - x_i, y_{i+1} - y_i)$, Eq. (19) can be re-written as $|\mathbf{U}_i - \mathbf{U}_{i+1}|$ and the related approximation for curvature becomes

$$\left|\frac{\mathbf{U}_i}{|\mathbf{U}_i|} - \frac{\mathbf{U}_{i+1}}{|\mathbf{U}_{i+1}|}\right| = \left(\frac{\Delta x_i}{\Delta s_i} - \frac{\Delta x_{i+1}}{\Delta s_{i+1}}\right) + \left(\frac{\Delta y_i}{\Delta s_i} - \frac{\Delta y_{i+1}}{\Delta s_{i+1}}\right) \tag{25}$$

This approximation is efficient for the curvature estimation, the related measure is scale and rotation invariant and it is faster then the complete curvature evaluation performed using Eq. (23). Authors use the three point approximation in the iterative minimization process of the active contour. Then, at the end of each iteration, a step is included which determines the curvature at each point on the new contour using Eq. (25) that better fits the original $\frac{d\theta}{ds}$ formula. If the value is a curvature maximum, β_i is set to zero for the *i*th control point in the next iteration.

There are some problems setting $\beta_i = 0$: a threshold has to be found in order to decide when a maximum exists or not. Furthermore, local maxima, displaced far from the desired edges, stop the snake evolution for that point and they do not allow a good description of the object boundaries.

In the next section, a novel method for minimizing the control point number and increasing the convergence speed is presented. The algorithm used a local curvature estimation of the active contour in each iteration step in order to perform a sort of regularization of the control point positions. As performed in [8], the idea is to use a good approximation, like Eq. (25), once for each point every *j* iteration steps, in order to reach a better control point distribution, and use the standard three points approximation of curvature, within the iterative procedure performed in the minimization process.

3. Proposed Model

For several years, many researchers have focused on the problem of computational complexity reduction for active contours. External field computation and iterative active contour functional minimization are complex processes that require a high and variable number of iteration in order to converge. The minimization process is a procedure which complexity depends on the algorithm implementation, on the image size, on the number of control points used to describe and move the active contour, on the approximation method used to describe the complete discretized curve for each iteration, and, finally, on the number of iterations needed in order to reach the complete and stable convergence to the desired boundaries.

As shown in the previous section, the iterative process for the snake minimization uses a $N^t \times N^t$ matrix, with N^t number of control points at the *t*th temporal iteration. Then, the computational complexity is related to the matrix size and, therefore, to the implementation capacity in minimizing the total number of control points N^t or optimizing the location with respect to the characteristics and positions of the desired boundaries.

The simplest methods for controlling the trade off between speed and accuracy is based on a regularization process performed, in each iteration step, on the number of control points within the parametric curve. The standard snake implementations (e.g. [4,6]) do not allow any type of control for this process except for a regularization of the control point positions, in term of an almost equi-spacing performed with a minimum and maximum distance constraint between consecutive control points ($d_{\min}$ and $d_{\max}$ represent respectively the minimum and maximum acceptable distance between two consecutive control points in each iteration step).
Obviously, any simplification of the original model, in term of minimization of the number of control points, brings to a variable loss of accuracy in the final contour.
With this approach, all the previous implementations, based on the standard approach, provide a simple method for controlling the process speed and the final contour accuracy. Absolute contour smoothness and precision (small $d_{\min}$ and $d_{\max}$) or process speed (big $d_{\min}$ and $d_{\max}$) can be chosen depending on the applications.
Such simple method of regularization does not take into account the topology of the active contour that depends on the external field intensity and direction. In other terms, the classical regularization phase does not take into account the boundaries of the objects within the image plane. Although this procedure is not directly connected to the minimization of the energy functional that brings the active contour parametric curve at a lower energy level, with respect to the previous iteration step, the regularization procedure is present in each numerical active contour implementation, because it is strongly necessary for the regridding and, therefore, the stability of the algorithm [16]. It is usually performed every *j* iterations and it represents an operation independent from both the external energy building and the iterative evolution process of the snake. Within the regularization step it can be possible to define a new time step *j*, corresponding to the number of iterations between two regularization phases of the control points. The purpose of this procedure is to obtain a compromise between a number of control points sufficient to an effective and efficient description of the desired contour and the minimum number of points necessary for allowing a faster snake convergence. Then, a smaller time step *j*, brings a better and more frequent control points optimization procedure but it involves a greater computational load. Nevertheless, when the regularization is performed using the $d_{\min}$ and $d_{\max}$ approach, whatever the time step j is chosen, whatever the number of control points N^t are chosen, where the curvature of the desired edges are big, the contour approximation is done in a less accurate manner, resulting in smooth boundaries because of the reduced control point number. Even with an accurate interpolation (for instance *n*th-order B-Splines) between control points, object vertexes cannot be preserved.
In the next section it will be shown how, under specific conditions, it can be possible to ensure that the loss in the active contour accuracy can be quite small and compensated by a substantial augment of speed in the convergence process.

3.1 Active Contour Energy Model

The whole initialization and evolution process of almost any active contour implementation is schematically summarized in Fig. 1.
In order to find an effective solution to the control point's regularization procedure, consider the problem under a more general point of view. Consider the control point number regularization like a new energy minimization process. A configuration energy term E_{comf} should be introduced for this purpose. This term should be related to the control points configuration, position and number and, hence, to the relative computational load needed in order to move the parametric curve towards the desired contour. In order to realize a definition for this energy term, the following notation is first introduced to define the set of $N(t)$ control points at the *t*th iteration, with $i = 1,2,...,N(t)$:

$$P_{N(t)}(t) \equiv \{x_i(t), y_i(t)\}. \tag{26}$$

The E_{conf} energy minimization process is a regularization step and it is performed every *j* temporal iterations. The process is computed through the reduction of the number $N(t)$ of control points, until the new value $M(t)$ is reached. This operation will be guided and controlled in order to obtain an optimized configuration with predefined constraints like a minimum distance from the original (non optimized) configuration.
Consider

$$d\left(P_{N(t)}(t), P_{M(t)}(t)\right) \tag{27}$$

as the difference between two snake configurations ($P_{N(t)}(t)$ and $P_{M(t)}(t)$) at the *t*th temporal iteration step. If $P_{N(t)}(t)$ represents the active contour configuration, the trade-off between convergence speed and active contour accuracy can be achieved finding an appropriate $P_{M(t)}(t)$, with $M(t) < N(t)$, that minimizes the following term

$$E_{conf} = \left| d\left(P_{N(t)}(t), P_{M(t)}(t)\right) - C \right| \tag{28}$$

where *C* is a constant value. The constant *C* controls the maximum admissible distance between the starting active contour representation $P_{N(t)}(t)$ at the given instant time *t* and the optimized one $P_{M(t)}(t)$. The new energy term is independent from both the external field and the internal energy used in any snake implementation and it gives a measure of the actual state and position of the active contour in the *t*th iteration step. Considering Eq. (28) as a further energy element in the classical active contour description, Eq. (1) becomes

$$E_{snake} = E_{\text{int}} + E_{image} + E_{con} + E_{conf} \tag{29}$$

For each temporal iteration, the minimization of E_{conf} gives the constraints for an optimized configuration $P_{M(t)}(t)$. The minimization of E_{comf}, i.e., the minimization of the number of control points, have to be guided and limited in order to obtain a configuration that is as close as possible to the reference configuration at each iteration *j*. In fact the minimization process could bring to a continuous elimination of points that will create a less accurate final contour.

