Real-time Interactive Path Extraction with On-The-Fly Adaptation of the External Forces

Olivier Gérard[1], Thomas Deschamps[1,2,3], Myriam Greff[1], and Laurent D. Cohen[3]

[1] Medical Imaging Systems Group, Philips Research France,
51 rue Carnot, 92156 Suresnes, France. Olivier.Gerard@philips.com
[2] Now with Lawrence Berkeley National Laboratory and
University of California, Berkeley, USA. tdeschamps@lbl.gov
[3] Ceremade, Université Paris 9-Dauphine,
75775 Paris Cedex 16, France. cohen@ceremade.dauphine.fr

Abstract. The aim of this work is to propose an adaptation of optimal path based interactive tools for image segmentation (related to *Live-Wire* [12] and *Intelligent Scissors* [18] approaches). We efficiently use both discrete [10] and continuous [6] path search approaches. The segmentation relies on the notion of energy function and we introduce the possibility of complete *on-the-fly* adaptation of each individual energy term, as well as of their relative weights. Non-specialist users have then a full control of the drawing process which automatically selects the most relevant set of features to steer the path extraction. Tests have been performed on a large variety of medical images.

1 Introduction

Approaches in image segmentation are numerous, ranging from fully automatic methods to fully manual methods. The first ones totally avoid user's interaction but still are an unsolved problem: even if they are well adapted to specific cases their success can not be guaranteed in more general cases. The second ones are time-consuming, hardly reproducible and inaccurate. To overcome these problems, interactive (or semi-automatic) methods are used. They combine knowledge of the user and computer capabilities to provide supervised segmentation, ranging from manual pixel level drawing to minimal user intervention.

The aim of this work is to develop a method to offer the possibility even for a non-expert to draw quickly the boundary of an object of interest. He could restrict its intervention to the setting of a start point in an image. Then a contour has to be automatically found and drawn in real-time between this start point and the current cursor position. This contour has to respond to a certain constraints, such as internal and external forces and action of the user. The developed method has to let the user validate or not the result. Using this validation, the tool has to be able to learn on-line to better estimate the different parameters of the model.

In contour oriented segmentation, one approach is to define a boundary as the minimum of an energy function that comprises many components such as internal and external forces. In the literature, there exist many techniques to perform this minimization.

Classical active contours (also called Snakes or Deformable Models), introduced by *Kass, Witkin and Terzopoulos* [16], have received a lot of attention during the past decade. But this technique presents three main problems. First, deformable models are very sensitive to the initialization step and often get trapped in a local minimum [3]. Second, user's control cannot be applied during the extraction but only during the initialization and the validation stages of the segmentation process. Third, the different parameters of the model are not meaningful enough in a user's viewpoint (especially for clinicians).

The application of the minimal path theory to image segmentation is a more recent technique. With this approach the image is defined as an oriented graph characterized by its cost function. The boundary segmentation problem becomes an optimal path search problem between two nodes in the graph. This approach overcomes the problem of local minima by using either dynamic programming (*Dijkstra* [10]), or a front propagation equation (*Cohen and Kimmel* [6]), mapping the non-convex cost function into a function with only one minimum. *Falcao and Udupa* with their *Live-Wire* [11, 12] and *Mortensen and Barrett* with their *Intelligent Scissors* [18–21] introduced interactivity into the optimal path approach. Their method is based on Dijkstra's graph search algorithm and gives to the user a large control over the segmentation process. The idea is the following: a start point is selected by the user on the boundary to be extracted, and an optimal path is computed and drawn in real time between this start point and the current cursor position (see Fig. 1). Thus, user's control is applied also

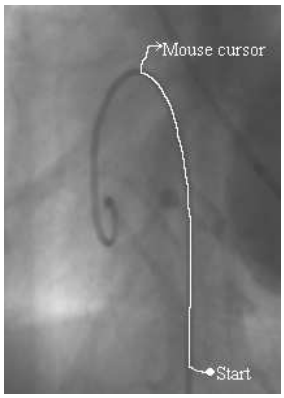


Fig. 1. Principle of interactive contour extraction with optimal path method: the optimal trajectory is extracted in real time between a user defined starting point and the mouse cursor.

during the extraction. *Mortensen and Barrett* [1, 20] also introduced functionalities called *Path-Cooling* and *On-The-Fly* training, which respectively lead to the possibility of drawing a closed contour, and to partial adaptation of the graph cost function. This technique seems to be the most adequate according to our target. The application consists therefore first in the implementation of a method based on minimal path search inspired from *Live-Wire* and *Intelligent Scissors* tools, as presented in section 2. Section 3 explains then several improvements brought to the interactive path extraction techniques, through the use of *Eikonal equation* for extracting the minimal path. Section 4 details the implementation of our *On-The-Fly* training method. Section 5 contains a discussion of the proposed method and future works.

