IPOL
IMAGE PROCESSING ON LINE

Published in Image Processing On Line on 2012–03–23.
Submitted on 2012–00–00, accepted on 2012–00–00.
ISSN 2105–1232 © 2012 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
http://dx.doi.org/10.5201/ipol.2012.abmh-rtmsa

A Real Time Morphological Snakes Algorithm

Luis Alvarez[1], Luis Baumela[2], Pablo Márquez-Neila[3], Pedro Henríquez[4]

[1]Universidad de Las Palmas de Gran Canaria, Spain (lalvarez@dis.ulpgc.es)
[2]Universidad Politécnica de Madrid, Spain (lbaumela@fi.upm.es)
[3]Universidad de Las Palmas de Gran Canaria, Spain (phenriquez@ctim.es)
[4]Universidad Politécnica de Madrid, Spain (p.mneila@upm.es)

Abstract

Active contours, or snakes, are computer-generated curves that move within images to find out salient image structures like object boundaries. Energy based formulations using a level set approach have been successfully used to model the snake evolution. The Euler-Lagrange equation associated to such energies yields to partial differential equations (PDE) which are usually solved using level set methods which involve contour distance function estimation and standard methods to discretize the PDE. Recently we have proposed a morphological approach to snake evolution. First, we observe that the differential operators used in the standard PDE snake models can be approached using morphological operations. By combining the morphological operators associated to the PDE components we achieve a new morphological approach to the PDE snakes evolution. This new approach is based on numerical methods which are very simple and fast. Moreover, since the level set is just a binary piecewise constant function, this approach does not require to estimate a contour distance function to define the level set. In this paper we present an algorithm to compute morphological snake evolution in real time.

Source Code

A basic source code where only BMP image format is used is available in the article web page (http://dx.doi.org/10.5201/ipol.2012.abmh-rtmsa). This is the reviewed version of the software.

Source code of the version which is able to load more image formats (JPEG, PNG, TIFF, BMP) is also available in the article web page.

Keywords: snakes

1 Overview

1.1 Short Presentation

Morphological snakes [1] are based on a morphological discretization of the PDE model proposed in a previous article [2] and given by the equation:

$$\frac{\partial u}{\partial t} = g(I)|\nabla u| \left(\operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) + \nu \right) + \nabla g(I) \nabla u \tag{1}$$

Luis Alvarez, Luis Baumela, Pablo Márquez-Neila, Pedro Henríquez, *A Real Time Morphological Snakes Algorithm*, Image Processing On Line, 2 (2012), pp. 1–7. http://dx.doi.org/10.5201/ipol.2012.abmh-rtmsa

We will approach the solution of the above equation by decomposing it in 3 terms : (i) the balloon force term, (ii) the smoothing term and (iii) the attraction force term. Next we will analyze these terms separately.

1.1.1 The Balloon Force Term

The balloon force operator term is given by the equation

$$\frac{\partial u}{\partial t} = g(I)\nu|\nabla u|. \tag{2}$$

The above equation is strongly related to the classical erosion and dilation morphological operators. In fact, according with the sign of ν the solution of the above equation is equivalent to an erosion or dilation on the level set images and can be approached using the numerical scheme

$$u^{n+1}(x) = \begin{cases} D_d u^n(x) & \text{if } g(I)(x) \geq t_1 \text{ and } \nu > 0 \\ E_d u^n(x) & \text{if } g(I)(x) \geq t_1 \text{ and } \nu < 0 \\ u^n(x) & \text{otherwise} \end{cases} \tag{3}$$

where D_d and E_d are the discrete versions of dilation and erosion. A disk with radius 1 and formed with eight neighbor pixels is used as a structure element. Erosions and dilations are implemented in a very simple way by iterations of 8 or 5 neighborhood maxima (or minima) computation. In the demo is used the 5 neighbors version, and at the boundaries we use an homogeneous Neumann type boundary condition. Another useful option to make this balloon type operator evolve is to use an image interval value, that is :

$$u^{n+1}(x) = \begin{cases} D_d u^n(x) & \text{if } I(x) \in [I_0, I_1] \text{ and } \nu > 0 \\ E_d u^n(x) & \text{if } I(x) \in [I_0, I_1] \text{ and } \nu < 0 \\ u^n(x) & \text{otherwise} \end{cases} \tag{4}$$

1.1.2 The Smoothing Term

Next we consider the regularization term given by the equation

$$\frac{\partial u}{\partial t} = g(I)|\nabla u| \left(\operatorname{div} \left(\frac{\nabla u}{|\nabla u|} \right) \right). \tag{5}$$