3.2 Model Implementation

Some problems should be solved in order to achieve the minimization of Eq. (28). A concrete implementation should define an appropriate threshold value *C* for an acceptable maximum distance, between reference and optimized configurations, which can easily take into account different needs and tolerance thresholds of several application fields. Furthermore, the method introduced for optimizing control points position brings further computational load for the minimization of Eq. (28). The following saving in terms of total control points number and final computational time should compensates this further computational load. Moreover, when classical distance metrics for polygonal curves, like Hausdorff or Fréchet, are used in order to evaluate Eq. (27), they do not give a useful measure of the differences between two discrete curves, in particular when the positions and the number of control points can change during the optimization process.

Consider a simplified but common case. Assume that control points can be dynamically regridded, using a simple mechanism like the minimum and maximum distance-based optimization or a more complex one. Control points can be added or deleted but they can not change their position within the same time step. In this case the addition of a new control point does not change the shape of the parametric curve, while the deletion of a control point *k* brings an area difference between the reference curve and the optimized one equal to the triangle characterized by the points *k-1*, *k*, *k+1*.
In this simplified case, the minimization of Eq. (28) corresponds to a point elimination process. Hence, the global threshold *C*, representing the maximum admissible distance between the two curves, becomes a total area difference between them.
The next step is the choice of an efficient and effective method for the point deletion needed by the configuration energy minimization. The proposed method is based on the connection between the active contour shape and the local curvature of the desired object boundaries. When the initial contour starts to approach the desired boundaries, the snake shape starts to look like the object contour. Then, the point elimination process done in each iteration step can be done obtaining the best possible description of high curvature zones of the final object boundary. The idea is that the algorithm should delete only the points which local curvature is smaller then a given local threshold (at each *j*th iteration).
Let us define

$$C^L(k) = \frac{1}{\delta^L(k) \cdot A^L(k)} \tag{30}$$

as an index representing the importance of a control point for the active contour representation, where $\delta^L(k)$ represents the local curvature computed using Eq. (25) and $A^L(k)$ is the area delimited by the same three points.
The energy minimization algorithm evaluates and, eventually, deletes iteratively the control points $V_k = V(x_k, y_k)$ in a given order, if

$$C^L(k) > C^{TH} \tag{31}$$

until

$$\sum_{i=1}^{N_{TH}} A_i^L < C \tag{32}$$

where C^{TH} is a local threshold chosen depending on the precision required by the application field and C is the global threshold representing the maximum admissible distance between the two active contours in the *t*th iteration and N_{TH} is the total number of control point deletions for which the distance between the active contours remains under the given global threshold C.
Summarizing, point deletion have to be performed, keeping into account the local value $C^L(k)$ that depends on the local curvature and tries to maintain the two curves *locally close*. Furthermore the point deletion have to be done considering the global threshold *C* that is the maximum admissible distance between the two active contours representations in the same iteration step.

4. Experimental results

Tests have been carried out comparing the proposed FACE implementation and the GVF and EVF active contour algorithms. In this paper twelve different images are chosen from the large image test set used containing binary, medical, video conference and synthetic pictures. Computational times and the final active contours are obtained using the FACE method with the classical control points regularization through the $d_{\min}$ and $d_{\max}$ approach. Then the proposed minimization algorithm, based on curvature estimation, have been performed every *t=5* iterations.

In order to assess the performance of the proposed method, GVF, EVF and FACE algorithms were implemented in C++ language on an AMD Athlon 2GHz with 128MBytes of RAM. The same visualization library and the same structure and algorithms have been used for the GVF, EVF and FACE implementations.

Informations on how to obtain the code of the proposed algorithm and the test images are available at [46].

The three algorithms have been applied to each test image starting from the same initialization active contour. The processing time and the number of iterations for the field construction and the snake convergence were obtained setting the parameters, characterizing the three methods, as follows:

$a = 0.05$, $\beta = 0$, $\gamma = 1$, $\kappa = 0.6$, $\mu = 0.2$, $d_{\min} = 0.5$, $d_{\max} = 2$.

The edge map used by GVF is normalized. The same external energy is used in both FACE and EVF implementations except for the use of a thresholded Sobel operator instead of a simple Sobel filter. The binarization performed with the thresholded Sobel filter becomes a critical step in both single-object or multi-objects cases. The chosen threshold is 10 for all the images except for some test pictures where a different value is used. For the two medical images, because of the intrinsic noisy nature of this type of images, the Sobel threshold is set to 50, while for the noisy Pliers test image the binarization threshold is set to 17.

Several visual results are presented in this paper. In the first set of images (Figs. 2-3) two examples of final contour are shown. The test pictures are respectively, one frame of *Claire* video sequence and the *Room* test image, originally used by Xu and Prince [6]. Only control points are drawn and superimposed to the two images. It can be observed how the FACE implementation allows a better and more efficient distribution of control points, resulting in a smaller number of points, located where final contour presents a maximum in his curvature, for example in *Room* image corners. For the *Room* test image, final contours are similar in both GVF and FACE cases but GVF snake takes about 1.18 seconds while FACE snake needs about 0.47 seconds. The shown final contour point distribution is reached after 100 iterations for GVF and 40 for FACE.

Figs. 4-13 show active contour initialization, and, in this order, the best GVF, EVF and FACE final results. The control points initialization is always located onto the image borders except when multiple objects (Figs. 6,7,13) are present. In this case, the only way to locate the desired object is to initialize the snake close to its boundaries. Images are all in 176 x 144 pixel monochrome having 8 bits per pixel, except for the binary 176 x 144 *U-shape* and the 352 x 288, 8 bits per pixel *Drop* images.

