Graph Partitioning Active Contours (GPAC) for Image Segmentation

Baris Sumengen, B. S. Manjunath
Department of Electrical and Computer Engineering
University of California, Santa Barbara, CA
{sumengen, manj}@ece.ucsb.edu

***Abstract*— In this paper we introduce new type of variational segmentation cost functions and associated active contour methods that are based on pairwise similarities or dissimilarities of the pixels. As a solution to a minimization problem, we introduce a new curve evolution framework, the graph partitioning active contours (GPAC). Using global features, our curve evolution is able to produce results close to the ideal minimization of such cost functions. New and efficient implementation techniques are also introduced in this paper. Our experiments show that GPAC solution is effective on natural images and computationally efficient. Experiments on gray scale, color, and texture images show promising segmentation results.**

***Index Terms*— curve evolution, active contours, image segmentation, pairwise similarity measures, graph partitioning.**

I. Introduction

In the past, variational methods have defined various cost functions for the task of image segmentation. A popular tool for the (local) minimization of some of these cost functions is the curve evolution framework and active contour methods (ACM). These variational cost functions can be roughly categorized as contour modeling, region modeling or a combination of these.

An example of a contour modeling cost function is the one proposed by Caselles et al. [1] for the geodesic active contour framework. The cost is defined along a curve C and minimized by evolving the curve in the normal direction.

$$\underset{C}{Min}\, E = \oint g(C)ds \tag{1}$$

where $g = 1/(1 + |\nabla \hat{I}|)$ and $\hat{I}$ is the Gaussian smoothed image. Due to the local minimization, this type of edge-based approaches depend on where the curve is instantiated. By Initializing curves at different image locations, different objects of interest can be captured. Global minimization techniques related to graph partitioning have also been applied to edge-based curve evolution [2, 3].

A well known example for the region modeling cost function is the Mumford-Shah functional [4]. A simplified version of this functional, which models the image with piecewise constant functions, has been minimized within the curve evolution framework by Chan et al. [5]:

$$\underset{C}{Min}\, E = \alpha \iint_{R_i} (I - c_1)^2 + \beta \iint_{R_o} (I - c_2)^2 + \gamma \oint ds \tag{2}$$

where R_i corresponds to the interior and R_o corresponds to the exterior of the curve C, c_1 and c_2 are constants.

We introduce a new class of variational cost functions that are based on pairwise similarities or dissimilarities between points. The most basic version of such cost function is:

$$\underset{C}{Min}\, E = \iint_{R_i} \iint_{R_o} w(p_1, p_2) dp_1 dp_2 \tag{3}$$

where $p_1 \in R_o$, $p_2 \in R_i$ and $w(p_1, p_2)$ is a metric for the similarity between the points p_1 and p_2. We will use the notation $w(p_1, p_2)$ for representing both similarity and dissimilarity measures within this paper and the meaning should be clear from the context. If w is a dissimilarity measure then (3) is maximized. The objective of minimizing (3) is to minimize the similarity between the regions R_i and R_o. We will show later in Section III that (3) also maximizes the similarity within the regions R_i and R_o.

Within ACM framework, the evolution of the curves can be controlled by segmentation cost functions, external force fields and geometric forces such as curvature flow and constant expansion forces (e.g., balloons, see [6]). Based on the driving force behind curve evolution, ACM is divided into two groups, edge-based and region-based ACM. Edge-based ACM attempt to fit an initial curve to its surrounding edges as best as possible. Usually the results are highly dependent on where the curve is initialized. On the other hand, region-based ACM use regional features of the interior and the exterior of the curve and it is well established that region-based active contours are not as dependent on the initial contour as their edge-based counterparts [7, Chapter 4]. If a curve is initialized at one side of the image, there is a chance that it may not reach the other side of the image. However, a grid-wise initialization of multi-part curves addresses this problem. Since we are solving a local minimization problem, this still does not guarantee a global minimum. On the other hand, our experimental results suggest that most of the time the curve evolution converges to a reasonably good segmentation result. Usually most local minima can be avoided.