The solution of the above PDE can be approached using line morphological operators. Let B the set of all line segments of length 2 centered at the origin. We define the morphological line operators as

$$\begin{array}{rcl} (SI_d u)(x) & = & \sup_{S\in B} \inf_{y\in x+hS} u(y) \\ (IS_d u)(x) & = & \inf_{S\in B} \sup_{y\in x+hS} u(y) \end{array}$$

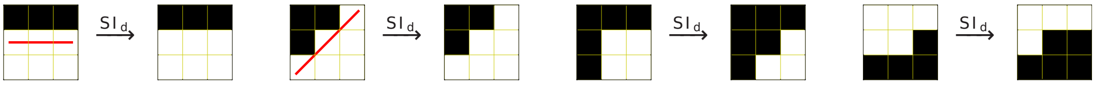
Using these line morphological operators the solution of the above PDE can be approached using the numerical scheme (approximates mean curvature motion)

$$u^{n+1}(x) = \begin{cases} (SI_d \circ IS_d u^n)(x) & \text{if } \quad g(I)(x) \geq t_2 \\ u^n(x) & \text{otherwise} \end{cases} \tag{6}$$

where SI_d and IS_d are discrete versions of the above morphological continuous line operators. Both SI_d and IS_d have their own version of the set $\mathcal{B}, \mathcal{P}$, which is a collection of four discretized segments centered at the origin:

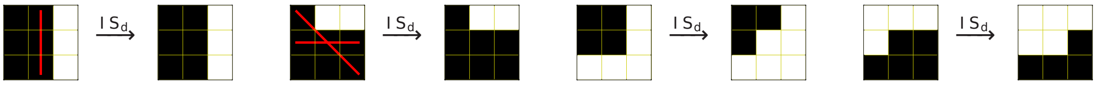
$$\mathcal{P} = \begin{pmatrix} P_0 & = & \{(0,0),(1,0),(-1,0)\} \\ P_1 & = & \{(0,0),(0,1),(0,-1)\} \\ P_2 & = & \{(0,0),(1,1),(-1,-1)\} \\ P_3 & = & \{(0,0),(1,-1),(-1,1)\} \end{pmatrix}$$

Below there are some examples of the effect of the SI_d and IS_d operator on individual pixels of binary images. In those cases where a straight line is found (marked in red), the central pixel remains active (cases a and b). When the central pixel does not belong to a straight line of active pixels, it is made inactive (cases c and d). For exemplification purposes, we assume the pixels on the borders are not affected by the operator.



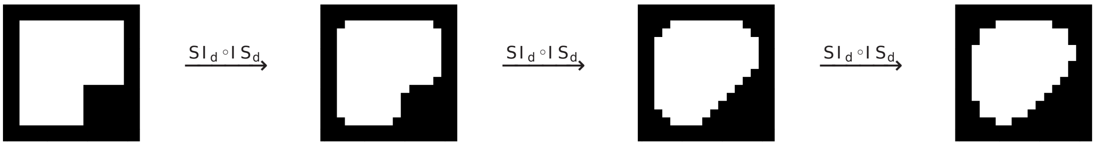
(a) (b) (c) (d)

Examples of the SI_h operator:



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

Example of the $SI_h \circ IS_h$ operator iterated until convergence:



1.1.3 The Attraction Force Term

The attraction force PDE term is given by the equation

$$\frac{\partial u}{\partial t} = \nabla g(I) \nabla u \tag{7}$$

The morphological discretization of this PDE term is very simple and given by

$$u^{n+1}(x) = \begin{cases} 1 & \text{if } \nabla u^n(x) \nabla g(I)(x) > 0 \text{ and } g(I)(x) \geq t_3 \\ 0 & \text{if } \nabla u^n(x) \nabla g(I)(x) < 0 \text{ and } g(I)(x) \geq t_3 \\ u^n(x) & \text{otherwise} \end{cases} \tag{8}$$

Gradient is implemented using basic finite difference schemes, we use central finite difference scheme. We observe that the image level set $u(x)$ we use is a binary function taking values in $\{0,1\}$ and the snake is given by the level set contour.

2 Algorithm

To design the algorithm for the snake evolution we take into account that the solution of the snake evolution PDE can be approached by composition of the solution of the 3 terms the original equation has been decomposed, so the algorithm is very simple, it is just iterations of the composition of the numerical schemes (3) or (4), (6) and (8). One important observation is that these 3 numerical schemes are morphological in the sense that they do not create extra level set values, that is if the input level set is a binary image, the output level set is also a binary image.

2.1 Parameter Selection

Snake initialization. The snake is initialized using a polygon provided by the user. From this polygon we build a binary level set image $u(x)$ assigning the value 1 inside the contour and 0 outside the contour.

