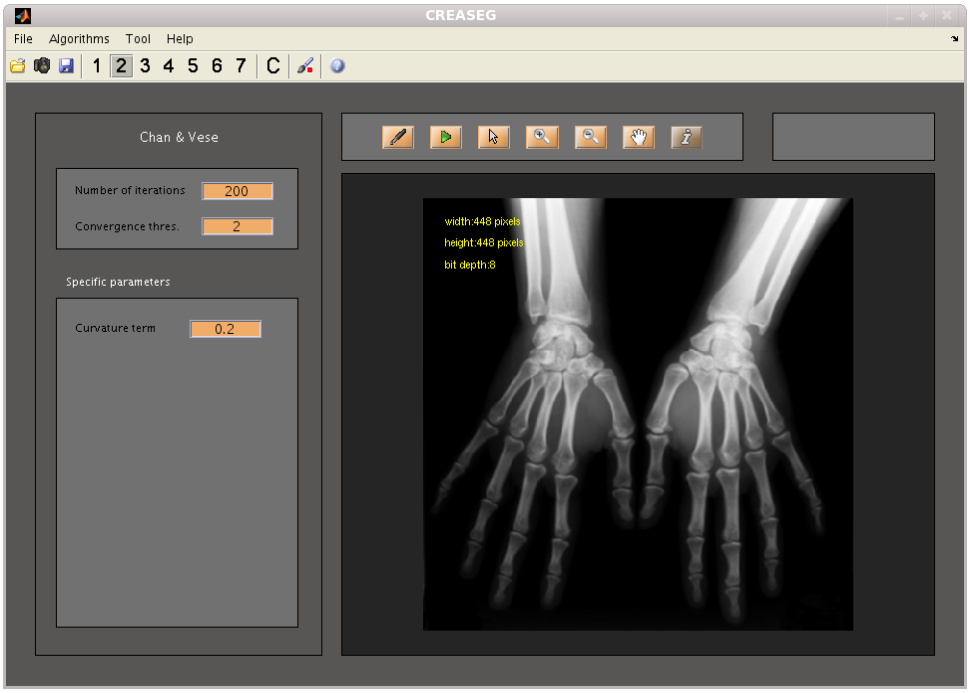
CREASEG

Level-set segmentation platform



Creatis

Authors: Olivier Bernard, Associate Professor
Thomas Dietenbeck, PhD Student
Denis Friboulet, Professor

Table of Contents

Introduction

This software is a computer program whose purpose is to evaluate the performance of different level-set based segmentation algorithms in the context of image processing (and more particularly on biomedical images).

The software has been designed for two main purposes.

- firstly, CREASEG allows you to use six different level-set methods. These methods have been chosen in order to works with a wide range of level-sets. You can select for instance classical methods such as Caselles or Chan & Vese level-set, or more recent approaches such as the one developped by Lankton or Bernard.

- finally, the software allows you to compare the performance of the six level-set methods on different images. The performance can be evaluated either visually, or from measurements (*e.g.* using the Dice coefficient or the PSNR value) between a reference and the results of the segmentation.

Copyright

CREASEG is an open source software license intended to give users significant freedom to modify and redistribute the software licensed hereunder.

The exercising of this freedom is conditional upon a strong obligation of giving credits for everybody that distributes a software incorporating a software ruled by the current license so as all contributions to be properly identified and acknowledged.

CREASEG is citationware. If you are publishing any work, where this program has been used, or which used one of the proposed level-set algorithms, please remember that it was obtained free of charge. You must reference the paper shown below and the name of the program CREASEG must be mentioned in the publication.

Olivier Bernard or Creatis do not offer any support for this product whatsoever. The program is offered free of charge. The executable program is copyrighted freeware by Olivier Bernard.

MATLAB is a trademark of The MathWorks, Inc. Trademarks of other companies and/or organizations mentioned in this documentation and web-site appear for identification purposes only and are the property of their respective companies and/or organizations.

Paper to be cited when using the Creaseg software

T. Dietenbeck, M. Alessandrini, D. Friboulet, O. Bernard. *CREASEG: a free software for the evaluation of image segmentation algorithms based on level-set.* In *IEEE International Conference On Image Processing.* Hong Kong, China, 2010.

Paper to be cited when using the level-set algorithm individually

Bernard Method

O. Bernard, D. Friboulet, P. Thevenaz, and M. Unser. *Variational B-Spline level-set: A linear filtering approach for fast deformable model evolution. IEEE Trans. Image Process.*, volume 18, no.06, pp.1179–1191, 2009.

Caselles Method

V. Caselles, R. Kimmel, and G. Sapiro. *Geodesic active contours. Int. J. of Computer Vision*, volume 22, pp.61–79, 1997.

Chan & Vese Method

T. Chan and L. Vese. *Active contours without edges. IEEE Trans. Image Process.*, volume 10, no.02, pp.266–277, 2001.

Lankton Method

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

Li Method

C. Li, C.-Y. Kao, J. C. Gore, and Z. Ding. *Minimization of region-scalable fitting energy for image segmentation. IEEE Trans. Image Process.*, volume 17, no.10, pp.1940–1949, 2008.

Shi Method

Y. Shi and W. C. Karl. *A real-time algorithm for the approximation of level-set based curve evolution. IEEE Trans. Image Process.*, volume 17, no.05, pp.645–656, 2008.