2 *Live-Wire* and *Intelligent Scissors* methods

2.1 Cost function

The optimal-path search is guaranteed to find the solution minimizing the energy function between two points. But if this function does not suit enough the object to extract, the contour obtained by this method will not be suitable. That is why we are proposing to automatically tune the parameters of the cost function during the extraction process. Nevertheless, the cost function relies on salient features of the image. A feature is supposed to describe certain properties of the boundary and of its environment (gray levels of the contour, of the background, ...). These features are then mapped through the use of a so called cost assignment function (CAF) into cost functions which are similar to the "'Potential"' instances of active contour models.

For instance, the following list records the cost functions $\mathcal{C}_x$ associated to some features quoted in the *Live-Wire* [12] and *Intelligent Scissors* [20] papers:

- $\mathcal{C}_g$: Gradient magnitude;
- $\mathcal{C}_d$: Direction of the gradient magnitude;
- $\mathcal{C}_L$: Laplacian feature;
- $\mathcal{C}_I$: Intensity on the positive (inside) side of the boundary,
- $\mathcal{C}_O$: Intensity on the negative (outside) side of the boundary,
- $\mathcal{C}_E$: Intensity on the boundary (edge).

The cost assignment process depends on how one wants to emphasize one or another value of the feature. For the gradient magnitude the inverse can for example be taken as a CAF in order to favor high contrasts, but a Gaussian function, centered on the gradient value one wants to highlight, could also be applied. For the Laplacian feature, the CAF is usually a zero-crossing detector. But many other types of CAF may be used.

Once satisfactory individual cost functions are defined, they are combined into a total cost function. Let us call potential the weighted sum of all the individual cost functions. On the directed graph-arc from a pixel p to an adjacent pixel q, the potential used by *Intelligent Scissors* [20] is defined by:

$$\begin{aligned}\mathcal{P}(p,q) = \omega_g \mathcal{C}_g(q) + \omega_L \mathcal{C}_L(q) + \omega_d \mathcal{C}_d(p,q) \\ + \omega_i \mathcal{C}_i(q) + \omega_o \mathcal{C}_o(q) + \omega_e \mathcal{C}_e(q)\end{aligned} \tag{1}$$

where each ω_x is the weight of the corresponding cost function in the previous list. The energy of a path to minimize with *Dijkstra* [10] is then defined by

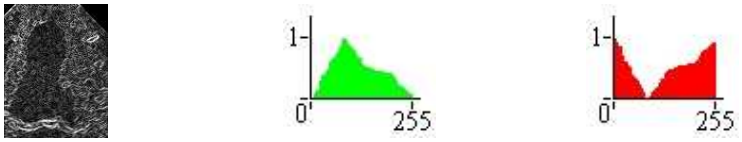
$$E_{\text{path}} = \sum_{(p,q)\in \text{path}} \mathcal{P}(p,q) \tag{2}$$

2.2 *Path-Cooling*

With the optimal path approach [10] it is not possible in general to extract a closed contour with only one seed point and one end point (a solution to this problem has been proposed in [4], based on saddle points). If the extracted path becomes too long, the optimal path search method will favor shorter paths cutting through the background: it is often necessary to set several points to draw the expected contour. *Path-Cooling* was introduced in *Intelligent Scissors* [1], as *Bordery Cooling* and achieves automatic generation of seed points. When a new seed point is generated, the boundary segment between this new point and the previous seed point is "frozen". A new seed point is generated when a pixel in the contour is considered stable enough. The stability criterion is function of both the time spent on the active boundary segment and the path coalescence (in other terms: how many times a point has been "drawn").

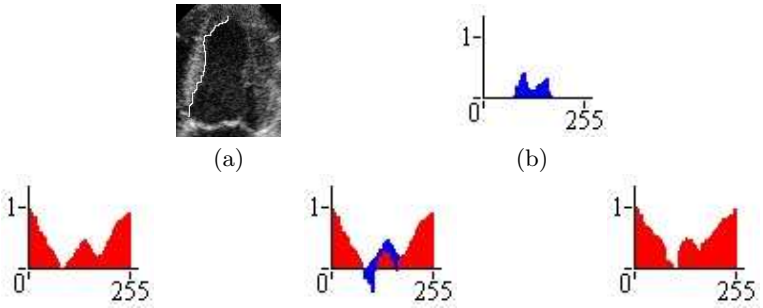
2.3 *On-The-Fly* training

For some features it is hard to decide without prior knowledge which values are to be preferred in the cost function. The notion of *On-The-Fly* training is introduced in *Intelligent Scissors* [20] and consists in adapting, during the extraction, the cost functions to the specificities of the desired contour, so as to track a contour with slowly changing properties. It is done for each feature independently from each other. The idea is the following: assuming that the user has drawn a long enough and valid boundary segment, the cost function of the feature has to be modified in order to favor contours with the same feature-values than those found on the segment, called training path. In practice, the process modifies directly the cost assignement function and not the feature itself. The *CAF* is basically represented as an histogram weighting, which is very easy to compute. During extraction, the *CAF* is iteratively modified by removing from it the training histogram (built on the training path) and scaling the result between 0 and 1. Figures 2 and 3 illustrate this process with the edge intensity feature. The advantage of *On-The-Fly* training is that the potential can be adapted in real-time during the extraction.