Attraction force *g(I)*. According with the application requirements we can choose between

$$\begin{array}{ll} g(I)=\dfrac{1}{\sqrt{10^{-20}+|\nabla G_s * I|}} & \text{Edge border attractor} \\ g(I)=|G_s * I| & \text{Dark center line attractor} \\ g(I)=1 & \text{No attractor defined} \end{array}$$

where G_s is the Gaussian function with standard deviation s.

Choice of t_1, t_2 and t_3 in expressions (3), (5) and (7). To simplify the parameter choice t_2 and t_3 are fitted to 0, that is regularization and snake attractor are always applied. t_1 (when applied) is estimated using a percentage of $g(I)$ histogram values. That is, the user provided a number $0 \leq p \leq 1$ and we take t_1 such that

$$\frac{|\{x \in \Omega : g(I)(x) \geq t_1\}|}{|\Omega|} = p \tag{9}$$

where Ω is the image domain. In the case balloon option (4) is used, we use as parameter the radius of interval $[I_0, I_1]$. The interval center is obtained using the median of the image value in a contour neighborhood region.

Maximum Number of iterations Since we deal with an iterative procedure user also provides the maximum number of iterations.

2.2 Algorithm Output

Contour collection The main algorithm output is a contour collection with the contour points for all algorithm iterations.

Illustrative image To illustrate the results we draw in the original image the contour for some algorithm iterations.

3 Implementation Details

As explained above, the snake evolution is performed by iterations of the composition of the numerical schemes (3) or (4), (6) and (8). To speed up the algorithm we store in memory the binary level set snake representation and a contour representation where only the snake contour points are stored. Using this representation numerical scheme iterations are performed in the following way:

step 1 : Initialization of contour points and binary snake level set from the contour provided by the user

step 2 : In each iteration of operators (3) (4), (6) or (8):

- For each point of the contour we compute the operator in a 3x3 neigbourhood of the point to update the binary snake level set.
- We update the contour using the new updated binary snake level set obtained above

Using this hybrid scheme where we manage the contour representation of the snake and the associated level set we obtain that at each iteration, only a 3×3 neigbourhood of the contour is considered for computation.

We observe that the snake evolution is always performed in the level set representation, never in the contour representation. On the other hand contour update from snake level set function is very easy : x is part of the contour line if $u(x) = 1$ and at least one of its 4 neighbors is equal to 0. For snake evolution purpose we do not need any special contour organization. Contour is a non-ordered collection of 2D points.

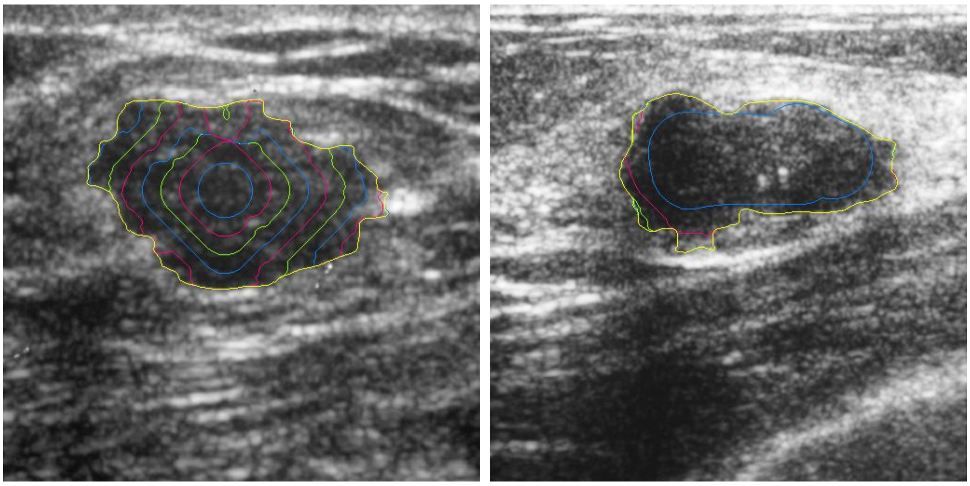
The linear differential operators we use, as the gradient, are implemented using simple centered finite difference scheme.

We use always homogeneous Neumann type boundary conditions in operator implementation.