The results for processing time and the number of iterations are reported in Table 1 and 2. Comparable speeds are reached by EVF and FACE external field but the convergence algorithm, optimized with the FACE process, is more than twice faster (Snake column in Table 1). In order to evaluate the reduction of the control points and the resulting increased speed, the FACE method was applied to the image set using the control point optimization algorithm (CPO) every five iteration. Than, the FACE method was applied using the same initialization and the same field and snake iteration number but with the CPO disabled. The final number of control points for the FACE algorithm with and without the proposed CPO is shown in Table 3. CPS column represents the control point saving percentage as

$$CPS = \frac{CPN_{No-CPO} - CPN_{CPO}}{CPN_{No-CPO}} \cdot 100 \tag{33}$$

where CPN_{CPO} and CPN_{No-CPO} are the final control point number with and without the optimization algorithm. Results show CPS values between a minimum of 38% to a maximum of 60%. TS column is the computational time saving percentage:

$$TS = \frac{SCT_{No-CPO} - SCT_{CPO}}{SCT_{No-CPO}} \cdot 100 \tag{34}$$

where SCT_{CPO} and SCT_{No-CPO} are the final snake computational times with and without optimization algorithm. TS values are between 33% and 60%.

Fig. 15 shows accuracy results for the proposed optimization algorithm. A manual segmentation has been performed. Then, segmentation results for the Face algorithm with and without CPO have been obtained. The final results have been computed for the two version of the FACE algorithm as percentage of global error with respect to manually-segmented images:

$$Error_{\%} = \frac{\sum_{i,j} \left| R(i,j) - P(i,j) \right|}{RPN} \cdot 100 \tag{35}$$

where $R(i,j)$ are the binary values for pixels belonging (one) or not (zero) to the manually-segmented reference mask and $P(i,j)$ are the binary values for pixels belonging (one) or not (zero) to the automatically segmented mask. RPN is the number of pixels belonging to the manually-segmented reference mask. Results of Eq. (35) computed for FACE with and without CPO (Fig. 15) show that the control point reduction and the resulting increased speed can be obtained with comparable final visual results.

In Table 4 a comparison between the properties of the active contour methods discussed in this paper is shown.

It should be noted that FACE implementation brings a further computational load, $O(m \cdot n)$, where m is the number of FACE optimizations. For this test session m is equal to the active contour iteration number divided by five, and n is the total number of control points, evaluated by FACE in each optimization phase, until the global threshold is reached.

Hence, the curvature-based point elimination algorithm increases the computation load. Nevertheless, the FACE optimization process decreases the control point number (Table 3), without losing accuracy especially when objects with high curvature edges are considered (Fig. 15). The global process of snake convergence results faster with respect to the similar GVF and EVF methods.

5. Conclusions

In this paper, an alternative approach to the regularization of the control points of an active contour method was proposed. The procedure achieves an increased computational speed when compared with other similar approaches for image segmentation. It is based on an accurate curvature analysis of the parametric model. Control points are added or deleted in order to achieve a better description of the desired contour. High curvature segments of the active contour tend to keep and preserve control points to better describe vertexes and line terminations, while a smaller number of control points is used to describe smoother edges. Even though the curvature evaluation procedure increases the computational load, the proposed method achieves a reduction of total processing time, when compared with previous similar algorithms, as it can be observed from the analysis of the experiments shown in the previous section. GVF approach is very efficient and effective for the object segmentation due to its property of convergence towards boundary concavities. So the model proposed in this paper is based on a GVF-like field, locally, in order to allow concavity convergence. Then the novel approach based on local curvature evaluation and global difference

minimization is used in addition with the standard regularization procedure based on minimum and maximum acceptable control point distance. Furthermore, a binary thresholded Sobel filter is used in the preprocessing phase. Although it brings a greater computational load with respect to classical gradient operator, it allows a superior construction of external vector field. This noise-resistant filtering brings to a better convergence towards desired final boundaries. From the analysis of the experiments it can be noted that the proposed model reduces the global computational times for almost all test images when compared with the other two similar active contour models, GVF and EVF. The improved speed performance is achieved without any appreciable loss in final contour details.

References

[1] D. Terzopoulos, and K. Fleischer, Deformable models, The Visual Computer, 4 (6) (1988) 306–331.

[2] T. McInerney and D. Terzopoulos, Deformable Models in Medical Image Analysis: A Survey, Medical Image Analysis, 1 (2) (1996) 91–108.

[3] A. Blake and M. Isard, Active Contours, Springer, London, 1998.

[4] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active Contour Models, International Journal of Computer Vision, (1) (1987) 321-331.

[5] L. D. Cohen, I. Cohen, Finite-Element Methods for Active Contour Models and Balloons for 2-D and 3-D Images, IEEE Trans. Pattern Analysis and Machine Intelligence, 15 (11) (1993) 1131-1147.

[6] C. Xu e J. L. Prince, Snakes, Shapes, and Gradient Vector Flow, IEEE Transactions on Image Processing, 7 (3) (March 1998).

[7] A.A. Amini, T.E. Weymouth, R.C. Jain, Using Dynamic Programming for Solving Variational Problems in Vision, IEEE Transaction On PAMI, 12 (9) (September 1990).

[8] D.J. Williams, and M. Shah, A Fast Algorithm for Active Contours and Curvature Estimation, CVGIP (Image Understanding), 55 (1) (1992) 14-26.

[9] M.O. Berger, Snake growing, Computer Vision - ECCV 90. Springer-Verlag, 1990, pp. 570-572.

[10] D. Terzopoulos, On matching deformable models to images, In Topical meeting on machine vision, Technical Digest Series, Optical Society of America 12 (1987) 160-163.

[11] L. D. Cohen, On Active Contour Models and Balloons, Computer Vision Graphics and Image Processing: Image Understanding 53 (2) (March 1991) 211-218.

[12] D. Geiger, A. Gupta, L.A. Costa and J. Vlontzos, Dynamical programming for detecting, tracking and matching deformable contours, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 17, No. 3, pp. 294-302, 1995.

[13] R. Ronfard, Region-based strategies for active contour models, International Journal of Computer Vision, 13 (2) (October 1994) 229-251.

[14] V. Caselles, F. Cattè, T. Coll, F. Dibos, A geometric model for active contours in image processing, Numerische Mathematik, n.66, 1993, pp. 1-31.

[15] V.Caselles, R. Kimmel, and G. Sapiro, Geodesic active contours, In Fifth International Conference on Computer Vision, 1995.

[16] S. Osher, J.A. Sethian, Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations, Journal of Computational Physics 79 (1988) pp.12-49.

[17] R. Malladi, J.A. Sethian, B.C. Vermuri, Shape Modeling with Front Propagation: A Level Set Approach, IEEE Trans. On Pattern Analysis and Machine intelligence, 17 (2) (February 1995).

[18] C. Perra, F. Massidda, D.D. Giusto, Active contour for automatic segmentation of video sequences, Workshop on Image Analysis for Multimedia Interactive Services, London, UK, April 9-11, 2003.

[19] D.D. Giusto, F. Massidda, C. Perra, A fast algorithm for video segmentation and object tracking, International Conference on Digital Signal Processing (DSP 2002), Santorini, Greece, July 1-3, 2002.

[20] F. Mohanna, and F. Mokhtarian, Improved Curvature Estimation for Accurate Localization of Active Contours, Proc. International Conference on Image Processing, vol. II, Thessaloniki, Greece, 2001, pp. 781-784.