(a) (b) (c)

Fig. 2. Training initialization: (a) the feature trained; (b) its corresponding histogram at initialization; (c) its corresponding cost assignment function.



(a) (b) (c) (d) (e)

Fig. 3. Training iteration: (a) the trained path; (b) the histogram along the trained path; (c) the cost assignment function at iteration i; (d) the CAF minus the trained histogram; (e) the scaled CAF for iteration $i+1$.

3 Adaptations and improvements

In this section, the original contribution essentially lies in the introduction of a more general path search scheme and in the adaptation of the cooling speed.

3.1 Eikonal minimal path extraction

We worked on using the path extraction method of *Cohen and Kimmel* [6] in the framework of 2D-*Live-Wire* and *Intelligent Scissors* . This extraction method uses *Eikonal equation* for propagation. In our implementation we use this continuous formulation and compare it with *Dijkstra* as used in *Live-Wire* [12] and *Intelligent Scissors* [20].

The main idea of *Cohen and Kimmel* method [6] is that the potential and the graph are considered to be continuous, producing a sub-pixel path. In [6], contrary to the classical snake model [16] (but similarly to geodesic active contours [2]), the contour is parameterized by the arc-length parameter s, which means that $\|C'(s)\| = 1$, leading to a new energy form. Considering a simplified energy model without any second derivative term leads to the expression $E(C) = \int \{w\|C'\|^2 + P(C)\}ds$. Assuming that $\|C'(s)\| = 1$ leads to

$$E(C) = \int_{\Omega} \{w + P(C(s))\}ds. \tag{3}$$

We have now an expression in which the internal forces are included in the external potential. With this approach, detailed in [5], the energy to be minimized is defined as the integral of a strictly positive functional having low values close to the desired features. Some improvements on finding the minimal path for 2D and 3D images have been introduced in [7–9].

3.2 Comparison between Dijkstra and *Eikonal equation*

As explained in [6], the main difference between *Dijkstra* and *Cohen and Kimmel* design of the minimal path lies in the considered metric. In the first case, the minimal path minimizes the sum of the potential along the nodes of the graph (L_1 path), and in the second case, it minimizes the integral of the potential on the image domain (L_2 path). With *Dijkstra*, the image is considered as a graph in which each pixel is a node and the weights on the vertices are functions of the energy to minimize. This method uses dynamic programming to compute the optimal path [10]. *Cohen and Kimmel* approach keeps a continuous framework for the problem and computes the path by solving the *Eikonal equation* with the *Fast-Marching* algorithm [5, 22]. Even if *Eikonal* method uses integrals to compute the optimal contour, it is not slow. The average computing time ratio between both methods on synthetic and real images is near 0.9. And, because *Eikonal equation* produces a sub-pixel path (L_2 path), it is more accurate. The continuous formulation of Cohen and Kimmel method has the advantage to keep a more general framework for the energy definition, thereby allowing applications using other kind of potentials. It also easily includes an offset term w (see (3)) to constrain the regularity of the path, while it is more difficult with dynamic programming methods [17, 13].

3.3 *Path-Cooling* improvement

Path-Cooling methods are based on the idea that if a point in the active contour is stable enough, the corresponding boundary segment should be "frozen". Our approach uses the two counters described in [20], with an adaptation: the time history is updated by multiplying (and not adding) a scaled potential-driven factor instead of a simple gradient-driven factor. The multiplication has a weighting effect which gives more influence to pixels with a low potential. At pixel p and for iteration i of the cooling process, the time history $\mathcal{TH}_i$ is defined as

$$\mathcal{TH}_i(p) = \mathcal{TH}_{i-1}(p) + t_{i-1} \times [1 - \mathcal{P}(p)]$$

where t_i is the consecutive time the path was active until iteration i and $\mathcal{P}$ the penalty in (1).

The time history is useful if we consider that when the user does not move (redraw history locked, but time history incremented), he wants to emphasize the already drawn contour. But, as both thresholds are to be satisfied, even if the user moves very slowly (or not at all), the active path will freeze slowly. In

practice, the definition of the thresholds is not obvious and the main difficulty lies in the interpretation of the mouse motion, in terms of speed and acceleration.