Previous work on region-based ACM segmented the images by modeling them as piecewise constant [5], piecewise smooth functions [8], by maximizing separation of the mean or variance of neighboring regions [9], or by clustering the histogram first to estimate region statistics offline and then tuning the ACM to these statistics [10]. These methods are typically based on statistics of unknown regions and make a priori assumptions about the image characteristics. In this paper, we propose a region-based ACM, *graph partitioning* *active contours* (GPAC), as a solution to the minimization problem given in (3).

One popular tool for minimizing pairwise similarity based cost functions is graph partitioning methods (GPM). A general advantage of GPM is their use of global minimization techniques, which are desirable for the segmentation problem. On the other hand, due to the computational issues, GPM often impose restrictions and simplifications to the original problem, which might compromise the segmentation quality. For example, in normalized cuts approach [11] pairwise similarities are restricted to local neighborhoods.

A challenge in tackling the problem given in (3) by using curve evolution techniques is finding computationally efficient methods. In Section IV, we introduce efficient implementation methods to address this issue. The implementation techniques are generic and applicable to most pairwise similarity based cost functions.

Main contributions of this paper include:

- *New variational cost functions*: Introduction of a new class of variational cost functions that are based on pairwise similarities. Since such cost functions are popular for GPM, we are able to combine and integrate many novel ideas from both variational and graph partitioning methods to create better techniques.
- *Curve evolution solution to these cost functions*: We derive steepest descent minimization of various pairwise similarity based cost functions within the curve evolution framework. Minimization of such complex segmentation cost functions has not been attempted in the past within the variational framework.
- *An efficient implementation framework*: Minimization frameworks for pairwise similarity based cost functions turn out to be computationally very expensive. Due to the excessive memory and CPU requirements, naive implementations are not practical even on high end workstations. We introduce novel numerical methods for efficient implementation of the curve evolution techniques that are derived for the minimization of pairwise similarity based cost functions.

The rest of the paper is organized as follows. In Section II we derive the curve evolution solution for the cost function given in (3). In Section III we analyze the problems with the original curve evolution solution and suggest maximum cut framework as an improvement. In Section IV, we discuss an efficient implementation framework for pairwise similarity based cost functions. Section V analyzes the behavior of ACM and GPM. In Section VI, we derive curve evolution solution for various well-known cost functions that are based on pairwise similarities. Section VII provides experimental results using GPAC. In Section VIII-A, we discuss different ways of integrating the edge information to GPAC. We conclude in Section VIII-B.

II. Curve Evolution based on Minimum Cut Criterion

We now derive a curve evolution solution to the minimum cut problem. The minimum cut problem originates from graph partitioning [11-16]. Minimum cut criteria can also be written as an energy functional in continuous domain. In this case, the problem is formulated as the partitioning of a continuous image with a curve, as opposed to a graph cut.

Assume $G = (V, E)$ is a representation of an undirected graph, where V are the vertices and E are the edges between these vertices. V can correspond to pixels in an image or small regions (set of connected pixels). Image segmentation problem can be formulated as the best bi-partitioning of the image (cutting the graph) into two regions, A and B. Consider the following cost function that is minimized by minimum cut technique [12], which is a graph partitioning method.

$$cut(A, B) = \sum_{u \in A, v \in B} w(u, v) \tag{4}$$

where $w(u, v)$ is a similarity metric. In continuous domain, the equivalent energy functional of (4) is then written as:

$$E = \iint_{R_i(C(t))} \iint_{R_o(C(t))} w(p_1, p_2) dp_1 dp_2 \tag{5}$$

where C is a curve, t is the time parameter of the evolution of C and R_i and R_o are the interior and the exterior of this curve. We solve this minimization problem using steepest descent method where we instantiate a curve and evolve this curve towards the minimum.