1 Interface presentation

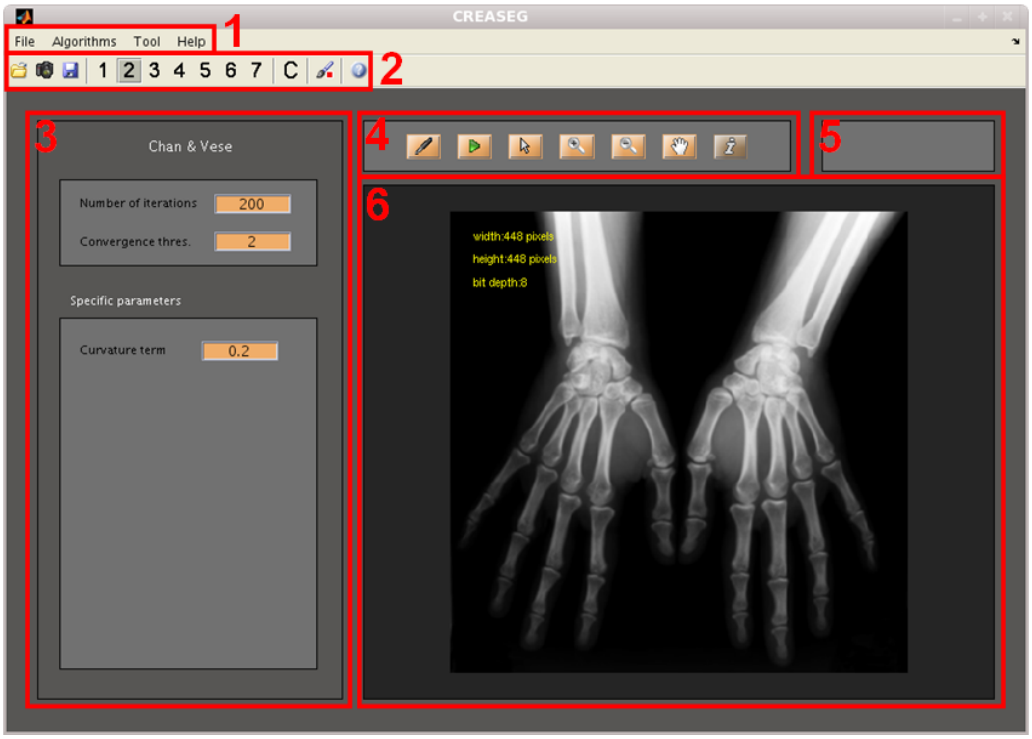


Fig. 1: *The interface*

The interface is divided into 6 parts. From fig. 1, we can identify:

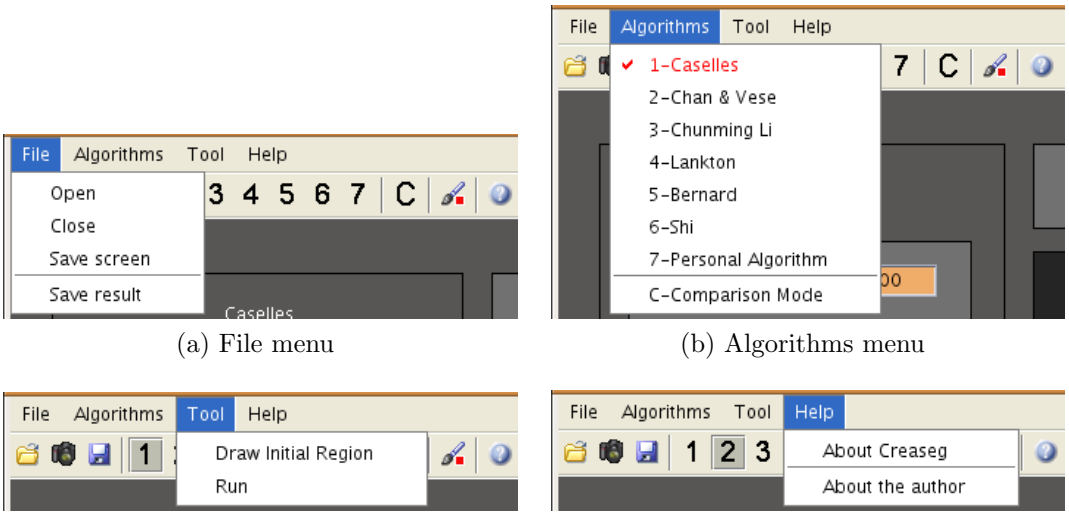
- a menu (area 1),
- a toolbar (area 2),
- the algorithm panels (area 3),
- the figure's icons (area 4),
- the mouse pointer informations (area 5),
- the image axis (area 6).

1.1 Menu

The interface menu is divided into four sub-menu:

- File (fig.2a): this item is used to open an image (File→Open), to save the image display into a file (File→Save screen), to save all the data (image, level-set, method used) in a ".mat" file (File→Save result), and to close the interface (File→Close).

- Algorithms (fig.2b): this item allows you to choose between six level-set algorithms, your personal method (that you have to implement) and the comparison mode. The chosen algorithm is highlighted in red and checked. If you are in comparison mode, the selected algorithms are highlighted in green while the *'C-Comparison mode'* is highlighted in red.
- Tool (fig.2c): this item allows you to either display the panel used to draw an initial contour into the image area or to run the selected algorithm.
- Help (fig.2d): this item allows you to either access to CREASEG website (File→About CREASEG) or to the author of the interface website (File→About the author).



(a) File menu (b) Algorithms menu

(c) Tool menu (d) Help menu

Fig. 2: *Presentation of the four sub-menus*

1.2 Toolbar

The toolbar allows you to perform the following actions (from left to right as shown in fig. 3):

- Open: open an image (of different format such as .jpg, .png, .bmp, .dcm, ...).
- Save screen: save the image and the contours that are displayed (in a single image file whose extension is chosen by the user).
- Save results: save the image, the final level-set and the method used in a .mat file.
- Number 1 to 7: display the panel of the selected algorithm.
- C: display the *'Comparison Mode'* panel.
- Brush: change the color of the evolving contour. The color will changed in the following order: Red (default) - Green - Blue - Yellow - White - Black.
- Help: open CREASEG website.

Fig. 3: *Toolbar*

1.3 Algorithm panel

Each algorithm s linked to a panel in which are displayed the parameters of the method. Each panel is splited into two parts as illustrated in fig. 4:

1. global parameters: parameters that are shared by each method:
   - number of iterations: maximum number of iterations after which the algorithm should stop (if it has not converged already);
   - convergence threshold: Number of pixel used in the convergence test. At the end of each iteration, a convergence test is performed: if during five consecutive iterations, the contour has not evolved more than the threshold value, the convergence is reached.

     Note: Because Shi algorithm has its own convergence test, this parameter is not available for this method.
2. specific parameters: parameters that will depend on the method used. More details are given in section 2.

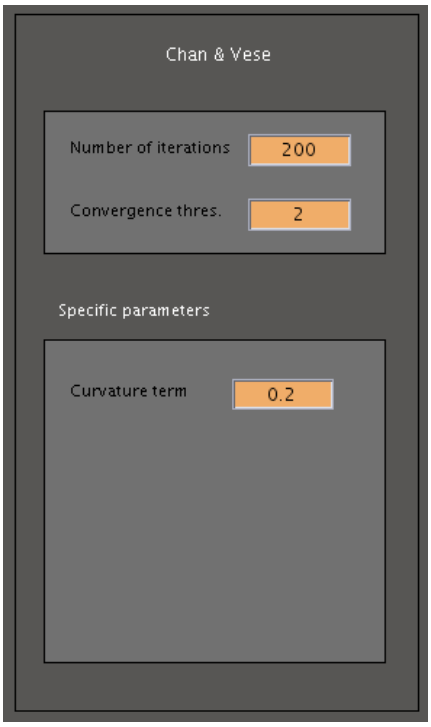


Fig. 4: *Example of the panel linked to Chan & Vese method*

1.4 Figure icons

The figure icons allow you to perform the following tasks (from left to right as shown in fig. 5):