[21] D. Perrin and C. Smith, Rethinking classical internal forces for active contour models, In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2001.

[22] T. McInerney and D. Terzopoulos, T-snakes: Topology adaptive snakes. Medical Image Analysis, 4(2):73–91, 2000.
[23] S.C. Zhu and A. Yuille, Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, no. 9, pp. 884-900, Sept. 1996.
[24] S.R. Gunn and M.S. Nixon, Robust snake implementation: a dual active contour, IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 19, No. 1, pp. 63-67, 1997.
[25] N. Peterfreund, The Velocity Snake: Deformable Contour for Tracking in Spatio-Velocity Space, Computer Vision and Image Understanding, Vol. 73, No.3, March 1999.
[26] Y. Zhong, A. K. Jain, M. Dubuisson-Jolly, Object Tracking Using Deformable Templates, IEEE Trans. Pattern Anal. Machine Intell., vol.22, pp. 544-549, 2000.
[27] S. Jehan-Besson, M. Barlaud, G. Aubert, A 3-Step Algorithm using Region-Based Active Contours for Video Objects Detection, WIAMIS 2001 - Workshop on Image Analysis for Multimedia Interactive Services, Tampere, Finland, 16-17 May 2001.
[28] Z. Yu, C. Bajaj, Image Segmentation Using Gradient Vector Diffusion and Region Merging, Proceedings of the 16th International Conference on Pattern Recognition, vol. 2, pg. 941-944, (Published by IEEE CS-Press), Quebec, Canada, August 11-15, 2002.
[29] N. Ray, S. T. Acton, T. Altes, E.E. de Lange, J.R. Brookeman, Merging Parametric Active Contours Within Homogeneous Image Regions for MRI-Based Lung Segmentation, IEEE Transaction On Medical Imaging, Vol. 22, No. 2, February 2003, 189-199.
[30] H.T. Nguyen, M. Worring, and R. van den Boomgaard, Watersnakes: Energy-Driven Watershed Segmentation, IEEE Trans.On Pattern Analysis And Machine Intelligence, Vol. 25, No. 3, March 2003.
[31] R. Zaritsky, N. Peterfreund, N. Shimkin, Velocity-Guided Tracking of Deformable Contours in Three Dimensional Space, International Journal of Computer Vision 51(3), 219–238, 2003.
[32] N. Paragios and R. Deriche, Geodesic active contours and level sets for the detection and tracking of moving objects, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(3):266–280, March 2000.
[33] G. Sapiro, Geometric Partial Differential Equations and Image Analysis, Cambridge University Press, New York, 2001.G. Sapiro, Geometric Partial Differential Equations and Image Analysis, Cambridge University Press, New York, 2001.
[34] J. Shah, A common framework for curve evolution, segmentation and anisotropic diffusion, In Proceedings IEEE CVPR'96, pages 136–142, June 1996.
[35] T.F. Chan, and L.A. Vese, Active Contours without Edges, IEEE Trans. On Image Processing, Vol. 10, No. 2, pp266-277 February 2001.
[36] M. Bertalmio, G. Sapiro, and M.G. Randall, Morphing active contours, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(7):733–737, 2000.
[37] D.L. Chopp, Computing minimal surfaces via level set curvature flow, Journal of Computational Physics, v.106 n.1, p.77-91, May 1993.
[38] DdS. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi, Gradient flows and geometric active contour models. In Proceedings ICCV'95, Boston, Massachusetts, June 1995.
[39] A.-R. Mansouri, J. Konrad, Motion Segmentation with Level Sets, ICIP (2), Kobe, Japan, 1999: 126-130.
[40] N. Ray and S.T. Acton, Image segmentation by curve evolution with clustering, Proc. Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, California, October 29 - November 1, 2000.
[41] J. A. Sethian, Level Set Methods and Fast Marching Methods, Cambridge: Cambridge University Press, 1999.
[42] D. Adalsteinsson , J.A. Sethian, A fast level set method for propagating interfaces, Journal of Computational Physics, v.118 n.2, p.269-277, May 1995.

[43] G. Kühne, J. Weickert, M. Beier, and W. Effelsberg, Fast Implicit Active Contour Models. In: Pattern Recognition, DAGM-Symposium 2002, 133-140.
[44] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky, Fast geodesic active contours, IEEE Transactions on Pattern Analysis and Machine Intelligence, 10(10):1467–1475, October 2001.
[45] R. Goldenberg, R. Kimmel, E. Rivlin, M. Rudzsky, Cortex Segmentation-A Fast Variational Geometric Approach , IEEE Workshop on Variational and Level Set Methods (VLSM'01), July 13 - 13, 2001 Vancouver, Canada.
[46] D.D. Giusto, Francesco Massidda, Cristian Perra, FACE: Fast Active-Contour Curvature-based Evolution, available at http://www.diee.unica.it/mclab/face, DIEE - Department of Electrical and Electronic Engineering, University of Cagliari.

Figures and tables.