Theorem 2.1: Let $\vec{N}$ be the outward normal of the curve C. The curve evolution equation that corresponds to the steepest descent minimization of (5) is:

$$\frac{\partial C}{\partial t} = \left(\iint_{R_i(C(t))} w(c, p) dp - \iint_{R_o(C(t))} w(c, p) dp \right) \vec{N} \tag{6}$$

where c is a point on the curve C.

Proof:

To find the steepest descent equations, we need to calculate the first variation of (5). Before attempting this, first we rewrite (5) as $M = \iint_{R_i(C(t))} G(X, t) dX$, where X is a point in 2-D and $G(X, t) = \iint_{R_o(C)} w(X, Y) dY$. Let us write M as:

$$M(t'(t), \tau(t)) = \iint_{R_i(C(t'))} G(X, \tau) dX \tag{7}$$

where $\tau(t) = t$, $t'(t) = t$, $C = C(t, p)$ is a closed curve and $p \in [0, 1]$ is a parametrization of this curve. The first variation of (7) with respect to t is:

$$\frac{\partial M}{\partial t} = \frac{\partial M}{\partial t'} \frac{\partial t'}{\partial t} + \frac{\partial M}{\partial \tau} \frac{\partial \tau}{\partial t} \tag{8}$$

We first calculate the second term, then we will calculate the first term.

$$\begin{aligned} \frac{\partial M}{\partial \tau} &= \frac{\partial}{\partial \tau} \iint_{R_i(C(t))} G(X, \tau) dX \\ &= \iint_{R_i(C(t))} \frac{\partial}{\partial \tau} G(X, \tau) dX \end{aligned} \tag{9}$$

Before calculating the first variation of M with respect to t', we write M as a boundary integral using the divergence theorem. To do this, we define a vector $\vec{S}$ as:

$$\vec{S} = \begin{bmatrix} \frac{1}{2}\int_0^x G(\lambda, y)d\lambda \\ \frac{1}{2}\int_0^y G(x, \lambda)d\lambda \end{bmatrix} \tag{10}$$

As can be seen, divergence of $\vec{S}$ is equal to G: $\nabla \cdot \vec{S} = G$. Using the divergence theorem, we can write M as:

$$\begin{aligned} M &= \iint_{R_i(C(t'))} \nabla \cdot \vec{S}\, dX \\ &= \oint_C \left\langle \vec{S}, \vec{N} \right\rangle ds \end{aligned} \tag{11}$$

where $\vec{N}$ is the *outwards* normal vector of the curve and $\langle , \rangle$ denotes the scalar product. Derivation of the first variation of (11) with respect to t' has been given by Zhu et al. [17, Appendix], Tsai [7, Appendix A] and Vasilevskiy et al. [18] independently. For completeness purposes, we also include our version of the solution to this problem in the Appendix. The first variation of (11) with respect to t' is then:

$$\frac{\partial M}{\partial t'} = \oint_{C(t')} \left\langle C_{t'}, G\vec{N} \right\rangle ds \tag{12}$$

where C_t corresponds to the derivative of C with respect to t. Combining (8), (12), and (9), we find that

$$\frac{\partial M}{\partial t} = \oint_{C(t)} \left\langle C_t, G\vec{N} \right\rangle ds + \iint_{R_i(C)} \frac{\partial}{\partial t} G(X, t) dX \tag{13}$$

Now, going back to the problem of calculating the first variation of (5), we utilize the result from (13) to solve this problem. By inserting $G(X, t) = \iint_{R_o(C)} w(X, Y)dY$ within (7), (5) becomes equivalent to (7). Using (13), we can write the first variation of (5) as:

$$\begin{aligned} \frac{\partial E}{\partial t} &= \oint_C \left\langle C_t, \left[\iint_{R_o(C)} w(c, p_2)dp_2\right] \vec{N} \right\rangle ds \\ &\quad + \iint_{R_i(C)} \frac{\partial}{\partial t} \left[\iint_{R_o(C)} w(p_1, p_2)dp_2\right] dp_1 \\ &= \oint_C \left\langle C_t, \left[\iint_{R_o(C)} w(c, p_2)dp_2\right] \vec{N} \right\rangle ds \\ &\quad + \iint_{R_i(C)} -\oint_C \left\langle C_t, w(p_1, c)\vec{N} \right\rangle ds\; dp_1 \\ &= \oint_C \left\langle C_t, \left[\iint_{R_o(C)} w(c, p)dp - \iint_{R_i(C)} w(c, p)dp\right] \vec{N} \right\rangle ds \end{aligned} \tag{14}$$

where c is a point on the curve C. In these calculations, we used the fact that $w(p_1, p_2)$ is a symmetric function and is not a function of t. Thus the first variation of G can be calculated as $\partial G/\partial t = -\oint_{C(t)} \left\langle C_t, w\vec{N} \right\rangle$. When integrating on R_o, the normal vector is in the opposite direction, hence the minus sign. From (14) we can see that E decreases fastest when:

$$\frac{\partial C}{\partial t} = \left(\iint_{R_i(C)} w(c, p)dp - \iint_{R_o(C)} w(c, p)dp\right) \vec{N} \tag{15}$$

This concludes the proof. ■

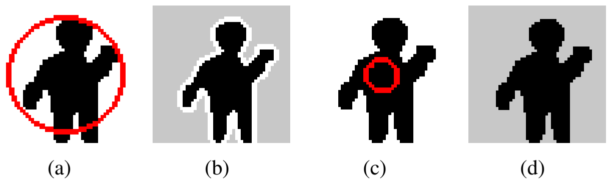


Fig. 1. a) A large balanced curve initialized, b) foreground object captured correctly, c) small curve initialized within the foreground, d) initial curve shrinks and disappears.

Note that each point c on the curve C is compared to the points within the interior and the exterior of the curve for similarity. If c is more similar to R_i, the curve expands, and if c is similar to R_o then the curve shrinks. This guides the curve towards the boundary between foreground and background regions in the image. Note that the theory for this result assume that the image only consists of a foreground and a background.

III. Maximum Cut and Region Stability

One problem with the minimum cut criterion, which has been pointed out in [11, 12], is that it favors cutting out small partitions. This is so, because for small partitions, the total sum across the cut is small. Our minimum cut framework introduced in Section II also inherits this problem. There is another problem with our minimum cut framework as illustrated in Fig. 1. Fig. 1(a) shows a black object against a white background. Let the similarity measure be $w(p_1, p_2) = \exp(-\frac{|I(p_1)-I(p_2)|}{\sigma_I})$ where I corresponds to pixel intensities. Fig. 1 shows results for two different instantiations of the curves. In Fig. 1(a), a more balanced curve is initialized, where the interior of the curve mostly consists of the foreground object and the exterior consists of the background. The curve evolution finds the correct result easily. In Fig. 1(c), instantiation of a smaller curve within the foreground shows one of the problems in using (15). The similarity of each point on the curve to both the interior and exterior of the curve is summed and compared according to (15). In this example, there are more black points outside the curve than there are inside. So, the total similarity to R_o is higher, making $\frac{\partial C}{\partial t}$ a negative number. This causes the curve to shrink and disappear (Fig. 1(d)), which is not the ideal result. The reason for this behavior can be tracked back to the choice of the similarity measure. In this example we chose the similarity measure similar to the measures proposed in [11, 12]. On the other hand, choosing a similarity measure such as $w(p_1, p_2) = -|I(p_1) - I(p_2)|$ would help fix this problem[1]. Unfortunately, defining the highest possible similarity between pixels as 0 is counter-intuitive.