- display the panel used to draw an initial contour (see fig. 6a). 6 different types of initial contour can be selected:
  - single rectangle: left click at one of the rectangle's corner then drag until the opposite corner is reached. If a new rectangle is drawn, it will delete the previous one;

Fig. 5: *Figure icons*

  - multiple rectangles: left click at one of the rectangle's corner then drag until the opposite corner is reached to draw. All the drawn rectangles are saved for the initialization;
  - ellipse: left click at one of the rectangle's corner then drag until the opposite corner is reached to draw the ellipse enclosed inside the rectangle. If a new ellipse is drawn, it will delete the previous one;
  - multiple ellipses: left click at one of the rectangle's corner then drag until the opposite corner is reached to draw. All the drawn ellipses are saved for the initialization;
  - one contour: draw one contour by adding points (left click). When the initial contour is completed, right click to end;
  - multiple contour: draw a specific contour by adding points (left click). When the contour is completed, right click to begin a new contour;

  An example of an initialization using multiple ellipses is given in fig. 6b;

- run the algorithm (while an algorithm is running, the buttons are disabled and the mouse pointer pass to watch mode);
- zoom in (to a factor of 2);
- zoom out (to a factor of 2);
- move the image (works only if the image is zoomed);
- display image informations such as its size. These informations are displayed on the top left corner of the image (see also paragraph 1.6).

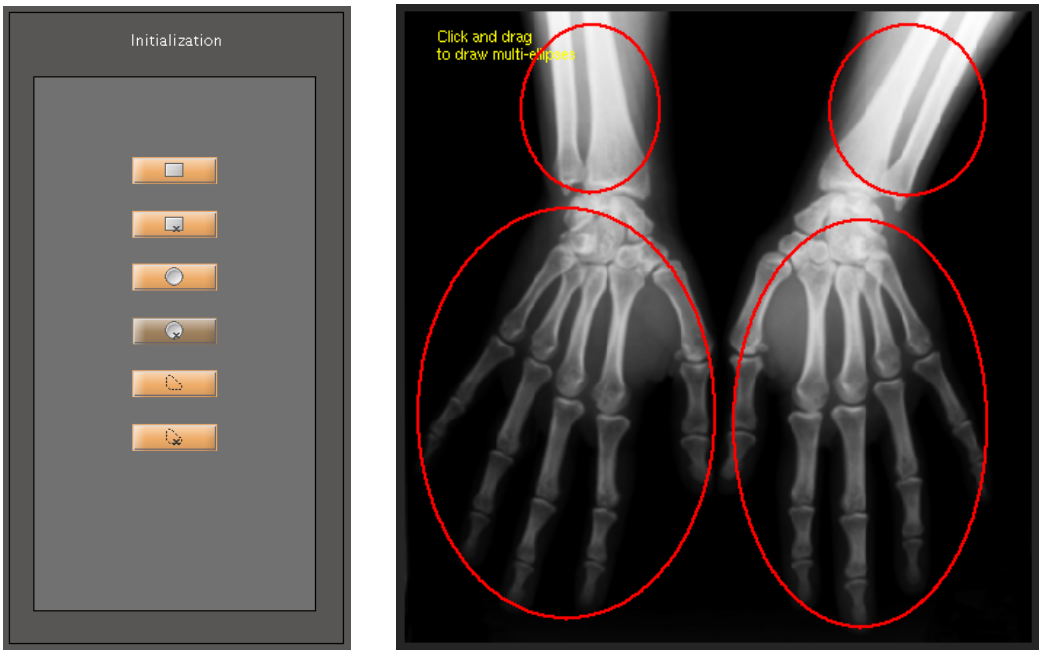
1.5 Mouse pointer informations

This panel is used to:

- display in real time both the mouse positioning inside the image reference and the corresponding pixel value (see fig. 7a).
- display informations about the procedure to follow in order to draw a contour(see fig. 7b).

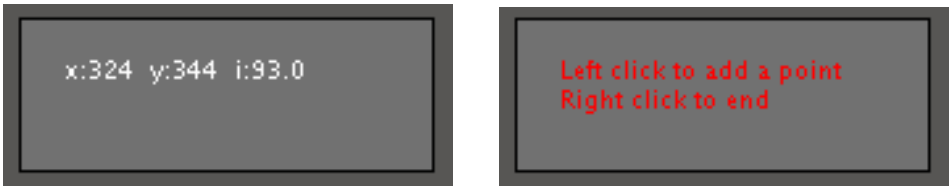
1.6 Image axis

When an image is opened (using either $File \rightarrow Open$ or the *Open* button), the image will be displayed on the right hand side of the interface (see fig. 8). Although all kind of images can be read, they will always be converted in gray level. Moreover the image will be resized to fit the width or height of the axis.



(a) (b)

Fig. 6: *(a) Initialization panel; (b) Example of an initialization using multiple ellipses*



(a) (b)

Fig. 7: *Mouse pointer informations: (a) Informations about the mouse positioning; (b) Informations about the procedure to follow*

The top left corner of the image is used to display different types of messages:

- image information such as the size, bitdepth or resolution (if available)
- error message when some of the parameters are missing (*e.g.* an initial contour, a reference,...)
- the current or final number of iterations of the running algorithm.

1.7 How to run an algorithm

To run an algorithm, you have to perform the following steps:

1. open an image using either File→Open or the open button from the toolbar.

Fig. 8: *Example of an image displayed on the right-hand side of the interface*

2. display the initialization panel by clicking on the *'Draw initial region'* icon in the figure icons panel. Choose the initialization type and create the contour accordingly (see section 1.4 for more details on initialization).

3. choose the algorithm you want to run (via the *'Algorithms'* menu or the corresponding button in the toolbar) and set its parameters.

4. run the algorithm by clicking on the *'Run'* icon present in the figure's icon panel.

When the algorithm has converged (because of the convergence test or because the maximum iteration number is reached) the result is put in place of the initial contour, allowing you to run again the algorithm from where it stopped.

2 Implemented methods

This section gives a short description of the implemented algorithms. For further details about these methods, the reader is invited to refer to the cited algorithms. For each method, we give the energy criterion which is minimized, the derived evolution equation and the main properties of the method.

2.1 Caselles [2]

Energy criterion

$$E(\Gamma) = \int_0^1 g(I(\Gamma(q)))\|\Gamma'(q)\|dq, \tag{1}$$

where

$$g(I) = \frac{1}{1 + \|\nabla(G * I)\|^2}, \tag{2}$$

$I(\cdot)$ corresponds to the image intensity, Γ is the parametric curve and G is a gaussian filter of variance 1.