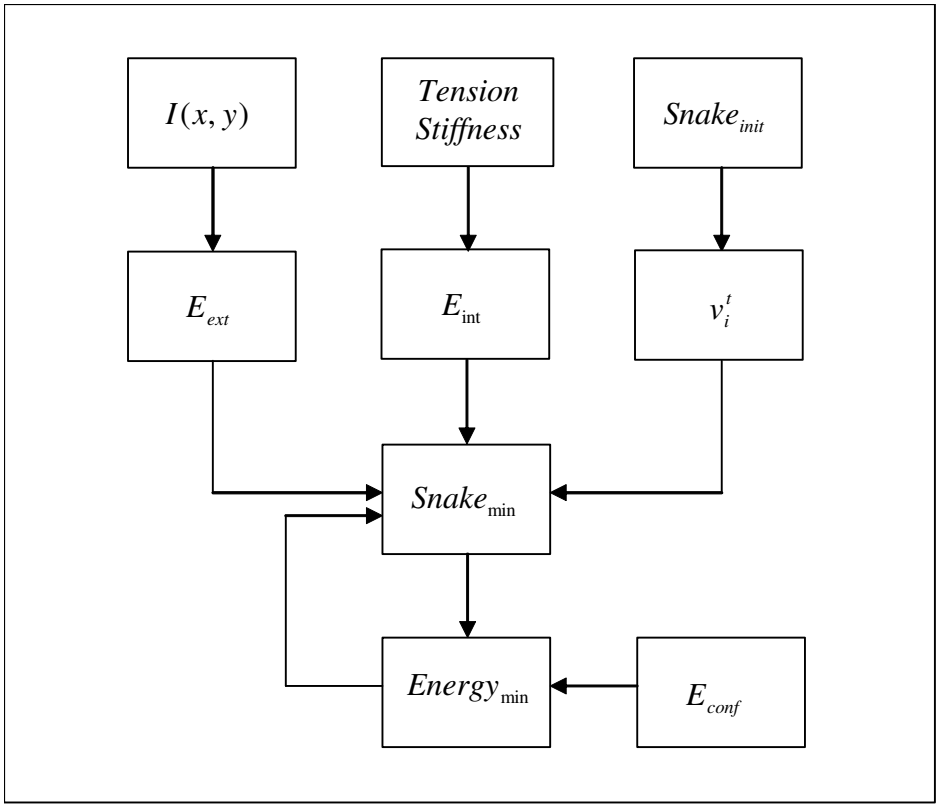
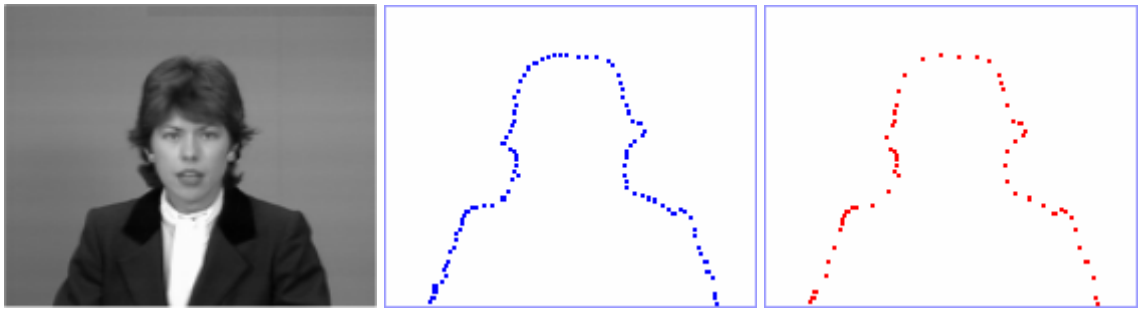
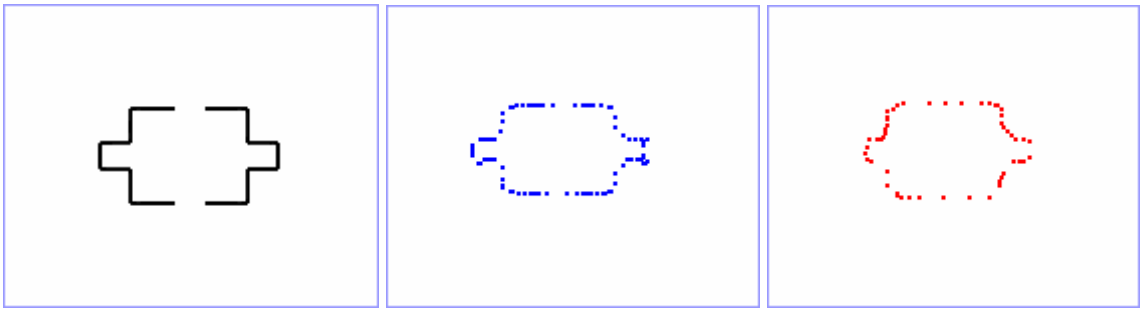


Fig. 1. The block diagram of the proposed model. Image data *I(x,y)* is used to create the external field. User sets the initial point distribution v_i^t and the iterative snake minimization procedure starts. At each step, a regularization process is done in order to obtain the configuration energy minimization.



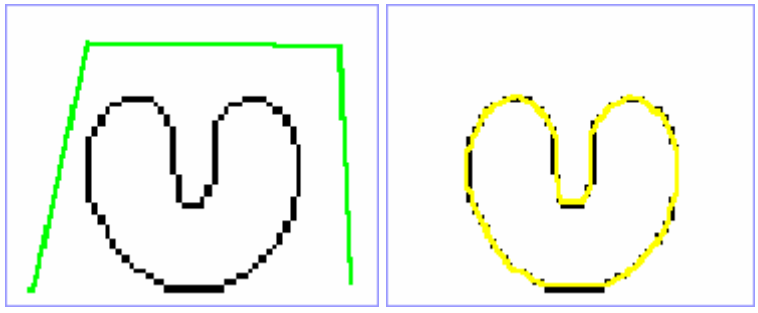
(a) (b) (c)

Fig. 2. Claire: Original image (a) and point distribution for a GVF-like implementation (b) and FACE implementation (c). The same final result is reached with a lower number of control points, mainly distributed along curvature maxima.

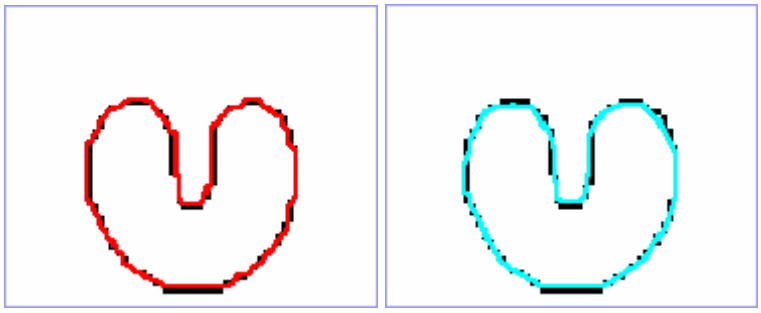


(a) (b) (c)

Fig. 3. Room: Original image (a) and point distribution for GVF implementation (b) and FACE implementation (c).

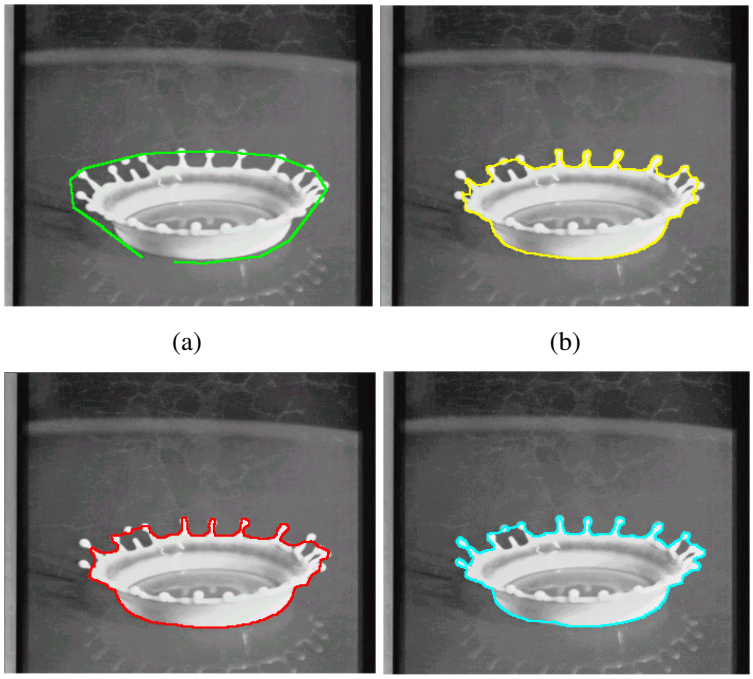


(a) (b)