Let us define a dissimilarity measure instead of a similarity measure. Consider

$$w(p_1, p_2) = |I(p_1) - I(p_2)| \tag{16}$$

as a dissimilarity measure, where similar pixels have 0 dissimilarity whereas a jump in intensity values cause a high dissimilarity. We choose this dissimilarity measure for its

[1]This type of similarity function is proposed in [14, Section 5.2] for ratio cut in a different context.

simplicity. More complex dissimilarity measures can integrate spatial distance of pixels and domain knowledge:

$$w(p_i,p_j)=\|\vec{F}(p_i)-\vec{F}(p_j)\|+\alpha\|p_i-p_j\|+\beta\|m(p_i)-m(p_j)\| \quad (17)$$

where $\vec{F}(p_i)$ is some low level image feature at point p_i (e.g. color), the second term measures the spatial distance between two points and the third term measures the distance using a function $m()$, which represents some sort of domain knowledge related to the problem at hand. We have successfully utilized such dissimilarity measures within GPAC framework for pruning categories in image databases [19]. Choice of the (dis)similarity measure is a research issue by itself and we will not address this problem in this paper. Based on these definitions of the dissimilarity, we also change the segmentation criterion from minimizing the graph cut to *maximizing* the cut in (5), which corresponds to maximizing the dissimilarity across the cut. We call this framework as *maximum cut*. Maximum cut addresses both of the shortcomings observed with the minimum cut (but introduces a bias towards equally sized partitions). The integral in (5) is maximized when there are as many connections as possible, which encourages larger partitions as opposed to the minimum cut's behavior of favoring small isolated regions. We also observe that both curve instantiations have a tendency to converge to the ideal result in Fig. 1. Since the cost function is maximized as opposed to being minimized, the new evolution equation is the negative of (15).

$$\frac{\partial C}{\partial t}=\left(\iint_{R_o(C)}w(c,p)dp-\iint_{R_i(C)}w(c,p)dp\right)\vec{N} \quad (18)$$

Similar to maximum cut framework, one of the objectives of normalized cut [11] is to fix the behavior of favoring small regions in minimum cut. Even though cost function for maximum cut is different from the cost function used in normalized cuts, there is some parallelism between them[2]. To clarify this consider the following intra-region dissimilarity:

$$E_2=\iint\limits_{R_i}\iint\limits_{R_i}w(p_1,p_2)dp_1dp_2+\iint\limits_{R_o}\iint\limits_{R_o}w(p_1,p_2)dp_1dp_2 \quad (19)$$

where $w(p_1,p_2)$ is a dissimilarity measure. The objective is to minimize the intra-region dissimilarity. If we find the curve evolution equation for minimizing E_2, we observe that it is exactly the same as (18). This is not surprising since $E_2=\iint_R\iint_R w(p_1,p_2)dp_1dp_2-2E=C-2E$, where $R=R_i\cup R_o$, and C is a constant. We can conclude that using a dissimilarity measure encourages similarity within the partitions while discouraging inter-region similarity. A parallel argument is made for normalized cuts based on the region associations [11].

In Fig. 2, we corrupt the binary image with Gaussian noise and apply the maximum cut segmentation. As can be seen, for two different curve instantiations, several one pixel wide noisy regions are captured. Note that the white points in 2(b) and (d) correspond to the curve and all boundaries together

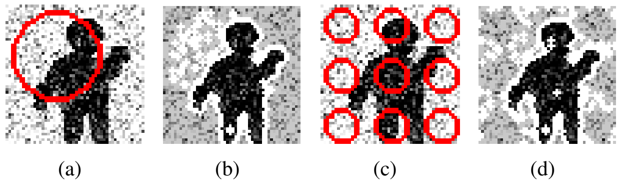


Fig. 2. Demonstration of how noise can effect the curve evolution. a) Single curve initialized overlapping both the foreground and the background, b) corresponding curve evolution result (all white points correspond to the curve), c) a multipart curve initialized in a grid fashion, d) corresponding curve evolution result (all white points correspond to the curve). The curve evolution on a noisy image splits the curve into many small pieces.

correspond to a single curve over which the cost function is optimized. GPM usually get around this problem by enforcing region connectivity, decreasing similarity with spatial distance and size constraints in its cost functions [14]. ACM solves this problem by adding a curvature flow component, which is a geometric component that smooths the curve at each iteration. Curvature flow can also be derived as minimizing the curve length $\oint ds$, which means that it disfavors splitting of the curve to small one pixel wide boundaries. The new evolution equation can be written as:

$$\frac{\partial C}{\partial t}=\left(\iint_{R_o}w(c,p)dp-\iint_{R_i}w(c,p)dp\right)\vec{N}-\gamma\kappa\vec{N} \quad (20)$$

where γ is a constant and κ is the curvature of the evolving curve.