Evolution equation

$$\frac{\partial \phi}{\partial t}(\mathbf{x}) = g(I(\mathbf{x}))\|\nabla\phi(\mathbf{x})\|(c + \kappa) + \nabla g(I(\mathbf{x}))\nabla\phi(\mathbf{x}). \tag{3}$$

where $\kappa = div\left(\frac{\nabla\phi(\mathbf{x})}{\|\nabla\phi(\mathbf{x})\|}\right)$ corresponds to the curvature of the evolving contour and c is a constant that acts as a balloon force.

Properties

- This algorithm is a contour-based method *i.e.* the gradient of the image is used to compute the force function. The curve will thus be driven to regions with high gradient.
- This method does not require any regularization term as it is intrinsic to the method.
- ϕ is implemented as a signed distance function and is reinitialized at each iteration.

Specific parameters

This algorithm has one specific parameter that can be modified from the corresponding panel:

- the propagation term c: it is as a constant that acts as a balloon force pushing the contour either inward or outward (default value is set to 1).

2.2 Chan & Vese [3]

Energy criterion

$$E(\phi) = \int_\Omega F(I(\mathbf{x}), \phi(\mathbf{x}))\, dx + \lambda \int_\Omega \delta(\phi(\mathbf{x}))\|\nabla\phi(\mathbf{x})\|dx, \tag{4}$$

where δ is the dirac function and

$$F(I(\mathbf{x}), \phi(\mathbf{x})) = H(\phi(\mathbf{x}))(I(\mathbf{x}) - v)^2 + (1 - H(\phi(\mathbf{x})))(I(\mathbf{x}) - u)^2, \tag{5}$$

H is the Heaviside function, u and v are two parameters updated at each iteration as follows:

$$u = \frac{\int_\Omega (1 - H(\phi(\mathbf{x}))) \cdot I(\mathbf{x})\, dx}{\int_\Omega 1 - H(\phi(\mathbf{x}))\, dx} \tag{6}$$

$$v = \frac{\int_\Omega H(\phi(\mathbf{x})) \cdot I(\mathbf{x})\, dx}{\int_\Omega H(\phi(\mathbf{x}))\, dx} \tag{7}$$

The first integral of (4) corresponds to a data attachment term and the second is a regularization term that minimizes the length of the curve thereby smoothing the evolving contour in the course of its evolution.

Evolution equation

$$\frac{\partial\phi}{\partial t}(\mathbf{x}) = \delta(\phi(\mathbf{x}))((I(\mathbf{x}) - v)^2 - (I(\mathbf{x}) - u)^2) + \lambda\delta(\phi(\mathbf{x}))\kappa. \tag{8}$$

Properties

- This algorithm is a region-based method. It tends to separate the image into two homogeneous region (according to their mean value).
- The evolution is only computed on the narrow-band of the level-set thus making it sensitive to initialization.
- ϕ is implemented as a signed distance function and is reinitialized at each iteration.

Specific parameters

This algorithm has one specific parameter that can be modified from the corresponding panel:

- the curvature term λ: it weights the influence of the regularization term of equation (8) (default value is set to 0.2).

2.3 Li [4]

Energy criterion

$$\begin{aligned} E(\phi) = \quad & \lambda_1 \int_\Omega \int_\Omega K_\sigma(\mathbf{x}-\mathbf{y})|I(\mathbf{y}) - f_1(\mathbf{x})|^2 H(\phi(\mathbf{x}))\, d\mathbf{y} d\mathbf{x} \\ & + \lambda_2 \int_\Omega \int_\Omega K_\sigma(\mathbf{x}-\mathbf{y})|I(\mathbf{y}) - f_2(\mathbf{x})|^2 (1 - H(\phi(\mathbf{x})))\, d\mathbf{y} d\mathbf{x} \\ & + \nu \int_\Omega \delta(\phi(\mathbf{x}))\|\nabla\phi(\mathbf{x})\| d\mathbf{x} + \mu \int_\Omega \tfrac{1}{2}\left(\|\nabla\phi(\mathbf{x})\| - 1\right)^2 d\mathbf{x}, \end{aligned} \tag{9}$$

where $I(\mathbf{x})$ is the image intensity at pixel $\mathbf{x}$, H is the Heaviside function, K_σ is a gaussian kernel defined as:

$$K_\sigma(\mathbf{u}) = \frac{1}{(2\pi)^{n/2}\sigma^n} e^{-\|\mathbf{u}\|^2/2\sigma^2}, \tag{10}$$

with a scale parameter $\sigma > 0$. f_1 and f_2 are two functions centered at pixel $\mathbf{x}$ and computed at each iteration as:

$$f_1(\mathbf{x}) = \frac{K_\sigma * (H(\phi(\mathbf{x}))I(\mathbf{x}))}{K_\sigma * H(\phi(\mathbf{x}))}, \tag{11}$$

$$f_2(\mathbf{x}) = \frac{K_\sigma * ((1 - H(\phi(\mathbf{x})))I(\mathbf{x}))}{K_\sigma * (1 - H(\phi(\mathbf{x})))}. \tag{12}$$

λ_1 and λ_2 are two constants that have been set to 1 in the interface.

The two first integrals of (9) correspond to data attached term, which are localized around each point x thanks to the Gaussian kernel K_σ. The third integral corresponds to the usual regularization term that smoothes the curve during its evolution. The last integral is a regularization term that forces the level-set to keep signed distance properties over the evolution process.

Evolution equation

$$\begin{aligned} \frac{\partial\phi}{\partial t}(\mathbf{x}) = & \delta(\phi(\mathbf{x}))\left(\lambda_1 \int_\Omega K_\sigma(\mathbf{x}-\mathbf{y})|I(\mathbf{y}) - f_1(\mathbf{x})|^2\, d\mathbf{y} + \lambda_2 \int_\Omega K_\sigma(\mathbf{x}-\mathbf{y})|I(\mathbf{y}) - f_2(\mathbf{x})|^2\, d\mathbf{y}\right) \\ & + \nu\delta(\phi(\mathbf{x}))\kappa + \mu\left(\nabla^2\phi(\mathbf{x}) - \kappa\right), \end{aligned} \tag{13}$$

Properties

- Because of the localization introduced by f_1, f_2 and K_σ, this algorithm is able to segment inhomogeneous objects.
- This algorithm segments the whole image.
- ϕ is a signed distance function.

Specific parameters

This algorithm has three specific parameters that can be modified from the corresponding panel:

- the curvature term ν: it weights the influence of the regularization of the evolving contour (default value is set to 0.003).
- the level-set regularization term μ: it weights the influence of the regularization of the shape of the level-set (default value is set to 1).
- the variance of the gaussian kernel σ (default value is set to 7).

Note: The parameter λ_1 and λ_2 have a fixed value of 1.

