

# Active Contours Using a Potential Field

David M. Honea, Wesley E. Snyder, and Griff Bilbro

Image Analysis Lab, Dept. of Electrical & Computer Engineering, North Carolina State University, Raleigh, NC

## Abstract

*In most implementations of active contours (snakes), the evolution of the snake depends only on image characteristics in the immediate neighborhood of the current snake points. This is true even when there is little edge data available in the current neighborhood, and even when the boundary of interest may be some distance away in the image. This paper proposes a vector potential field at each point in the image that is derived from the "pull" exerted by all edge points in the image; the pull for a given edge is inversely proportional to the square of the distance from the pixel it pulls. This potential field acts as a force, and snake points are moved based on the force at their current location, rather than moving to minimize energy at a candidate position. The resulting algorithm allows edges to influence snake evolution earlier and from a greater distance, and results in faster and better convergence to the final boundary under a variety of image characteristics.*

## 1. Introduction

Many image analysis operations require finding a closed contour, and simple edge detection methods often fail in such applications because portions of the boundary are obscured by noise or blurred. In such applications, the concept of Adaptive Contours becomes useful. Adaptive Contours proposes an initial contour and then modifies that contour until it conforms with measured edge data. In addition to preferring areas of high "edginess", adaptive contours may also impose smoothness constraints.

Algorithms based on adaptive contours have often been referred to as "snakes" because of the way they evolve. In this paper, we will use the term "snake" for a two-dimensional adaptive contour.

In this work, we improve both the performance and speed of conventional snakes by making use of an observation: rather than simply moving towards or away from the center or moving normal to the current boundary, it makes more sense for snake points to be attracted to edges in the image, even if they are some distance away.

This development has been made in reaction to a problem with conventional implementations of snakes. A conventional snake algorithm attempts to minimize some energy function, typically a sum of an "internal energy" which describes the smoothness of the snake, and an "external energy" which describes how well the contour aligns with edge points in the image. The problem with conventional techniques is that the domain of the search is likely to be restricted. Some algorithms restrict the movement of

the snake points to only the nearest point. Others provide a band of finite, predetermined size about the present boundary in which to search for new snake point. Given this, if a snake point exists within a large homogeneous region, image data does not provide any guidance as to what the appropriate direction should be.

Such algorithms suffer from the same problem that was described as a "Parzen window"[1] in the pattern recognition literature — a point inside the search boundary is fully considered, a point outside that boundary is ignored. What we need is an algorithm which will consider an edge point as attracting a snake point, no matter how far away it is.

To some extent, this problem may be addressed by introducing inflation or deflation forces, shape constraints, or other strategies which influence motion but are unrelated to image information. A preferable technique would be one which allows edges to influence snake evolution from a distance.

The technique proposed in this paper corrects this problem by precomputing a potential field which influences the motion of each snake point, no matter where that point is located. We will first describe the simplest implementation of this concept, and then enhance it to improve performance in situations with different geometries.

## 2. Background

Kass *et al.* [2] proposed the concept of snakes as a method of combining image information and shape information as a means of guiding the evolution of a closed boundary. They formulated this as an energy minimization problem in

which the edge information from the image is contained in an “external” energy term, and some measures of smoothness are included in the “internal” energy. Thus, minimizing the total energy produces a contour which is closed, smooth, and close to features of interest in the image such as edges. Many variations to the concept of adaptive contours have been proposed, including different expressions for the energies. See [3] for an excellent survey of the field.

Two fundamentally different strategies exist regarding motion and initialization of adaptive contours — those that initialize outside the true contour and shrink [4], and those that initialize inside the true contour and expand. In either case, the typical strategy has been to design the smoothness term so that the shape energy has a natural bias which drives the snake in the desired direction until it reaches a location where the external energies dominate. In this paper, we follow an expansion philosophy, in which we assume that the interior of the region is relatively homogeneous.

Either of these cases creates a dependence on the initialization because as the contour moves, it may pass through a local minimum of the energy function. The original snake algorithms and most subsequent ones find the minimum by simple descent and therefore stop when a local minimum is reached. In these cases, the algorithm designer attempts to avoid local minima by making the convergence space around the correct boundary as large as possible, by creating energy functions which do not have local minima near the desired boundary. The user is tasked with finding an initialization which is close enough to the desired minimum that the algorithm will converge to that minimum. Even techniques such as simulated annealing, which can avoid incorrect local minima [5], cannot do so unless the search area includes the global minimum. Ensuring the global minimum is included in the search may prohibitively increase computational complexity.