We can either increase the cooling speed with the mouse cursor speed or decrease it. An argument to increase the mouse speed is the following: if the user moves fast it means there is no real difficulty with the drawing. But where the contour has lots of details, the user must act slowly if he has to define little areas. An argument to decrease the cooling speed is the following: the areas where the user goes slowly are the areas which are not very well defined. On the other hand, it induces that going too slowly in other areas (hesitating, for example) may create false paths. No choice is totally satisfying in a general case, but once one method is chosen, depending on the application, *Path-Cooling* is a very satisfying tool to extract closed boundaries, as shown in Fig. 4.

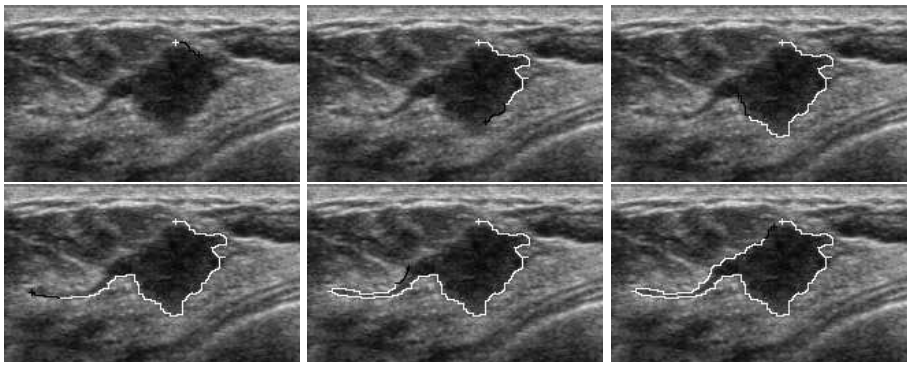


Fig. 4. *Path-Cooling* on an ultrasound image of a breast: iterations of the delineation of a tumor; the white cross is the seed point at iteration 0; the black cross is the mouse cursor; the white path is the "cooled" one; the black path is the currently drawn path between the mouse cursor and the new seed point at each iteration.

4 *On-The-Fly* training improvement

4.1 Training path

Training, as described in [20], is based on the distribution of the feature values on a validated contour segment. As depicted in Fig. 5, we tested three different definitions of the training path, which can either be:

1. the last section of the frozen contour; training is applied at each setting of a seed point (white points of the path in Fig. 5);
2. the last section of the active boundary; training is applied at each mouse movement, *i.e.* at each path extraction (black points of the path in Fig. 5);
3. the set of points where a mouse motion event is sent to the system; training is applied at each such event (black crosses in Fig. 5).

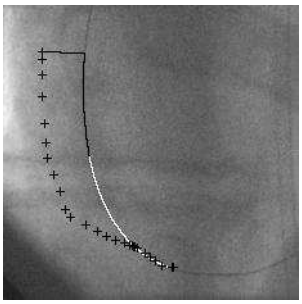


Fig. 5. Several training paths: on this fluoroscopy image, the cursor position is shown with crosses, the path frozen part is in white, and the current free path in black.

We found that the first definition leads to better results. Training is effective when a new seed point is manually or automatically set. This setting of a seed point can be seen as the validation of a path segment. The free path, because of its high variability and dependence on the local potential, is not a suitable training area, as well as for the mouse movement marker, because it constrains the mouse cursor to stay in the vicinity of the desired contour.

4.2 New way of training

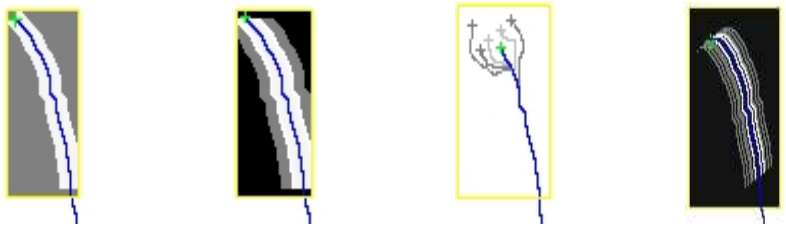
We have developed an improvement of the classical training method [15, 14]. In existing methods the training area is limited to a portion of the path itself and only takes a positive information into account (see section 2.3). The original technique we use is based on the addition of another training area based on "negative" information. The positive training area contains pixels that could belong to the contour, while the negative training area contains pixels that do not belong to the contour. The positive area has a reinforcing role while the negative area has a penalizing role in the cost assignment process. This improvement has two consequences: firstly, the new potential is more robust and secondly, the comparison of the distributions of the features (*i.e.* histograms) on each area helps adapting the weights of the individual cost functions in the total potential.