2.4 Lankton [5]

Energy criterion

$$E(\phi) = \int_\Omega \delta(\phi(\mathbf{x})) \int_\Omega B(\mathbf{x},\mathbf{y}) \cdot F(I(\mathbf{y}),\phi(\mathbf{y}))\, d\mathbf{y} d\mathbf{x} + \lambda \int_\Omega \delta(\phi(\mathbf{x})) \|\nabla\phi(\mathbf{x})\| d\mathbf{x}, \tag{14}$$

where δ is the Dirac function, B is a ball of radius r centered at point $\mathbf{x}$ and defined as follow:

$$B(\mathbf{x},\mathbf{y}) = \begin{cases} 1, & \|\mathbf{x}-\mathbf{y}\| \leq r \\ 0, & otherwise, \end{cases} \tag{15}$$

and

$$F(I(\mathbf{y}),\phi(\mathbf{y})) = \begin{cases} H(\phi(\mathbf{y}))(I(\mathbf{y})-v(\mathbf{x}))^2 + (1-H(\phi(\mathbf{y})))(I(\mathbf{y})-u(\mathbf{x}))^2, & \text{Chan \& Vese feature,} \\ (v(\mathbf{x})-u(\mathbf{x}))^2, & \text{Yezzi feature,} \end{cases} \tag{16}$$

where H is the Heaviside function, $u(\mathbf{x})$ and $v(\mathbf{x})$ are two functions updated at each iteration as follows:

$$u(\mathbf{x}) = \frac{\int_\Omega B(\mathbf{x},\mathbf{y}) \cdot (1-H(\phi(\mathbf{y}))) \cdot I(\mathbf{y})\, d\mathbf{y}}{\int_\Omega B(\mathbf{x},\mathbf{y}) \cdot (1-H(\phi(\mathbf{y})))\, d\mathbf{y}} \tag{17}$$

$$v(\mathbf{x}) = \frac{\int_\Omega B(\mathbf{x},\mathbf{y}) \cdot H(\phi(\mathbf{y})) \cdot I(\mathbf{y})\, d\mathbf{y}}{\int_\Omega B(\mathbf{x},\mathbf{y}) \cdot H(\phi(\mathbf{y}))\, d\mathbf{y}} \tag{18}$$

The first integral of (14) corresponds to a data attached term and the second is the usual regularization term that smoothes the contour.

Evolution equation

$$\frac{\partial\phi}{\partial t}(\mathbf{x}) = \delta(\phi(\mathbf{x})) \int_\Omega B(\mathbf{x},\mathbf{y}) \cdot \nabla_\phi F(I(\mathbf{y}),\phi(\mathbf{y}))\, d\mathbf{y} + \lambda\delta(\phi(\mathbf{x}))\kappa, \tag{19}$$

where

$$\nabla_\phi F(I(\mathbf{y}),\phi(\mathbf{y})) = \begin{cases} \delta(\phi(\mathbf{y}))((I(\mathbf{y})-v(\mathbf{x}))^2 - (I(\mathbf{y})-u(\mathbf{x}))^2), & \text{Chan \& Vese feature,} \\ \delta(\phi(\mathbf{y}))\left(\frac{(I(\mathbf{y})-v(\mathbf{x}))^2}{A_v} - \frac{(I(\mathbf{y})-u(\mathbf{x}))^2}{A_u}\right), & \text{Yezzi feature,} \end{cases} \tag{20}$$

where A_u and A_v are the area of the local interior and local exterior regions respectively given by

$$A_u = \int_\Omega B(\mathbf{x},\mathbf{y}) \cdot (1-H(\phi(\mathbf{x})))\, dy \tag{21}$$

$$A_v = \int_\Omega B(\mathbf{x},\mathbf{y}) \cdot H(\phi(\mathbf{x}))\, dy \tag{22}$$

Properties

- This algorithm is a region-based method.
- Its feature term is computed locally. This property allows the algorithm to segment non homogeneous objects. However this make the method sensitive to initialization.
- ϕ is implemented as a signed distance function and is reinitialized at each iteration.

Specific parameters

This algorithm has four specific parameters that can be modified from the corresponding panel:

- the feature type: the user can choose between Yezzi or Chan & Vese feature type (see equations (16), (20)).
- the neighborhood type: the user can choose between a circle or square neighborhood when computing the local statistics.
- the curvature term λ: it weights the influence of the regularization term of the evolving contour (default value is set to 0.2).
- the radius term r: it allows to fix the radius size of the neighborhood defined by B (default value is set to 9).

2.5 Bernard [6]

Model

Let Ω be a bounded open subset of $\mathbb{R}^d$ and let $f : \Omega \mapsto \mathbb{R}$ be a given d-dimensional image. In the B-spline level-set formalism, the evolving interface $\Gamma \subset \mathbb{R}^d$ is represented as the zero level-set of an implicit function $\phi(\cdot)$ expressed as a linear combination of B-spline basis functions

$$\phi(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^d} c[\mathbf{k}]\, \beta^n(\frac{\mathbf{x}}{h} - \mathbf{k}). \tag{23}$$

Here, $\beta^n(\cdot)$ is the uniform symmetric d-dimensional B-spline of degree n. The knots of the B-spline are located on a grid spanning Ω, with a regular spacing. The coefficients of the B-spline representation are gathered in $c[\mathbf{k}]$. h is a scale parameter which directly influences the degree of smoothing of the interface.

Energy criterion

$$E(\phi) = \int_\Omega F(I(\mathbf{x}), \phi(\mathbf{x}))\, d\mathbf{x}, \tag{24}$$

In the implementation proposed in CREASEG, F corresponds to the Chan&Vese data attachment term, given as

$$F(I(\mathbf{x}), \phi(\mathbf{x})) = H(\phi(\mathbf{x}))(I(\mathbf{x}) - v)^2 + (1 - H(\phi(\mathbf{x})))(I(\mathbf{x}) - u)^2. \tag{25}$$

In this expression, H is the Heaviside function, u and v are two parameters updated at each iteration according to equation (6) and (7).

Evolution equation

The minimization of the functional (24) can be done with respect to the B-spline coefficients $c[\mathbf{k}]$. The derivatives with respect to each B-spline coefficient $c[\mathbf{k_0}]$ may be expressed as

$$\frac{\partial E}{\partial c[\mathbf{k_0}]} = \int_\Omega \frac{\partial F(\mathbf{x}, \phi(\mathbf{x}))}{\partial \phi(\mathbf{x})} \cdot \beta^n(\frac{\mathbf{x}}{h} - \mathbf{k_0})\, d\mathbf{x}, \tag{26}$$

with

$$\frac{\partial F(\mathbf{x}, \phi(\mathbf{x}))}{\partial \phi(\mathbf{x})} = \delta(\phi(\mathbf{x})((I(\mathbf{x}) - v)^2 - (I(\mathbf{x}) - u)^2). \tag{27}$$

The level-set evolution may then be computed through a gradient descent on the B-spline coefficients. The corresponding variation of the B-spline coefficients is given as:

$$\mathbf{c}^{i+1} = \mathbf{c}^i - \lambda \nabla_c E(\mathbf{c}^i), \tag{28}$$

where λ is the iteration step and ∇_c corresponds to the gradient of the energy relative to the B-spline coefficients given by (26).