(c) (d)

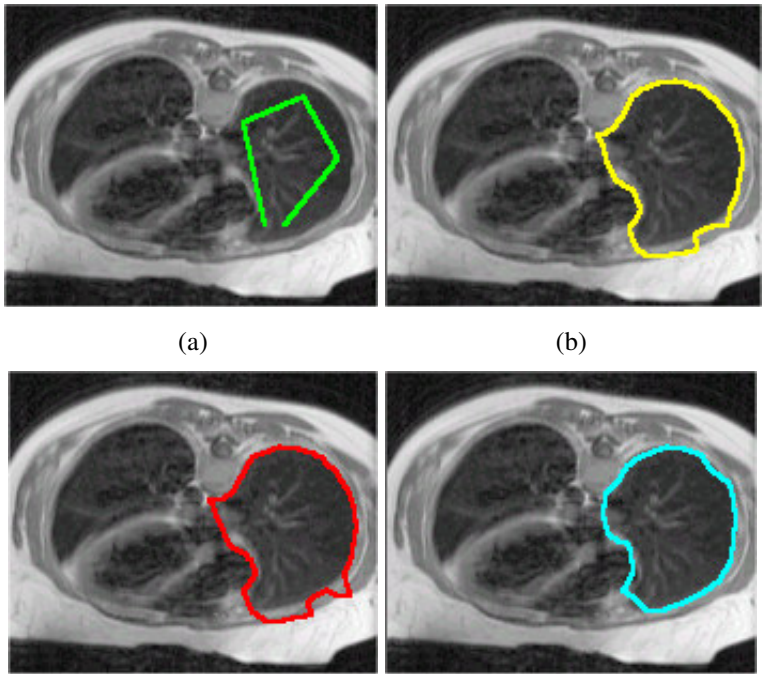
Fig. 4. U-Shape: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

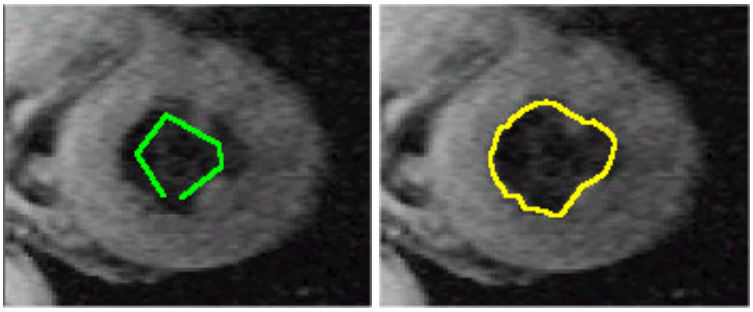
Fig. 5. Drop: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



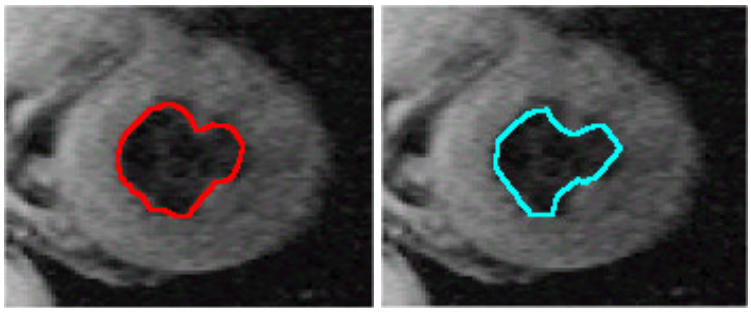
(a) (b)

(c) (d)

Fig. 6. Chest: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).

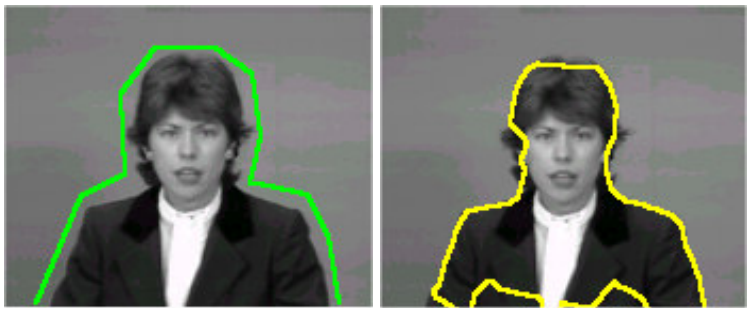


(a) (b)



(c) (d)

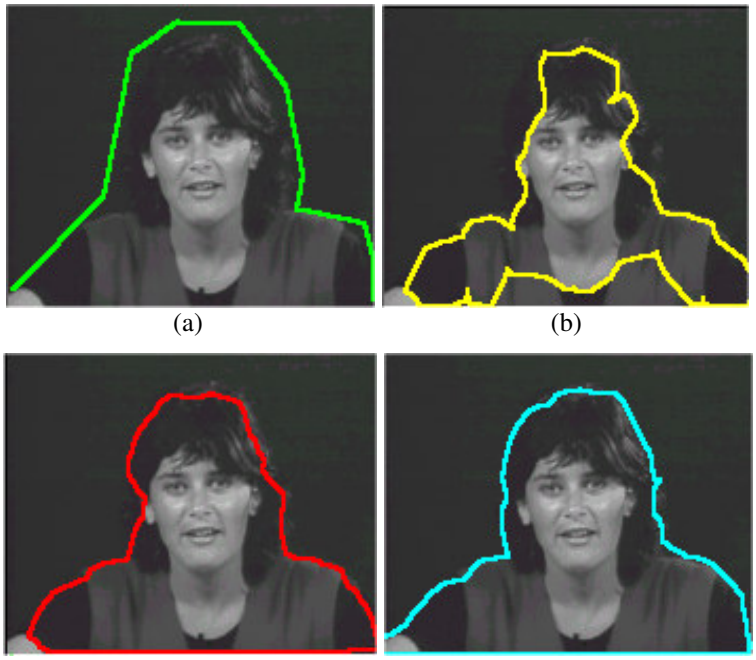
Fig. 7. Heart: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