The adaptive contour philosophy can be modified from an energy minimization philosophy to one which models the motion of the contour as describable by partial differential equation. The most popular current implementations of this concept characterize the contour of the region as the zero level set of some higher dimensionality function which is modified over the course of the algorithm [6,7].

This paper describes an extension of the adaptive contours concept which allows influence of image features in a global rather than a purely local sense, by computing a global vector force field. For clarity, we describe this concept in terms of the original “snakes” formulation with a discrete number of snake points; however the vector field concept is applicable to level set implementations.

### 3. Approach

Let  $X = \{\vec{x}_i\}$   $i = 1, \dots, N$  denote a set of sites, consisting of all the locations in the image, and let  $S = \{\vec{s}_j\}$   $j = 1, \dots, m$  denote the set of pixels at which edges exist. Let  $e(s_j)$  denote the edge strength at  $\vec{s}_j$ , and similarly, let  $f(\vec{x}_i)$  denote the brightness at image point  $\vec{x}_i$ . We can compute the Euclidean (or other metric) distance from  $\vec{x}_i$  to  $\vec{s}_j$  and denote that distance by  $d_{ij}$ . We would then model the attraction of edge point  $\vec{s}_j$  on a snake point  $\vec{x}_i$  as a gravitational field

$$E_{ij} = \alpha \frac{e(s_j)}{d_{ij}^2} \quad (1)$$

A snake point at  $\vec{x}_i$  then feels a net force vector from the edges of

$$\vec{E}_i = \sum_{j=1}^m E_{ij} \frac{\vec{s}_i - \vec{x}_j}{|\vec{s}_i - \vec{x}_j|} \quad (2)$$

Figure 1 illustrates an original image, its edge map, and the horizontal component of this potential field. Figure 2 il-

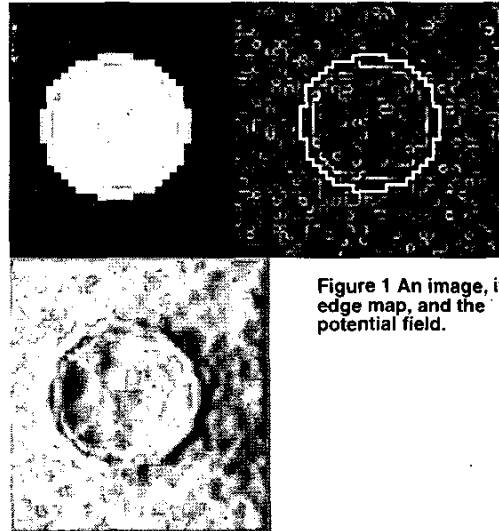


Figure 1 An image, its edge map, and the potential field.

lustrates the field in a CT cross section of a phantom constructed by suspending a grape in gelatin.

This method requires that an edge map be computed showing the edge strength at every point in the image. To reduce processing time during segmentation, the potential

field at each pixel can be precomputed. This field will have both a magnitude and direction at every point.

The fundamental concept of allowing edges to influence external energy from a distance can be included in any model for snakes, so decisions about internal energy and energy minimization techniques can be made based on properties that have been widely discussed in the literature. We allow force from the field to determine movement directly, rather than computing an energy for several candidate snake points and moving to the “best” point. Because our external “energy” is now a force vector, we also need to formulate the internal (shape-based) energies as force vectors.

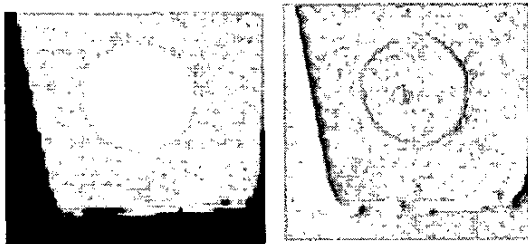


Figure 2 (left) CT section through a grape suspended in gelatin. (right) horizontal field.