Properties

- This algorithm computes the level-set evolution on the whole image. So new contours could emerge far from the initialization.
- This algorithm is a region-based method and tries to separate the image into two homogeneous region (according to their mean value).

Specific parameters

This algorithm has one specific parameter that can be modified from the corresponding panel:

- the scale factor h: it allows to select the degree of smoothness of the evolving contour (default value is set to 1).

Note: h must be an integer and $h \leq 4$.

2.6 Shi [7]

Model

This method is a fast algorithm based on the approximation of the level-set based curve evolution. The implicit function is approximated by a piece-wise constant function taking only four values (-3, -1, 1, 3) corresponding respectively to the interior points, the interior points adjacent to the evolving curve, the exterior points adjacent to the evolving curve and the exterior points. The two narrow-bands that enclosed the evolving contours are gathered into two lists that are updated at each iteration from simple rules, making the algorithm particularly fast.

The curve evolution process is approximated in a two-cycle algorithm:

1. during Na iterations, the curve evolve using a data attachment term F_d.
2. then the curve is smoothed during Ns iterations; the regularization term F_r is computed for each point of L_{in} and L_{out} using a gaussian filter of variance σ and size Ng×Ng.

Evolution equation

In [7], several data attached terms are given. In the platform, we choose to use the Chan&Vese one given by:

$$F(I(\mathbf{x}), \phi(\mathbf{x})) = H(\phi(\mathbf{x}))(I(\mathbf{x}) - v)^2 + (1 - H(\phi(\mathbf{x})))(I(\mathbf{x}) - u)^2 \tag{29}$$

where H is the Heaviside function, u and v are two parameters updated at each iteration according to equation (6) and (7).

Properties

- This method is an approximation of level-set based curve evolution. It is a very fast method and evolves only on the narrow-band. However, this property is attenuated by the use of Matlab which does not offer optimized structures that deals with list management as in C++.

Specific parameters

This algorithm has four specific parameters that can be modified from the corresponding panel:

- Na: Number of iteration in the data dependent cycle (default value is set to 30).
- Ns: Number of iteration in the regularization cycle (default value is set to 3).
- σ: variance of the gaussian filter (default value is set to 3).
- Ng: size of the gaussian filter (default value is set to 1).

2.7 How to add your own algorithm

CREASEG has an additional panel that can be used to add your personal method, in order to compare it with the other implemented methods for validation. This section describes the procedure to follow to add your personal algorithm.

Default parameters

With the default settings of the panel *'Personal Algorithm'*, two parameters can be used. Changing their name or default value can be done at line 296 to 299 of the file 'creaseg-gui.m'. Changing the value of the String parameter on line 296 and 298 will change the name of the parameter while changing the value of the String parameter on line 297 and 299 will change the default value of the parameter.

Adding parameters

If your algorithm has more than two parameters, the procedure to add those parameters in the interface is:

1. In 'creaseg.m', add the position and size of the label and edit text for your parameters at the end of the array ud.personalAlgoConfig (line 145).
2. In 'creaseg_gui.m', add the following line (after line 299):
   - for a label: h7x = uicontrol('parent', h78, 'units', 'normalized', 'position', ud.personalAlgoConfig(yy,:), 'Style', 'text', 'String', 'Parameter Name', 'FontSize', 9, 'HorizontalAlignment', 'left', 'Backgroundcolor', [113/255 113/255 113/255]);
   - for an edit text: h7x = uicontrol('parent', h78, 'units', 'normalized', 'position', ud.personalAlgoConfig(yy,:), 'Style', 'edit', 'String', '1', 'BackgroundColor', [240/255 173/255 105/255]);
3. Add the component h7x to the ud.handleAlgoPersonal array (line 300 when 'creaseg_gui.m' has not be modified): ud.handleAlgoPersonal = [h71;...;h784; h7x];

Running the algorithm

Before being able to run your algorithm, you must uncomment the lines 60-61 and 181-186 of the file 'creaseg_run.m'. To be able to use your algorithm in comparison mode, you must also uncomment the lines 255-260 in the same file.

3 Comparison mode

CREASEG allows you to compare the different algorithms implemented, in order to choose the best suited algorithm for your application or to validate your own. The results of this comparison are automatically saved in a ".txt" file and displayed in a table in the interface (if your Matlab version allows it). This chapter explains the procedure to follow.

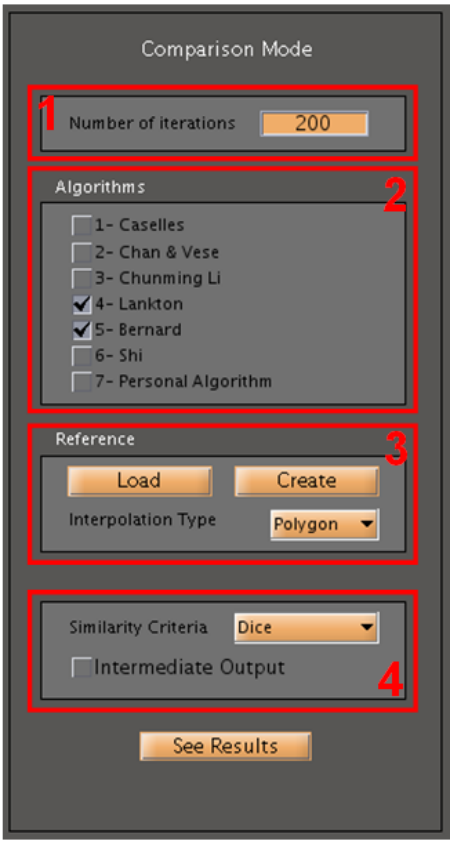


Fig. 9: *Comparison mode panel*

3.1 Comparison procedure

To select the comparison mode, you can either click on *'C- Comparison Mode'* in the *'Algorithms'* menu or click on the button *'C'* in the toolbar.

3.1.1 Choosing algorithms

When the comparison panel is displayed, choose the algorithms you want to compare using the check-boxes (fig.9 → area 2). You will notice in the toolbar that the icons of the selected algorithm have changed as well as their color in the *'Algorithms'* menu (fig.10). This allows you to easily remember which algorithm has been selected.