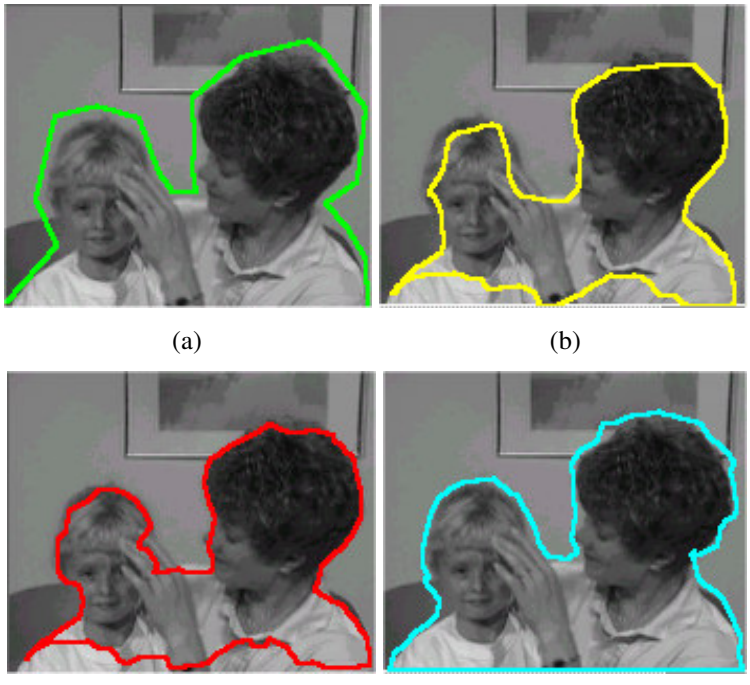
Fig. 8.Claire: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

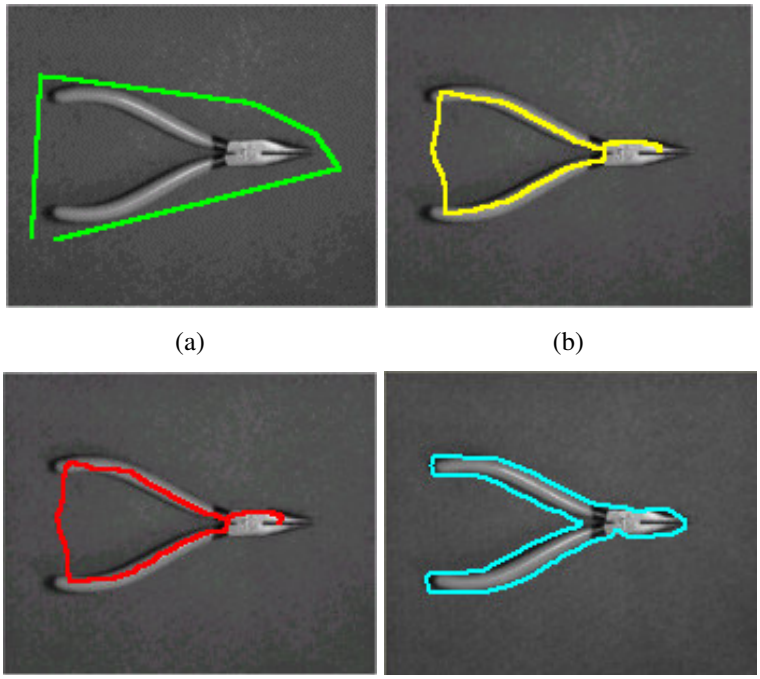
Fig 9. Miss America: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

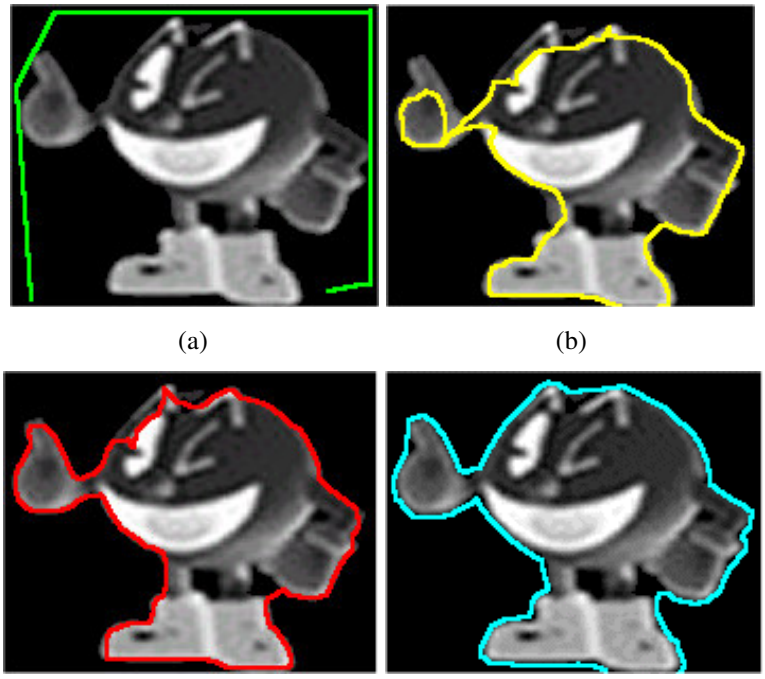
Fig. 10. Mother and Daughter (Mtd): snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

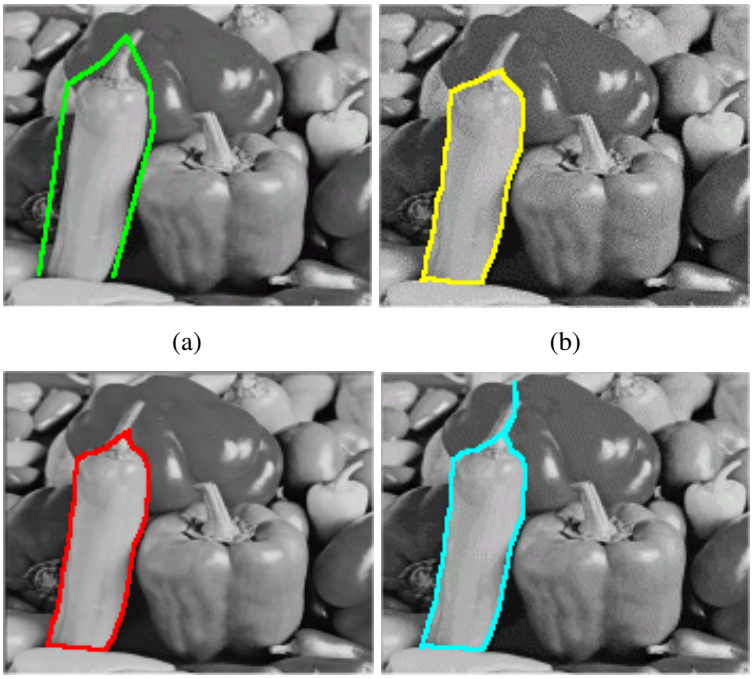
Fig. 11. Pliers: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

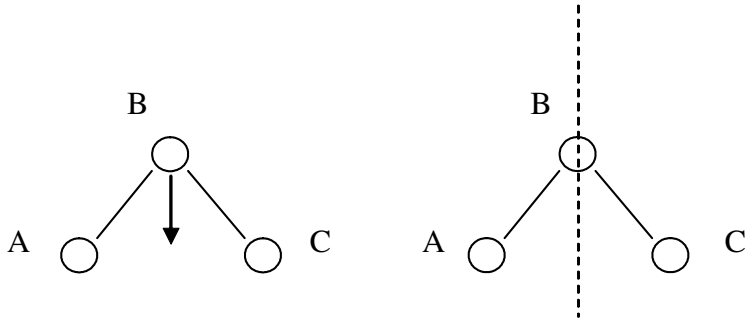
Fig. 12. Mms: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

(c) (d)

Fig. 13. Peppers: snake initialization (a) and results for GVF implementation (b), EVF implementation (c) and FACE implementation (d).