Definition of the positive and negative training areas. The main problem is to define suitable positive and negative areas that describe well enough which pixels belong to the contour and which don't. We tested four approaches for the negative area definition:

1. In the minimal box including the training path, the points closer to the path than a certain distance d are considered in the positive area, the other points of the box are considered in the negative area (see Fig. 6-(a));
2. In this box, the points closer to the path than a certain distance dp are considered in the positive area and the points further form the path than the

distance *dp* and closer to the path than a certain distance *dn* are considered in the negative area (see Fig. 6-(b));
3. the positive and negative areas are made from paths coming from the neighborhood of the click-position. The paths coming from nearest neighborhood form the positive area and the path coming from furthest neighborhood form the negative area (see Fig. 6-(c));
4. The training set of points is made from translations of the path: in the minimal box including the training path, nearest translations are the positive area and furthest translations are the negative area (see Fig. 6-(d)).

For each approach, it is possible to add a weighting function based on the distance to the training path. The first approach is not accurate enough for the



(a) (b) (c) (d)

Fig. 6. Different definitions of the positive and negative areas: Positive area in white, negative area in gray, other points of the box in black; (a) Positive area is a ribbon around the path, negative area is the rest of the box; (b) Positive area is a ribbon around the path, negative area is a ribbon around the positive area; (c) Areas built by the paths coming from a region around the mouse cursor; (d) Areas built with translations of the training path.

definition of the negative area. Indeed it is possible that other points of the box belong to another right path section. It is the motivation for introducing the second approach. The third method seemed *a priori* to be the most accurate to define a negative area. But all the paths quickly merge, thus limiting the size of the training set. Second and last approaches are very similar and give the best results with this difference that the first one is a continuous version of the second one. We therefore choose the last approach: the positive area is the set of p nearest translations and the negative area is the set of n next translations of the training path. The translation direction is given by the mean normal direction to the training path. The different translations are weighted according to their distance to the training path (see Fig. 7-(d)). The negative/positive areas are symmetric for the gradient magnitude and the edge intensity features (figure 7-(a)). But, in order to consider the non symmetric aspect of the inside and outside features, we adopt for them non symmetric training areas, depending on the direction of the gradient on the path, the path direction and the considered feature. See examples with Fig. 7-(b)-(c). The positive/negative training

(a) (b) (c) (d)

Fig. 7. Training areas: in white lines the positive area and in black lines the negative one. The longest line is the training path. (a) Symmetric; (b) Asymmetric for inside feature; (c) Asymmetric for outside feature; (d) Schematic weighting function.

sets of points are used to build two distinct histograms of the feature values. The function of these histograms is twofold: to build an adapted cost function for the feature (through the construction of an adapted CAF) and to adapt automatically the weight of the individual cost function in the global potential.

***On-The-Fly* adaptation of individual cost functions** Individual cost functions are built with a CAF applied to the feature. Training is used to dynamically modify this cost assignment function. With our method, the algebraic difference between the positive and the negative histograms is removed from the iteratively modified CAF and the resulting cost is normalized.

The positive and negative histograms are scaled the following way: firstly, to have a same scale for both training histograms, the negative histogram values are multiplied by the ratio between the number of points used to build the positive histogram and the number of points used to build the negative histogram:

$$H^-[i] = \frac{\text{card}\{H^+\}}{\text{card}\{H^-\}} \times H^-[i]$$

where H^+ and H^- are respectively positive and negative histograms. Then, both histograms are normalized using their common min/max.

The initialization used in [20] favors the gray values with highest occurrence in the feature. This is not a good initialization for images where there is a large homogeneous background as in Fig. 8-(a). We prefer an approach that does not carry any *a priori* about the expected feature values: the CAF is initialized with a flat line, giving the same cost to every pixels. So, the CAF obtained during the process depends only on the past and the present training data.

If the training contour is not uniform enough, producing a too wide positive histogram, the classical approach will favor feature values that are perhaps not specifically relevant to the expected contour. The negative information used in our method leads to a description of what the values should not be. Combining both positive and negative information thus produces a more suitable cost function. This effect is shown in Fig. 8 on the septum wall of an echocardiographic left-ventricle image.

The trained feature is the inside intensity one (Fig. 8-(b)). The classical method uses a CAF initialized with the inverse of the distribution of the feature,