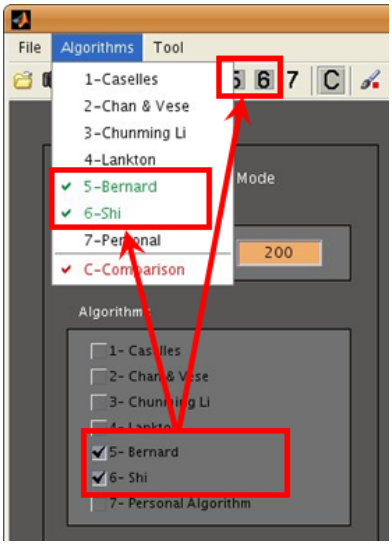


Fig. 10: *Selecting an algorithm for comparison will modify its color in the* 'Algorithms' *menu and its icon in the toolbar.*

As soon as an algorithm is selected in the comparison mode, if you click on its icon or if you select it in the *'Algorithms'* menu, its panel will be displayed but you won't quit the comparison mode. This allows you to modify the parameters of the algorithm.

Note: In comparison mode, all the algorithms will run during the same number of iterations. This explains why you can only modify the iteration number in the comparison panel (fig.9 → area 1).

3.1.2 Reference

In comparison mode, a similarity criterion is computed between the result and a reference mask. This reference can be either loaded from a data base or created using the same procedure as for creating an initial region.

Creating a reference To create a reference, click on the *'Create'* button in the comparison panel. The reference creation is done as follow:

- left click to draw a specific contour by adding points;
- right click to begin a new contour;
- middle click to save the reference.

Depending on the Matlab toolbox present on your computer, the reference mask can be modeled either with polygon or spline interpolation (fig.9 → area 3).

3.1.3 Comparison criteria

In comparison mode, you have access to three different types of comparison criterion:

1. visual criterion: This criterion allows you to plot the results of the selected algorithms on the image to compare them with the reference you have selected.
2. computation time
3. similarity criterion (fig.9 → area 4): Four similarity criteria can be computed between the result of the algorithms and the reference (see also fig.11 and table 1):
   - Dice criterion: $\text{Dice} = \frac{2(A \cap B)}{A+B}$, where A and B are the reference mask region and the result mask region of an algorithm;
   - PSNR: $PSNR = 10log_{10}\left(\frac{d}{MSE(A,B)}\right)$, where d is the maximum possible value of the image and $MSE(A,B)$ is the mean square error computed between A and B: $MSE(A,B) = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\|A(m,n) - B(m,n)\|^2$;
   - Hausdorff distance: $\text{Hausdorff} = max(D_1(A,B), D_1(B,A))$, where A and B are the reference contour and the result contour of an algorithm and $D_1(A,B) = \underset{\mathbf{x}\in A}{max}(\underset{\mathbf{y}\in B}{min}(\|\mathbf{x}-\mathbf{y}\|))$;
   - Mean Sum of Square Distance (MSSD): $MSSD = \frac{1}{N}\sum_{n=1}^{N} D_2^2(A, B(\mathbf{x}_n))$, where A and B are the reference contour and the result contour of an algorithm, N is the size of the result contour and $D_2(A, B(\mathbf{x})) = \underset{\mathbf{y}\in A}{min}(\|\mathbf{y}-\mathbf{x}\|)$;

Note 1: For computation of the Dice criterion and the PSNR, both the results and the reference are binary images (*i.e.* 1=inside, 0=outside).

Note 2: The default settings will display intermediate outputs, which can biased the computation time. In order to avoid this, you can disable the intermediate outputs by unchecking the corresponding check box in the *Comparison Mode* panel (fig.9-4).

3.2 Results

3.2.1 Results presentation

Once the comparison is finished, the *Results* panel is displayed. It contains a table in which the results are displayed (fig.12 → area 1) as well as check boxes that allow you to choose the contours you want to display on the image (fig.12 → area 2).

The results are also saved in a file named 'results.txt' in the *results* folder.

If an algorithm has not been selected for comparison, the computation time and similarity criterion will be set to 0.

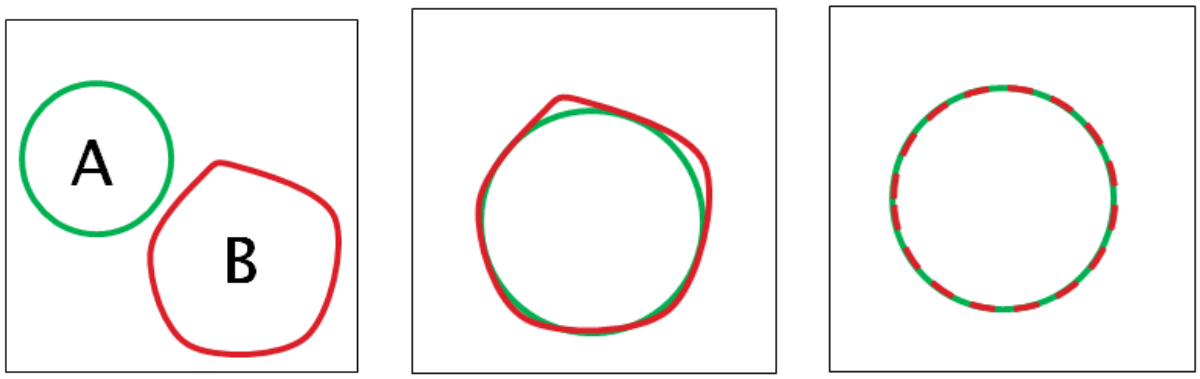
3.2.2 Results visualisation

At the end of a comparison, every contours (*i.e.* the reference and the result of each algorithms) are displayed. You can simply make disappear/appear a specific contour by unchecking/checking the corresponding checkboxes (fig.12 → area 2).

Note: Only the check boxes corresponding to an algorithm that was selected in the comparison panel are enable.

	$A \cap B = \{\phi\}$	$A \approx B$	$A = B$
Dice	0	≥ 0.9	1
PSNR	$0dB$	$\geq 20dB$	$+\infty$
Hausdorff	$+\infty$	≤ 15	0
MSSD	$+\infty$	≤ 5	0

Tab. 1: *Value of the similarity criteria for three different situations.*



(a) (b) (c)

Fig. 11: *Three different situations when computing the similarity criterion: (a) $A \cap B = \{\phi\}$; (b) $A \approx B$; (c) $A = B$.*