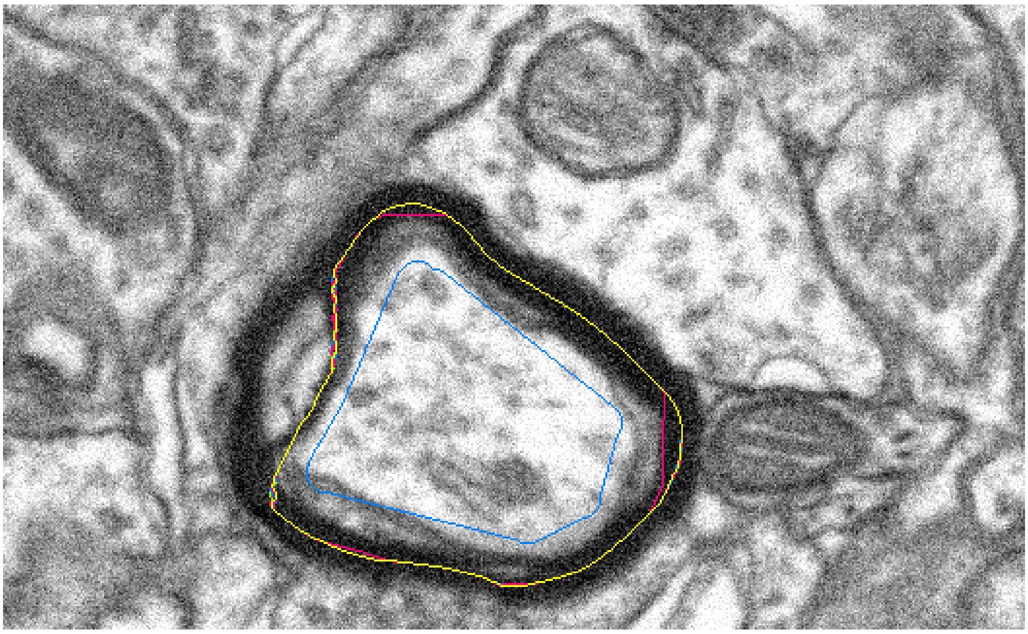
4 Examples

We present some examples to illustrate the performance of snake algorithm.

First, we present 2 medical images. The initial snake is in blue inside the object. Using different colors we draw different iterations of snake algorithm.



Next we present the results of snake algorithm in the case of a electronic microscopy image.

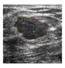
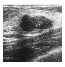


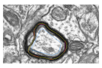
5 Online Demo

An on-line demo of the algorithm is available on the article web page (`http://dx.doi.org/10.5201/ipol.2012.abmh-rtmsa`). With this demo, one can:

- Upload any image and segment an image structure.
- Fit the main snake evolution parameters which are (we strongly simplify the parameter choice):
  - The interest structure type. We have 2 options: `HIGH_CONTRAST_REGION` (a contrasted region in the image) or `DARK_CENTER_LINES` (center lines of dark band structures).
  - The snake balloon force type. We have 3 options: `NO_BALLOON_FORCE` (no ballon force is used), `EXPANSION_BALLOON_FORCE` (an expansion ballon force) or `CONTRACTION_BALLOON_FORCE` (a contraction balloon force).
  - The number of iterations.
  - The standard deviation of the Gaussian to fit the scale to look for interest structures. This parameter is used in $g(I)$ snake operator function. The mask size is showed with a circle to give the user more information about the changes produced by this parameter.
- Define the initial snake by clicking in the image.
- Run the snake algorithm. The output of snake algorithm is an ASCII file with the final contour and an illustrative image with some iterations of the snake algorithm.
- Run the snake algorithm using a previous output as new input.

Image Credits

  courtesy Servicio de Radiodiagnóstico del Hospital Universitario Insular de Gran Canaria, CC-BY

 courtesy Center for Biomedical Technology, Universidad Politécnica de Madrid, CC-BY

References

[1] M. Aleman-Flores, L. Alvarez and P. Henriquez, “AmiLabContours: A tool for image structure segmentation,” IADIS Multi Conference on Computer Science and Information Systems, Freiburg, Germany, 2010. http://ami.dis.ulpgc.es/ami/amilabcontours.html

[2] L. Alvarez, L. Baumela, P. Marquez-Neila and P. Henriquez, “Morphological Snakes,” CVPR 2010, San Francisco, U.S.A. http://www.sciweavers.org/publications/morphological-snakes

[3] V. Caselles, R. Kimmel, and G. Sapiro, “Geodesic active contours,” International Journal of Computer Vision, vol. 22, no. 1, pp. 61–79, 1997. http://dx.doi.org/10.1023/A:1007979827043

[4] F. Catté, F. Dibos, G. Koepfler, “A morphological scheme for mean curvature motion and applications to anisotropic diffusion and motion of level sets,” SIAM Journal on Numerical Analysis, vol. 32, no. 6, pp. 1895–1909, 1995. http://dx.doi.org/10.1137/0732085

[5] F. Guichard, J-M. Morel, R. Ryan, “Contrast invariant image analysis and PDE’s.” http://mw.cmla.ens-cachan.fr/~morel/JMMBookOct04.pdf.