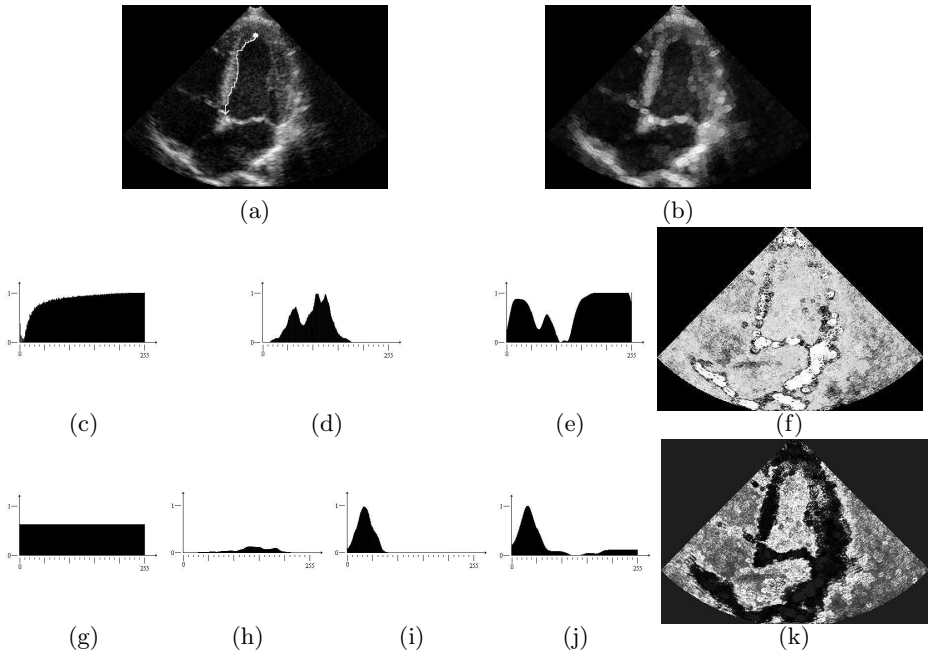


Fig. 8. Training on the septum wall of an echographic left-ventricle image: (a) echographic left-ventricle image; (b) inside Feature $\mathcal{C}_i$; (c to f) **classical training**: (c) initial CAF; (d) training histogram; (e) resulting CAF with (f) corresponding cost function; (g to k) **improved training**: (g) initial CAF; (h) positive and (i) negative training histograms; (j) resulting CAF with (k) corresponding cost function.

where black levels are predominant and thus favored (Fig. 8-(c)). The training histogram is scaled between 0 and 1 (Fig. 8-(d)) and removed from the initial CAF to build a new CAF (Fig. 8-(e)) favoring very specific values. With this example the training path is homogeneous and the resulting cost function (Fig. 8-(f)) is good. But with a non uniform enough training contour, the resulting potential will not be consistent. Our method initializes the CAF (Fig. 8-(g)) with an arbitrary value between 0 and 1. The positive and negative histograms (Fig. 8-(h) and 8-(i)) are jointly scaled and the difference between them is removed from the initial CAF producing a less specific but carrying more accurate information histogram (Fig. 8-(j)). Thus the new cost function (Fig. 8-(k)) is more relevant.

On-The-Fly adaptation of the individual cost functions is shown in Fig. 9, where iterations of the modification of the CAF of the feature $\mathcal{C}_e$ are shown.

Adaptation of the weights. One of the main advantage of using positive and negative training areas is the evaluation of the differences between the histograms, which helps adapting the weight of the corresponding individual cost function in the global potential. If the histograms are distinct enough, we can assume that the considered feature is discriminating enough and that its weight

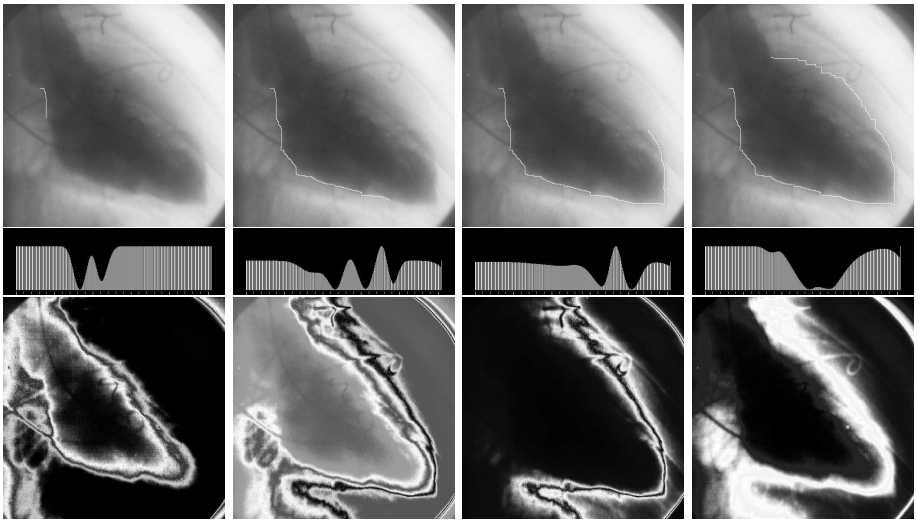


Fig. 9. Results of the training on a left ventricle image: First row - iterations of the real-time contour extraction with training, on a X-Ray image of the heart left-ventricle; second row - modification of the CAF of the feature $\mathcal{C}_e$ at the same iterations; third row - corresponding feature potential $\mathcal{C}_e$ at the same iterations.

should be more important. In a first approach we take the mean difference between both histograms as dissimilarity criterion, expressed as:

$$c = \frac{1}{256} \sum_{i=0}^{255} |H^+[i] - H^-[i]|.$$

Indeed, if the histograms are very similar this criterion will be very low and if they are different, it will be high. In practice, this criterion often produces very low values and hardly ever values above 0.5. Hence a stretching function is used to transform it into a weight ω_c associated with the cost function c in the total potential. See on Fig. 10 examples of stretching functions.