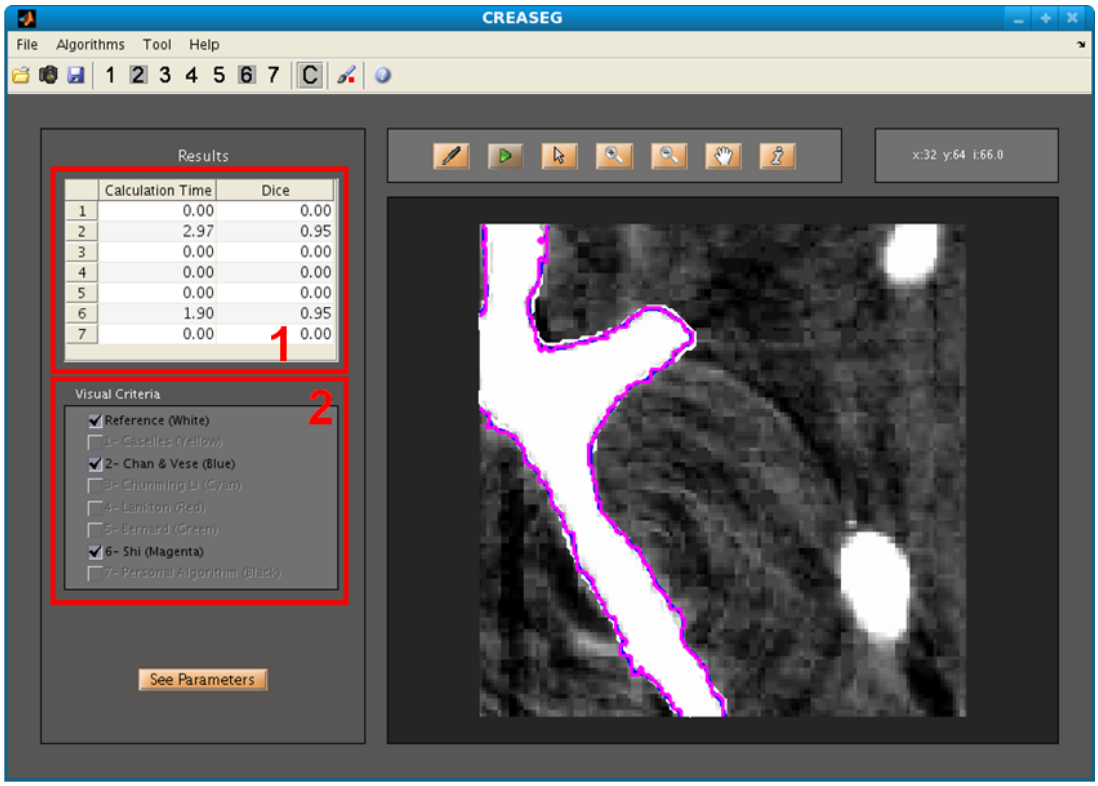


Fig. 12: *Presentation of the Results panel*

3.2.3 Saving results

In comparison mode, when you click on the *Save results* button or in the *File* menu, all the contours that are currently being displayed will be saved in an image file.

When you click on the *Save data* button or in the *File* menu, the corresponding ".mat" file will contain three fields:

- img: the image,
- levelset: a 3D-matrix of dimensions size(image,1)×size(image,2)×8. The first matrix (levelset(:,:,1)) is the reference, the following matrix being the results of the algorithms. If an algorithm has not been selected for comparison, its matrix will be filled with 0.
- method: an 1×8 array containing the name of the methods that correspond to each slice of the levelset matrix.

Appendix

.1 Image dataset

.1.1 Simulated images

- Leaf.tif: borrowed from [6],
- ThreeObj.bmp: borrowed from [4].

.1.2 Natural scenes

- Airplane.jpg: borrowed from `http://www.shawnlankton.com/`,
- Eagles.png, Mushroom.png, Statue.png, Tree.png, WaterCountry.png: borrowed from the Berkeley segmentation dataset and benchmark [8],
- Europe-Night-Lights.png, Spirale.png: borrowed from [3],
- Monkey.png: borrowed from [5].

.1.3 Medical images

- Arms_XRay.png: borrowed from `http://rsbweb.nih.gov/ij/index.html` (NIH),
- Brain_MRI.png: borrowed from `http://www.itk.org/` (Kitware),
- Heart_CT.pgm: image of a dog heart, acquired in CT, with a pixel size of 0.9 mm^2. The contrast between the LV cavity and the surrounding myocardium was enhanced through a Roentgen contrast agent injected prior to the scan,
- Heart_MRI.dcm was acquired at Hopital Cardiologique Louis Pradel HCL (Lyon, France),
- Heart_US1.png, Heart_US2.png were acquired using a Toshiba Powervision 6000 (Toshiba Medical Systems Europe, Zoetermeer, the Netherlands) equipped with an RF interface for research purposes and a 3.75 MHz-probe,
- Nucleus_FluorescenceMicrograph.gif: borrowed from [9],
- TwoCells_Microscopy.bmp: borrowed from [10],
- Vertex_MicroCT.pgm: Data from Synchrotron X-Ray CT of calcaneus bone (with a pixel size of 80 μm^2) are courtesy of Dr Francoise Peyrin from Creatis at Lyon and ESRF (European Synchrotron Radiation Facility) at Grenoble, France,
- Vessel_CTA1.bmp, Vessel_XRay2.bmp: borrowed from [4],
- Yeast_FluorescenceMicrograph.png: borrowed from [6].

Bibliography

[1] T. Dietenbeck, M. Alessandrini, D. Friboulet, and O. Bernard, "Creaseg: a free software for the evaluation of image segmentation algorithms based on level-set," in *IEEE International Conference On Image Processing. Hong Kong, China*, 2010.

[2] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *Int. J. of Computer Vision*, vol. 22, pp. 61–79, 1997.

[3] T. Chan and L. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, pp. 266–277, February 2001.

[4] C. Li, C.-Y. Kao, J. C. Gore, and Z. Ding, "Minimization of region-scalable fitting energy for image segmentation," *IEEE Trans. Image Process.*, vol. 17, pp. 1940–1949, 2008.

[5] S. Lankton and A. Tannenbaum, "Localizing region-based active contours," *IEEE Trans. Image Process.*, vol. 17, pp. 2029–2039, November 2008.

[6] O. Bernard, D. Friboulet, P. Thevenaz, and M. Unser, "Variational B-Spline Level-Set: A Linear Filtering Approach for Fast Deformable Model Evolution," *IEEE Trans. Image Process.*, vol. 18, pp. 1179–1191, June 2009.

[7] Y. Shi and W. C. Karl, "A real-time algorithm for the approximation of level-set based curve evolution," *IEEE Trans. Image Process.*, vol. 17, pp. 645–656, May 2008.

[8] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th Int'l Conf. Computer Vision*, July 2001, vol. 2, pp. 416–423.

[9] O. Bernard, D. Friboulet, P. Thevenaz, and M. Unser, "Variational B-spline level-set method for fast image segmentation," in *IEEE International Symposium on Biomedical Imaging (ISBI), Paris, France*, 2008, pp. 177–180.

[10] C. Li, C. Xu, C. Gui, and M. D. Fox, "Level set evolution without re-initialization: A new variational formulation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2005, vol. 1, pp. 430–436.