Adaptive Image Processing via Snake Filters

Nilanjan Ray and Scott T. Acton

Virginia Image and Video Analysis (VIVA)
Department of Electrical and Computer Engineering,
University of Virginia, Charlottesville, VA 22904.
{nray, acton}@virginia.edu

Abstract

We propose an image enhancement and scaling technique using adaptive snakes that modify the image according to region area and perimeter. Snake filters consist of geometric active contours that eliminate image regions below the area scale and above the specified perimeter. The snake filters are efficiently implemented using front propagation via a fast marching method. We compare the snake filters to similar filters based on connected component analysis and area morphology. The paper demonstrates that snake filters improve upon the computational cost and the flexibility of area morphological filters such as the area open-close filter. Applications of object detection based on area and perimeter scales are given for biomedical images.

1. Introduction

A significant development in image morphology is the introduction of *area morphology* [1,2] or *flat zone filtering* [3]. Within a binary image, *area open* removes all the connected components having area less than a given size. The *area close* performs the same area-based operation on the complement of the connected components (on the zeros instead of the ones). For a gray scale image having N gray levels, the area open or area close operation first forms a stack of N binary images by thresholding the image at gray levels i=1 to N-1. These binary images are called *level sets* of the image. Area open (area close) is then applied to each of these N-1 level sets to form another stack of level sets, and finally a stacking operation is used to reconstruct the gray level image. Such a straightforward implementation is highly computationally intensive. A considerable amount of effort has already been generated toward designing efficient algorithms of area open or area close for gray level imagery [4,5].

In this paper, we reformulate the area open and area close operation on a gray level image as an active contour or snake propagation technique [6]. The snake filter, using the fast marching method, allows efficient filtering of the image further based on the area and the perimeter of the connected components. The proposed filter modifies the image by adaptively growing a front that encompasses connected components in the image level sets. As the front grows, components are eliminated that do not meet the prescribed area/perimeter criteria. So, the proposed filter represents a superset of filters containing the area open and area close operations. In fact, any filtering involving area scale as well as the perimeter scale of the connected components of a level set can be conceivable with the proposed snake filter. One such specific filtering is demonstrated here where we restrict the connected components of the level sets to be above an area threshold but below a perimeter threshold. Important applications of the filtering on biomedical images are illustrated.

2. Front propagation by fast marching method

A *front* may be best defined by an example – the forest fire movement. In such a case, the front is the interface between the untouched regions and the burned portions of the forest. The front does not move "backwards." So, in terms of the forest fire, one may say that once a portion of the forest burns, it remains burnt, and cannot be ignited again. In such cases one can characterize the moving front location by the arrival time $T(x,y)$ of the front as it crosses each point (x,y) of the plane. We know that speed is the ratio of distance traversed and the time elapsed. So, if $F(x,y)$ is the speed of the front at location (x,y)

0-7803-7147-X/01/$10.00©2001 IEEE

then the front propagation phenomenon can be formulated as a boundary value problem [6]:

$$|\nabla T(x,y)|F(x,y)=1 \text{ and } T(x,y)=0 \text{ when } (x,y)\in \partial D \quad (1)$$

where ∂D is the initial front. When the speed only depends on the space variables (x,y), (1) is known as Eikonal [6]. Sethian [6] has proposed a fast algorithm for such front propagation in discrete grid. The algorithm, known as the *fast marching method*, is described in the next section in the context of the proposed snake filtering technique.

3. Proposed algorithm

Instead of analyzing the levels sets of an image independently, the snake filter starts at local maxima in intensity and progressively expands, eliminating subscale components in the process. The basic principle behind the proposed snake filtering technique is that we want to explore the image plane, or more precisely want to visit each pixel location, in a prescribed order. The exploration starts at local maxima in intensity and moves towards neighboring pixels with successively decreasing gray levels. As a result we compute an area opening by front propagation on the image. We will show that with the aid of such a technique, we can also compute the perimeter of each front and use this information for an additional filtering criterion. Using the front propagation approach, other criteria can be combined with the area criterion in a straightforward manner.