### 3.1 . Internal forces

To incorporate smoothness constraints (internal energy) we add two forces based on the shape of the contour. The first pushes each snake point away from all other snake points. This prevents multiple snake points from converging to a single strong edge. This force is inversely proportional to the distance between points. If we let  $c_i$  denote the snake point at  $x_i$  and let  $c_{ix}$  be the  $x$ -coordinate of  $c_i$ , then the internal force at  $c_i$  resulting from repulsion is:

$$\vec{R}_i = \left( \sum_{j \neq i} \frac{1}{c_{ix} - c_{jx}}, \sum_{j \neq i} \frac{1}{c_{iy} - c_{jy}} \right) \quad (3)$$

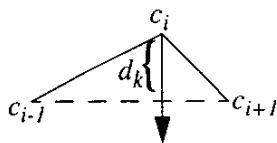


Figure 3 Force vector which encourages smoothness is perpendicular to the line between neighboring snake points.

The second internal force acts to minimize curvature, by pulling a point towards the line between its two neighbors. Let  $c_{i-1}$ ,  $c_i$ , and  $c_{i+1}$  be three consecutive snake points, as shown in Figure 3. Let  $\vec{R}$  be the unit vector normal to  $(c_{i-1}, c_{i+1})$  as illus-

trated, with magnitude proportional to the distance  $d_k$ . When added to the edge field strength and the repulsive force, this gives us a total force acting on a point of:

$$\vec{F}_i = w_e \vec{E}_i + w_1 \vec{R}_i + w_2 d_k \vec{R} \quad (4)$$

where the  $w$ 's are scale factors.

In reaction to this force, the snake point may respond in a variety of ways, depending on the dynamics defined for this system — it may accelerate, respond with an instantaneous velocity equal to the magnitude of the field, *etc.*, with the velocity as a function of the force. Using a dynamic model which includes inertia and “damping” results in the following update rule for the velocity of a snake point at iteration  $k$ :

$$v^{k+1} = v^k + \frac{\vec{F}^{k+1} - \alpha v^k}{m}, \quad (5)$$

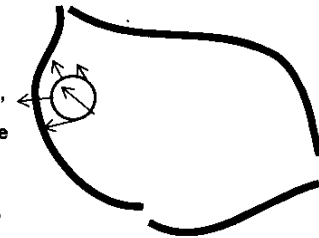
where we have dropped the  $i$  subscript for notational convenience. In eq.5  $m$  and  $\alpha$  are weights analogous to mass and damping. Using an inertia-free model results in making the velocity proportional to the force.

### 4. Refinements

The method we have described effectively achieves our goal of allowing edge points to exert influence on the behavior of a snake at distances greater than the typical search radius. However, it introduces a new problem: snake points can be drawn by any edges, including those that lie on the opposite side of the currently segmented region. If the snake is initialized in an off-center position, this can create a situation where every point on the snake is pulled towards the same side of the object.

Figure 4 illustrates the need for a refinement. In that

Figure 4 Edge points in the image are illustrated by dark lines. The initial location of the snake, illustrated by the circle, is chosen to be close to one of the edges; so close that the potential field, illustrated by arrows, on both sides of the snake contour pull the snake toward the same edge.



figure, the potential field on both sides of the snake is dominated by the left edge, and therefore, the entire snake, both sides, are pulled toward the same edge.

We correct this with the following heuristic: if an edge is on the other side of the snake, it does not affect the field. Of course, determining what “on the other side of the snake” means for arbitrary, nonconvex snakes is a significant problem itself. We solve it by ray tracing:

Given a snake point  $\zeta$ , and an edge point,  $\xi$ , calculate the vector  $\zeta - \xi$ . Trace along that vector from  $\zeta$  toward  $\xi$ , and set a flag, INTERIOR to zero. Let  $\hat{x}$  denote a point along that vector. Let  $\Psi(\hat{x})$  be a function which indicates if  $\hat{x}$  is at a point which is in the current interior of the snake. Perform the following two tests:

if ( $\Psi(\hat{x}) = \text{TRUE}$ ) then set INTERIOR to 1.

if ( $\Psi(\hat{x}) = \text{FALSE}$ ) AND INTERIOR = 0), (the ray tracing has been inside the snake and has now left) then break out of the ray tracing and ignore this edge point.