Fig. 3 demonstrates curve evolution for four different types of initializations of the curves. All four of these evolutions are conducted using (20). This example illustrates that (a) for a simple and connected object, maximum cut curve evolution has a tendency to converge to the same segmentation result, and (b) that curvature-based flow increases robustness of the curve evolution under noisy conditions.

Normalized Maximum Cut

One of the advantages of using geometric ACM is that we can introduce geometric properties or constraints into the curve evolution equation without changing or re-solving the energy minimization problem. Since curve evolution is an iterative process, we can add normalization for the integrals in (20) by dividing them with their corresponding areas. This can be thought of as re-solving the curve evolution equation at each iteration[3]. We call this setup as *normalized maximum cut*. The evolution equations become:

$$\frac{\partial C}{\partial t}=\left(\frac{1}{A_o}\iint_{R_o}w(c,p)dp-\frac{1}{A_i}\iint_{R_i}w(c,p)dp\right)\vec{N}-\gamma\kappa\vec{N} \quad (21)$$

This shows the flexibility of active contour framework compared to GPM. Fig. 4 shows normalized maximum cut segmentation on an image that is corrupted by Gaussian noise and applied an illumination effect to the top left corner.

One of our target application domain is natural images. Fig. 5 shows maximum cut segmentation results on a flower image

[2]See Section VI-C for a curve evolution solution of the normalized cuts method.

[3]In Section VI-B, we will also show the curve evolution solution if A_i and A_o are taken as functions of t

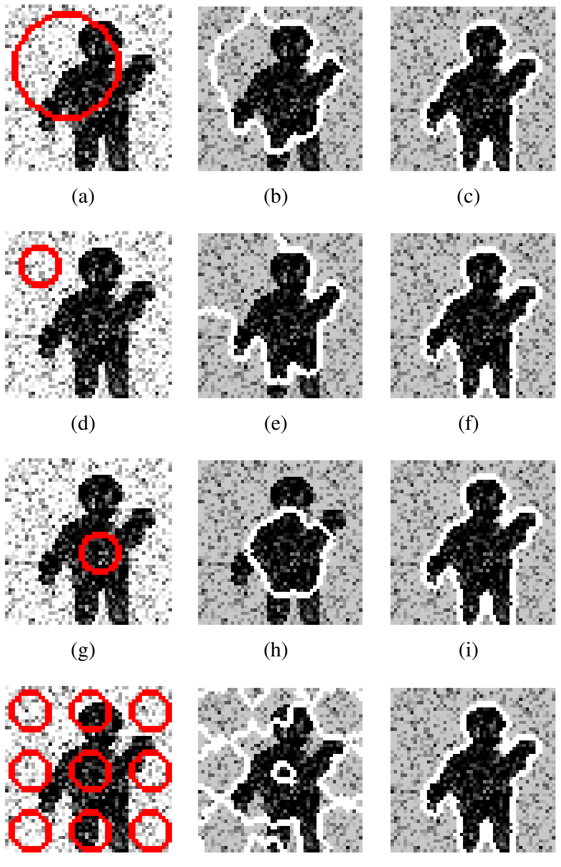


Fig. 3. Each row corresponds to maximum cut evolution under 4 different type of curve instantiations. First column corresponds to various initializations of the curve. Second column shows a state of the curve during the evolution. Third column shows the segmentation result at convergence. This example demonstrates the stability of maximum cut framework with respect to curve instantiation and noise.