Before detailing the proposed algorithm we need to examine the front propagation mechanism more closely. Sethian has defined three sets to characterize the movement of the front; they are the "known", "alive" and "far" set of points of the image domain on which the front moves [6]. Among these sets "known" points are points where the front has already reached, "far" points are points where the front is yet to reach, and "alive" points are the front itself. For our convenience let us name the "known" and the "alive" points together as a *front unit*. In our algorithm we identify the front unit as the *heap* data structure [7]. In the snake filtering context, one front unit starts from each image plateau (maximum in intensity). At initiation, the starting pixel on the image plateau is "known" and the neighboring pixels are "alive" and all other pixels are "far." As the front moves, the set of "known" pixels grows in number. Such front units can be treated independently until one front collides with another and they merge into an expanded front unit. It is interesting to note that the growing set of "known" pixels of such a front unit always remains connected and designates a connected component in an image level set.

Now we discuss the steps of the proposed algorithm:

Snake Filter

The following six steps are used in the snake filter:

1. Detect the local maximum regions or plateaus of the input image I. Such a region is defined to be the set S of pixels p such that if $p\in S$ and $q\in S$ then $I(p)=I(q)$. Also, there exists an unbroken path between p and q in S. If $q\notin S$ and q is a 4-connected neighbor of any $p\in S$ then $I(q)<I(p)$. In the input image I, let there be M such plateaus, $S[i]$, $i=1$ to M.
2. Select any one representative pixel, $r[i]\in S[i]$, for $i=1$ to M.
3. Create $Heap[i]$: $i=1$ to M, each having two types of lists, *Alive* and *Known*, to store pixel coordinates. Create two scalar fields *Length* and *Area* for each heap. Do: $Heap[i].Known \leftarrow$ NULL, $Heap[i].Alive \leftarrow Heap[i].Alive \cup \{r[i]\}$ and $Heap[i].Length \leftarrow 0$, $Heap[i].Area \leftarrow 0$, for $i=1$ to M.
4. Create the output matrix AOI, representing the area opening of I, using the same size as I. Let $minI$ denote the minimum gray level of I, *i.e.*, for any pixel location q, $minI \le I(q)$. Assign $minI$ to all elements of AOI.
5. Create a matrix T that is the same size as I. Create another matrix Far of the same size and initialize all its elements to 1. Do: $T(r[i]) \leftarrow 1/(I(r[i])-minI+1)$, and $Far(r[i]) \leftarrow 0$ for $i=1$ to M.
6. Repeat the following substeps until no remaining "alive" points exist in all M heaps:
   i. Let K be the set all "alive" points:
   $K = \bigcup_{i=1}^{M} Heap[i].Alive$,
   and let K' be the following set:
   $K' = \{q : q\in K \text{ and for any } m\in K,\ I(q)\ge I(m)\}$.
   Choose a point, $p \in K'$ such that for any $q\in K'$, $T(p)\le T(q)$. Let n denote the index of the heap from which p is selected. Do: $Heap[n].Known \leftarrow Heap[i].Known \cup \{p\}$ and $Heap[n].Alive \leftarrow Heap[i].Alive - \{p\}$.
   ii. Obtain all 4-neighbors – *up*, *down*, *right*, *left* of p. If $up\in Heap[i].Known$ for some $i\ne n$, then do: $Heap[n].Known \leftarrow Heap[i].Known \cup Heap[n].Known$, $Heap[i].Known \leftarrow$ NULL,

Heap[*n*].*Alive*←*Heap*[*i*].*Alive*∪*Heap*[*n*].*Alive*, *Heap*[*i*].*Alive*←NULL, *Heap*[*n*].*Area*←*Heap*[*i*].*Area*+*Heap*[*n*].*Area*, *Heap*[*n*].*Length*←*Heap*[*i*].*Length* + *Heap*[*n*].*Length*. Repeat for *down*, *right* and *left*.

iii. If $Far(up)=1$ then $Far(up) \leftarrow 0$, and *Heap*[*n*].*Alive*←*Heap*[*n*].*Alive* ∪ {*up*}. Repeat step iii for points - *down*, *right*, and *left*.

iv. If $I(up) \leq I(p)$ then *Speed*←1 else *Speed*←0. Recompute $T(up)$, substituting F in equation (1) by *Speed*. Repeat step iv for the points *down*, *right*, and *left*.

v. Do: *Heap*[*n*].*Area*←*Heap*[*n*].*Area*+1 and *Heap*[*n*].*Length*←*Heap*[*n*].*Length* + 4 − 2|*C*|, where |*C*| denotes cardinality of the set:
$C = \{left, right, up, down\} \cap (\bigcup_{i=1}^{M} Heap[i].Known)$.

vi. If *Heap*[*n*].*Area* ≥ *areascale* then for all $q \in Heap[n].Known$ and $AOI(q) = minI$, do: $AOI(q) \leftarrow I(p)$.