The effect of this process is to allow edge points to influence a snake point so long as the ray from the snake point to the edge point does not pass into and back out of the interior of the snake contour.

This refinement potentially increases computational complexity. It is now necessary to keep track of  $\Psi(\hat{x})$ , the function which maps pixels in the interior of the snake. The potential field at any point is dependent on the current geometry of the snake, so it is no longer possible to precompute the field for the entire image. If the snake requires a large number of iterations to converge to the appropriate boundary, the fact that field values are recomputed each iteration becomes significant. Computing the potential field at an individual pixel is also more complex, since for every edge point a ray trace is required to determine if that edge lies on the opposite side of the snake from the pixel in question. However, this increase in complexity is offset by the fact that we only compute the field each iteration at a relatively small number of snake points, rather than for every pixel in the image. Let the number of snake points be  $n$ , the number of iterations required for convergence be  $i$ , and the number of pixels in the image be  $N$ . This modification to the algorithm does not increase computation time until

$$ni > \frac{N}{\beta} \quad (6)$$

where  $\beta$  is a number determined by the complexity of tracing the ray between the snake points and the edge points.

## 5. Preliminary Results

As of this writing, the original and refined forms of the potential-field snake have both been implemented on synthetic and real data, including examples with non-convex boundaries, using the inertia-free dynamic model. The initialization for the algorithm was a single user-defined point about which a circular initial contour was generated.

The original version performs very well when initialized near center of the object of interest. As expected, it fails when initialized too close to one edge of the object. The algorithm converges to the correct boundary in many fewer

iterations than a traditional snake. Also, the traditional snake needs to have an inflation bias in the smoothness terms to allow it to find edges when initialized in a large homogeneous region, while the potential-field snake, when initialized in the center of the object, can move into the neighborhood of the boundary with just the influence of the image-derived term.

The refined version of the potential-field snake is able to find the boundary even in the case of a very off-center initialization, as illustrated in Figure 4. This version is actually faster than precomputing the field for the entire image for the number of iterations required to find the boundary.

To date all experiments have been conducted using the inertia-free dynamic model. Because the zero inertia model was used, the snake obtains the boundary easily but oscillates about that boundary. An animation of the snake evolution under different operating conditions will be presented at the conference.

## 6. Future Work

We are considering different dynamic models - combinations of snake point velocity, acceleration, and damping effects - on the evolution and ultimate solution provided by the potential-field snake. We are also studying the effect of varying the weighting given to each term in Eq. 4.

A thorough evaluation of sensitivity to noise, boundary geometry, and occlusions is under way, and results will be presented at the conference. This will include results from real medical imaging data.

Extensions to three dimensions have been implemented but are not described in this paper.

## 7. References

- [1] Parzen, E., *Modern Probability Theory and Its Applications*, Wiley & Sons, New York, 1992.
- [2] Kass, M., A.Witkin, and D.Terzopoulos, "Snakes: Active Contour Models," *Int'l. J. of Computer Vision*, 1987, 1:321-331.
- [3] Xu, C., D.L.Pham, and J.L.Prince, "Image Segmentation Using Deformable Models" in *Handbook of Medical Imaging, Vol.2, Medical Image Processing and Analysis*, eds. M.Sonka and J.M.Fitzpatrick, SPIE Press, 2000, p.129-174.
- [4] Honea, D.M., Y.Ge, W.E.Snyder, P.Hemler, and D.J.Vining, "Lymph-Node Segmentation Using Active Contours," *Medical Imaging 1997 - SPIE 3034*, Newport, 265-273.
- [5] Honea, D.M, and W.E. Snyder, "Three-Dimensional Active Surface Approach to Lymph Node Segmentation", *Medical Imaging 1999 - SPIE 3661*, San Diego.
- [6] Malladi, R., J.Sethian, and B.Vemuri, "Shape Modeling with Front Propagation: A Level Set Approach," *IEEE Trans. on Pattern Analysis & Machine Intell.*, 1995, 17(2), 158-175.
- [7] Yezzi,A., S.Kidenassamy, P.Olver, A.Tannebaum, "A Geometric Snake Model for Segmentation of Medical Imagery," *IEEE Trans. on Med.Imaging*, 1997, 16(2): 199-210.