(a) (b)

Fig. 14. Internal forces implementations: (a) the classical tension force tries to push B toward AC segment, making a straight line, (b) the tension force has a minimum on the perpendicular of AC segment and control point B is now in equilibrium everywhere, within the dashed line.

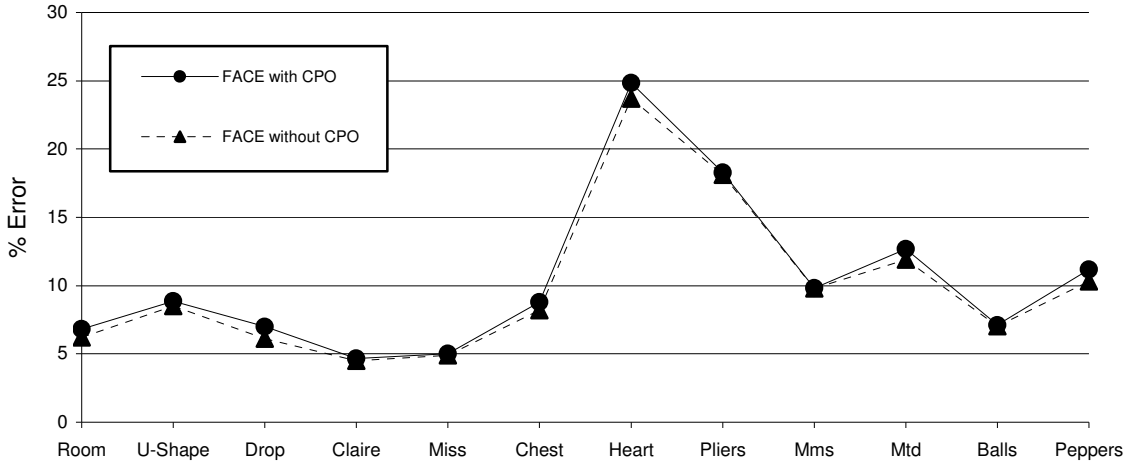


Fig. 15. Accuracy measures for the Face algorithm with and without Control Point Optimization (CPO). Accuracy rates are computed comparing the obtained final segmentation with a manual segmentation. Results are shown as percentage of global error with respect to manually segmented images. Some visual results of Face algorithm with CPO are shown in Figs. 4-13 (d).

Table 1
Performance comparison between GVF, EVF and FACE algorithms. *Field* column reports the execution time for computing the external field while *Snake* column reports the execution time for the snake evolution.

Execution Time (seconds)									
	GVF			EVF			FACE		
	Field	Snake	Tot	Field	Snake	Tot	Field	Snake	Tot
Room	0.52	0.66	1.18	0.15	0.61	0.76	0.17	0.30	0.47
U-Shape	1.46	2.60	4.06	1.44	2.61	4.05	1.44	1.70	3.14
Drop	1.64	3.02	4.66	1.63	5.00	6.63	1.64	0.90	2.54
Claire	0.30	0.41	0.71	0.18	0.40	0.58	0.18	0.18	0.36
Miss_am	0.31	0.75	1.06	0.20	0.70	0.90	0.21	0.30	0.51
Chest	0.55	0.29	0.84	0.31	0.32	0.63	0.32	0.10	0.42
Heart	0.50	0.20	0.70	0.41	0.27	0.68	0.16	0.20	0.36
Pliers	0.31	0.73	1.05	0.32	0.70	1.01	0.15	0.66	0.81
Mms	0.51	4.00	4.51	0.16	2.55	2.71	0.17	0.74	0.91
Mtd	0.50	0.66	1.16	0.15	0.82	0.97	0.16	0.31	0.47
Balls	0.30	1.10	1.40	0.18	0.60	0.78	0.18	0.32	0.50
Peppers	0.30	0.11	0.41	0.21	0.12	0.33	0.22	0.04	0.26

Table 2
Comparison of iteration number for GVF, EVF and FACE algorithms. *Field* column reports the number of iterations for computing the external field while *Snake* column reports the number of iterations for the snake evolution. For particular images (deep boundaries concavities), EVF needs a greater number of snake iterations in order to obtain comparable results. This lower ability of interpolated EVF field to drive the active contour into boundary concavities is partially solved using a thresholded Sobel filter.

Number of iterations						
	GVF		EVF		FACE	
	Field	Snake	Field	Snake	Field	Snake
Room	100	100	20	90	20	40
U-Shape	350	350	350	350	350	350
Drop	100	120	100	200	100	70
Claire	50	30	20	50	20	25
Miss_am	50	50	20	40	20	30
Chest	100	50	50	50	50	20
Heart	100	40	80	50	20	40
Pliers	50	120	50	120	20	150
Mms	100	200	20	160	20	80
Mtd	100	30	20	40	20	30
Balls	50	90	20	40	20	30
Peppers	50	15	20	15	20	10

Table 3
Control point number for FACE algorithm with and without Control Point Optimization (CPO). The starting control points and iteration number are the same used for Table 1 and 2 final results. The common parameters are: α=0.05, β=0, γ=1, κ=0.6, μ=0.2 d_{min}=0.5, d_{max}=2. CPO algorithm have been performed every t=5 iterations. Control Point Saving Percentage (CPS) and Time Saving Percentage (TS) are also shown.

	Starting Control Point Number	Iteration Number		Final Control Point Number		CPS	TS
		Field.	Snake	FACE without CPO	FACE with CPO		
Room	4	20	40	79	49	38%	36%
U-Shape	4	350	350	179	79	56%	56%
Drop	16	100	70	359	166	54%	59%
Claire	6	20	25	109	67	39%	45%
Miss_am	6	20	30	155	70	55%	33%
Chest	5	50	20	99	44	56%	43%
Heart	6	20	40	58	33	43%	36%
Pliers	6	20	150	181	73	60%	46%
Mms	6	20	80	239	114	52%	41%
Mtd	16	20	30	224	92	59%	54%
Balls	9	20	30	134	63	53%	45%
Peppers	8	20	10	126	65	48%	60%

Table 4
Comparison of vector field properties (Y=Yes, N=No)

	Active contour vector field			
Properties	Traditional	GVF	Distance Map	FACE
Large capture range	N	Y	Y	Y
Convergence to boundary concavities	N	Y	N	Y
Fast Field computation	Y	N	N	Y