As already mentioned, the algorithm first identifies the local maximum plateaus (defined in step 1). From any point in a plateau at the maximum gray level, it then starts moving the front by the fast marching method. Step i guarantees that the point chosen is at the maximum gray level of all the points not chosen so far. On the other hand, the underlying front propagation strategy asserts that the chosen point always comes from a connected component. Thus we are able to compute the area opening of the image. Recomputation of the arrival time is done based on equation (1) and can be performed in $O(1)$ computational complexity. A detailed method for the recomputation of arrival time can be found in [6].

By conceiving area open operation with a front propagation method, we obtain not only the area, but also the perimeter of each connected component of a level set at any time instant. Steps ii, iii and iv show that connected component perimeter computation is also obtainable with such a technique. Knowing both perimeter and area of a connected region allows one to build up a class of filters that are more general and powerful than area open (area close) filters. As an example application, we will show in this paper that this general filtering technique is efficacious for cell detection and cell nucleus detection.

4. Computational complexity

For the snake filter algorithm, the requisite implementation should efficiently and rapidly 1) compute the minimum/maximum element of a set; 2) take unions of two sets; 3) insert and delete elements in a set; and 4) decrease the key value of an element of a set. A good choice that fulfills all these requirements includes any mergeable heap, *e.g.*, a binomial heap or a Fibonacci heap [7]. For a total of N pixels in an image, it can be shown that with a binomial heap, the worst case computational complexity of the proposed algorithm is $O(N\log_2 N)$. Amortized computational complexity in Fibonacci heap can even be $O(N)$ [7]. We also know that for area open or close operation the lower bound is $O(N)$ since all the image pixels should be examined at least one time. An implementation with a Fibonacci heap typically approaches this lower bound. A straightforward implementation of area open (or area close) operation takes $O(LN)$ operations for a L-gray level image [4] and hence is much slower than the proposed snake filtering with fast marching. The worst case computational complexity of our algorithm is comparable to that of some other fast algorithms, such as the pixel-queue or union-find method for area openings and closing [5].

A further reduction of complexity in the proposed algorithm is possible by exploiting the inherent independent nature of the front units or heaps. A front unit propagates its front independent of other fronts belonging to other front units until collision. At the collision, two or more front units merge. After merging the merged front and remaining ones can again start growing independently. A parallel version is thus possible. The degree of parallelism will of course depend on the image data – on the number of fronts M.

5. Results

A suitable application of our proposed snake filtering process could be the detection of the nucleus of a cell knowing the approximate area and perimeter of the nucleus. Other prior knowledge about the nucleus, which is also utilized here, is that the nucleus appears brighter than its surroundings in these images. Thus we first apply area close to the cell image shown in Figure 1. The area threshold for this image is 500. This preprocessing removes any inside hole (darker in appearance than the nucleus itself) of the nucleus leaving the nucleus shape and size intact. Next, on the area-closed image we apply the proposed snake filtering with an area threshold

of 500 and a perimeter threshold of 150. The snake filtering selects connected components of a level set that are greater than 500 in area but are less than 150 in perimeter. The result of filtering is shown in Figure 2. More precisely, Figure 2 shows the union (OR) of all the level sets retained through snake filtering. Figure 3, on the other hand, shows the area-closed and then area-opened image. Though Figure 3 gives some indication about the location of the nucleus, the snake filter is able to extract the location, as shown in Figure 2. In Figure 4, we overlay the edge of the detected nucleus edge on the original cell image.

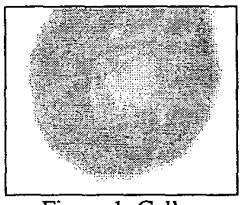

Figure 1. Cell.

Figure 2. Detection of nucleus by snake filter.

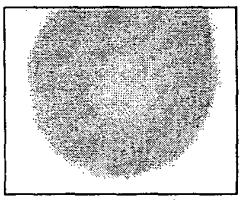

Figure 3. Result of area close-open.

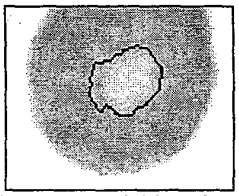

Figure 4.Overlaid nucleus boundary on the cell.