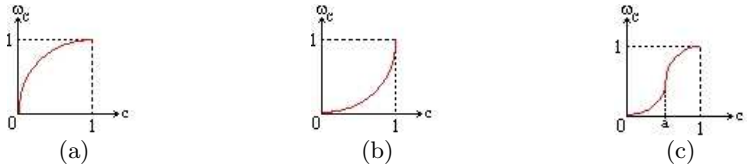


Fig. 10. Examples of stretching functions: (a) Privileges small values of the cost; (b) penalizes small values of the cost; (c) penalizes values of the cost below a and favors values of the cost above a.

5 Discussion

We have described a very general method of path extraction. This path extraction is interactive, works in real-time, is able to achieve sub-pixel accuracy and makes use of on-line training to adapt its internal parameters. Figure 11 shows several results of the developed tool on different medical images. But of course, it can be applied to any type of digital images. The described work is not focused

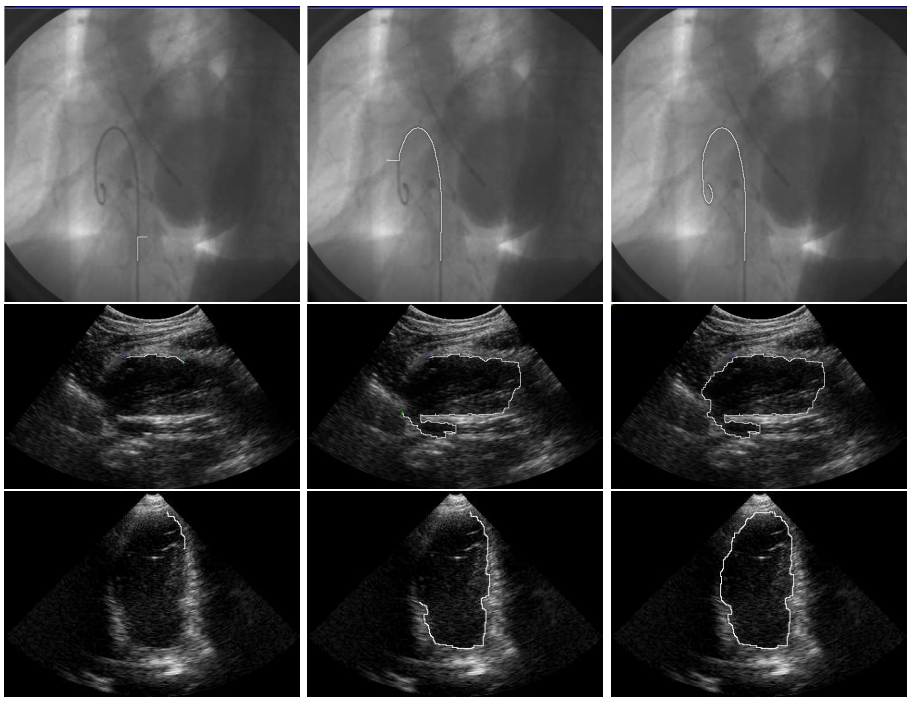


Fig. 11. Results on different datasets: First row - Extraction of a guide-wire in a fluoroscopy image; second row - tumor delineation in an ultrasound breast image; third row - extraction of the left ventricle in an echocardiographic image.

on the best features to compute for the path extraction problem at hand (a low-level task subjects to a lot of ad-hoc choices), but is rather trying to make the best use of the available features in order to give satisfactory results to the user. The "live" satisfaction of the user is a crucial input for the method and several methods have been proposed to "measure" it (see the discussions in sections 3.3 and 4.1). The proposed training method is clearly able to adapt the use of each individual feature, because the training process is rather simple and general, and because we make simultaneous use of both positive (reinforcing) and negative (penalizing) information, see section 4.2. This training can also work in real-time because it acts on the cost assignement function level and not in the lower

feature level (which is more time consuming to compute). Thus, to make the best use of our work, a lot of features should be computed (once, at the start of the extraction process) and proposed for selection to the method. Of course, our work is not limited by the sample list of features presented in 2.1 and both "general" and specific features should compete. The competition we propose is also based on the automatic setting of the individual weight applied to each feature (see section 4.2). Using positive and negative information allows us to estimate the pertinence of each feature and thus to adapt the corresponding weight accordingly. As presented here, this automatic weight adaptation requires further improvements to fully take advantage of this very nice potentiality. Then, the proposed method could be applied to a database of images using a wide range of features, leading to an "automatic" selection (provided that an experienced user has used the tool) of the most relevant features for the current task.