In the next example we apply the snake filtering method to detect leukocytes from the video frames obtained by transillumination of the cremaster muscle in a living mouse. Recently, much attention has been given to the study of rolling leukocytes *in vivo*, as the motion of these white blood cells gives clues about the mechanism of inflammatory disease [8]. So, automated tracking of leukocytes is an important problem.

To this end, the very first step in an automated tracking of leukocytes from *in vivo* video sequences would be detecting the leukocytes. The leukocytes may appear either darker or brighter than the surrounding background as Figure 5 illustrates. The area and the perimeter scale are chosen as 150 and 75, respectively, for such leukocytes. To detect leukocytes brighter than the surrounding we first apply area close to the image, and then apply the snake filter to the area-closed image to achieve area opening with the perimeter constraint.

The result is shown in Figure 6. Figure 7 shows the result of area close and area open operations applied in succession to the image in Figure 5 with same area scale of 150. For detecting the darker leukocyte of Figure 5 we reverse our strategy, *viz.*, we first apply area open and then apply snake filtering to implement area close on the area-opened image with a perimeter constraint of 75. The result is shown in Figure 8. Figure 9 shows the area open-close result of Figure 5. The snake filter is able to determine the location of the leukocyte without additional analysis. Figure 10 shows the overlaid contour of the detected leukocytes on the original leukocyte image.

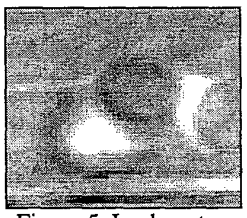

Figure 5. Leukocytes.

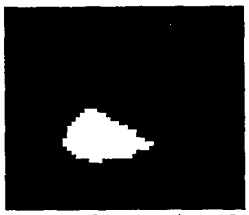

Figure 6. Detection of the brighter leukocyte through snake filter.

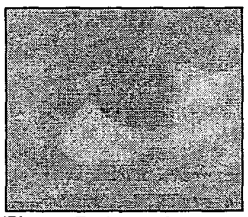

Figure 7. Area close-open of the leukocyte image.

Figure 8. Detection of the darker cell using the snake filter.

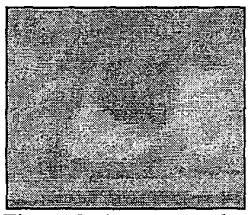

Figure 9. Area open-close of the leukocyte image.

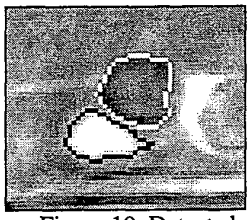

Figure 10. Detected leukocyte contours are overlaid on the leukocyte image.

6. Conclusion

This paper proposes an efficient morphological filtering for image analysis. The proposed filtering, based on the fast marching technique for front propagation, can filter connected components within

image level sets based on the area as well as the perimeter of the level set. Important applications of such filtering are also identified in this paper. In our future research we aspire to develop this contour based filtering into a scale space theory and to design additional snake filters that are sensitive to the shape of connected components.

References

[1] L. Vincent, "Morphological area openings and closings for grey-scale images," In *Shape in picture: Mathematical description of shape in grey level images*, Y.-L.O, A. Toet, D. Foster, H. J. A. M. Heijmans, and P. Meer, editors, pp.197-208, NATO, 1993.

[2] F. Cheng and A.N. Venetsanopoulos, "An adaptive morphological filter for image processing," *IEEE Transactions on Image Processing*, vol.1, pp.533-539, 1992.

[3] P. Salembier and J. Serra," "Flat zones filtering, connected operators and filters by reconstruction," *IEEE Transactions on Image Processing*, vol.8, pp. 1153-1160, 1995.

[4] S. T. Acton, "Fast algorithms for area morphology," *Digital Signal Processing*, vol.11, pp.187-203, 2001.

[5] A. Meijster and M.H.F. Wilkinson, "A comparison of algorithms for connected set openings and closings," to appear in *IEEE Trans. Pattern Anal. Mach. Intell.*

[6] J. Sethian, *Level set sethods and fast marching methods*. Cambridge University Press, 1999.

[7] T. H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to algorithms*. Cambridge, MA: MIT Press, 1990.

[8] K. Ley, "The selectins as rolling receptors," In *The Selectins: Initiators of Leukocyte Endothelial Adhesion*. D. Vestweber, editors, Amsterdam: Harwood Academic Publishers, 1997.