6 Conclusion

We have developed an interactive, real-time and user-guided image segmentation software. The user has a full control of the drawing process (which is very important for physicians). Also, the non-specialist user has the possibility to outline the contour of an object in an image without a very precise drawing, for example with the track-ball on an echograph. We are currently testing this feature for a dedicated echographic vessel detection application.

Some improvements have been brought to the bibliographical background of interactive extraction of optimal path. A very general optimal path extraction method has been efficiently used, producing in real-time very precise paths. Finally, a new method of *On-The-Fly* training has been developed to adapt, during extraction of the contour, the individual cost-functions and their relative weights in the total potential.

References

1. W. Barrett and E. Mortensen, "Interactive Live-Wire boundary extraction", *Medical Image Analysis*, vol. 1, no. 4, pp. 331–341, 1997.
2. V. Caselles, R. Kimmel, G. Sapiro, "Geodesic active contours", in *Proc. of ICCV'95*, Cambridge, USA, pp. 694–699, 1995.
3. L. D. Cohen, "On active contour models and balloons", *CVGIP: Image Understanding*, vol. 53, no. 2, pp. 211-218, March 1991.
4. L. D. Cohen and R. Kimmel, "Global Minimum for Active Contour Models: A Minimal Path approach", in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*, pp. 666–673, 1996.
5. L. D. Cohen and R. Kimmel, "Fast marching the global minimum of active contours", in *Proc. IEEE International Conference on Image Processing (ICIP'96)*, pp. I:473–476, 1996.
6. L.D. Cohen and R. Kimmel, "Global Minimum for Active Contour Models: A Minimal Path approach", *IJCV*, vol. 24, no. 1, pp. 57–78, Aug. 1997.

7. T. Deschamps, *Curve and Shape Extraction with Minimal Path and Level-Sets techniques - Applications to 3D Medical Imaging*, Ph.D. thesis, Université Paris IX, December 2001.
8. T. Deschamps and L.D. Cohen "Minimal paths in 3D images and application to virtual endoscopy", in *Proc. sixth European Conference on Computer Vision*, pp. II:543-557, 2000.
9. T. Deschamps and L.D. Cohen, "Fast extraction of minimal paths in 3D images and applications to virtual endoscopy", *Medical Image Analysis*, vol. 5, no. 4, pp. 281–299, December 2001.
10. E.W. Dijkstra, "A note on two problems in connection with graphs", *Numerische Mathematic*, vol. 1, pp. 269–271, 1959.
11. A.X. Falcao, J.K. Udupa, S. Samarasekera, B.E. Hirsch, "User-steered image boundary segmentation", in *SPIE Proceedings*, vol. 2710, pp. 278–288, 1996.
12. A.X. Falcao, J.K. Udupa, S. Samarasekera, S. Sharma, B.E. Hirsch, R. de A. Lotufo, "User-Steered Image Segmentation Paradigms: Live-Wire and Live-Lane", *GMIP*, vol. 60, no. 4, pp. 233–260, 1998.
13. D. Geiger, A. Gupta, L. Costa, J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours", *IEEE Trans. PAMI*, vol. 17, no. 3, pp. 294–302, March 1995.
14. M. Greff, O. Gérard, T. Deschamps, "Adaptation of potential terms in real-time optimal path extraction", Patent Pending, April 2001, FR 01 401 696.8.
15. M. Greff, "Real-time user-guided image segmentation", Internship report, Institut National des Télécommunications, July 2001.
16. M. Kass, A. Witkin, D. Terzopoulos, "Snakes: Active contour models", *IJCV*, vol. 1, no. 4, pp. 321–331, 1988.
17. N. Merlet, J. Zerubia, "A curvature-dependent energy function for detecting lines in satellite images", in *Proc. of Scandinavian Conference on Image Analysis*, Tromso, Norway, pp. 699–706, May 1993.
18. E. Mortensen, B. Morse, W. Barrett, J.K. Udupa, "Adaptive boundary detection using live-wire two-dimensional dynamic programming", in *IEEE Proc. of Computers in Cardiology*, pp. 635–638, October 1992.
19. E.N. Mortensen and W.A. Barrett, "Intelligent Scissors for Image Composition", in *Proc. of Computer Graphics, SIGGRAPH'95*, pp. 191–198, 1995.
20. E.N. Mortensen and W.A. Barrett, "Interactive Segmentation with Intelligent Scissors", *GMIP*, vol. 60, no. 5, pp. 349–384, September 1998.
21. E.N. Mortensen and W.A. Barrett, "Toboggan-Based Intelligent Scissors with a Four Parameter Edge Model", in *Proc. of the IEEE CVPR'99*, pp. II:452–458, 1999.
22. J.A. Sethian, *Level set methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences*, Cambridge University Press, 1999.