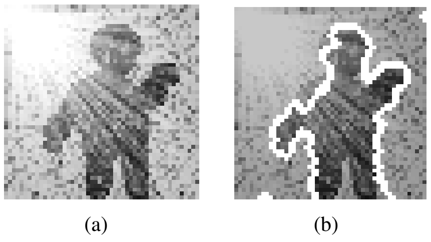


Fig. 4. Segmentation of an image corrupted by Gaussian noise and applied an illumination effect. a) Original image, b) corresponding curve evolution result using normalized maximum cut segmentation.

for $\gamma = 0.005$ and $\gamma = 0.1$ using color information. Distance between color features are calculated using L_1 distance. Our main algorithm stays the same except the use of a different dissimilarity measure. Fig. 5(a) is not an easy image for edge-based methods since there are many edges within the foreground and significant clutter in the background. For small γ more details are captured whereas for large γ connected regions with smoother boundaries are favored. In this experiment, one multi-part curve with 144 sub-parts is initialized uniformly over the image. We use this type of initialization for the rest of the paper. Note that if two sub-parts of the curve touch each other, their boundaries will merge. Level set methods [20], which are the numerical methods used for all the implementations in this paper, is able to handle merging and splitting of the curves naturally and also automatically keep track of what is the interior and what is the exterior to a given curve.

Fig. 6 shows normalized maximum cut segmentation for various curve instantiations. Fig. 6(d) demonstrates the possibility that GPAC converges to an undesired local maximum. On the other hand, gridwise instantiations of large number of curves converge to similar and reasonably well segmentations (Fig. 6(g-l)).

Both maximum cut and normalized maximum cut when applied to Fig. 5(a) are able to segment the foreground from the background successfully. In general we observe that normalized maximum cut gives slightly better results. Main reason for this is that maximum cut favors equal size regions. For images where the foreground is larger or smaller than the background, it is more efficient to use normalized maximum cut.

IV. Efficient Implementation

The integral calculations in (18) are usually the bottleneck in terms of computational complexity of the curve evolution. On the other hand, curve evolution itself can be implemented efficiently and accurately using narrow band level set methods. Suppose the image is of size $N \times M$, then we need to calculate and keep in memory about $N^2M^2/2$ dissimilarities. This is equivalent of generating a symmetric dissimilarity matrix W with NM rows and columns, where an element in the ith row and jth column is $w(p_i, p_j)$. Even for small images, this will become hard to fit into memory and require too many computations. We will address these issues in this section and propose an efficient way of calculating dissimilarities and implementing the curve evolution given in (18).

For an efficient implementation of our framework, we create a dissimilarity matrix W' of size $NM \times nm$, where $n \ll N$ and $m \ll M$. The elements of W' correspond to the dissimilarity of each pixel (x, y) to a subregion of the image, e.g. a rectangular tile. W' is an approximation of W and we will demonstrate that minimal segmentation precision is lost by this approximation. Consider a partitioning of the image area where we divide the image into $n \times m$ equal size tiles T_{ij} and average the image features $F(x, y)$ within each tile.

$$F'_{ij} = \frac{1}{A_{ij}} \int_{T_{ij}} F(x, y)dxdy \tag{22}$$

The elements of W' for each coordinate (x, y) that falls into the tile T_{ij} are calculated as $\|F(x, y) - F'_{ij}\|$.

Using W', (18) can be implemented efficiently. For each point c on the curve (actually for all points in the narrow band), the summation is done over all the tiles whose center falls inside and outside the curve. In doing this, we exclude the tile containing c from the calculations. Assume T_k, $k = 1, \ldots, nm$ are the tiles, P_k is the center coordinate of T_k. The difference of integrals in (18) can be simplified to $\sum_{k,P_k \in R_o} W'(c, k) - \sum_{k,P_k \in R_i} W'(c, k)$. As can be seen, we are representing the features in a tile with their centroid. This approach would

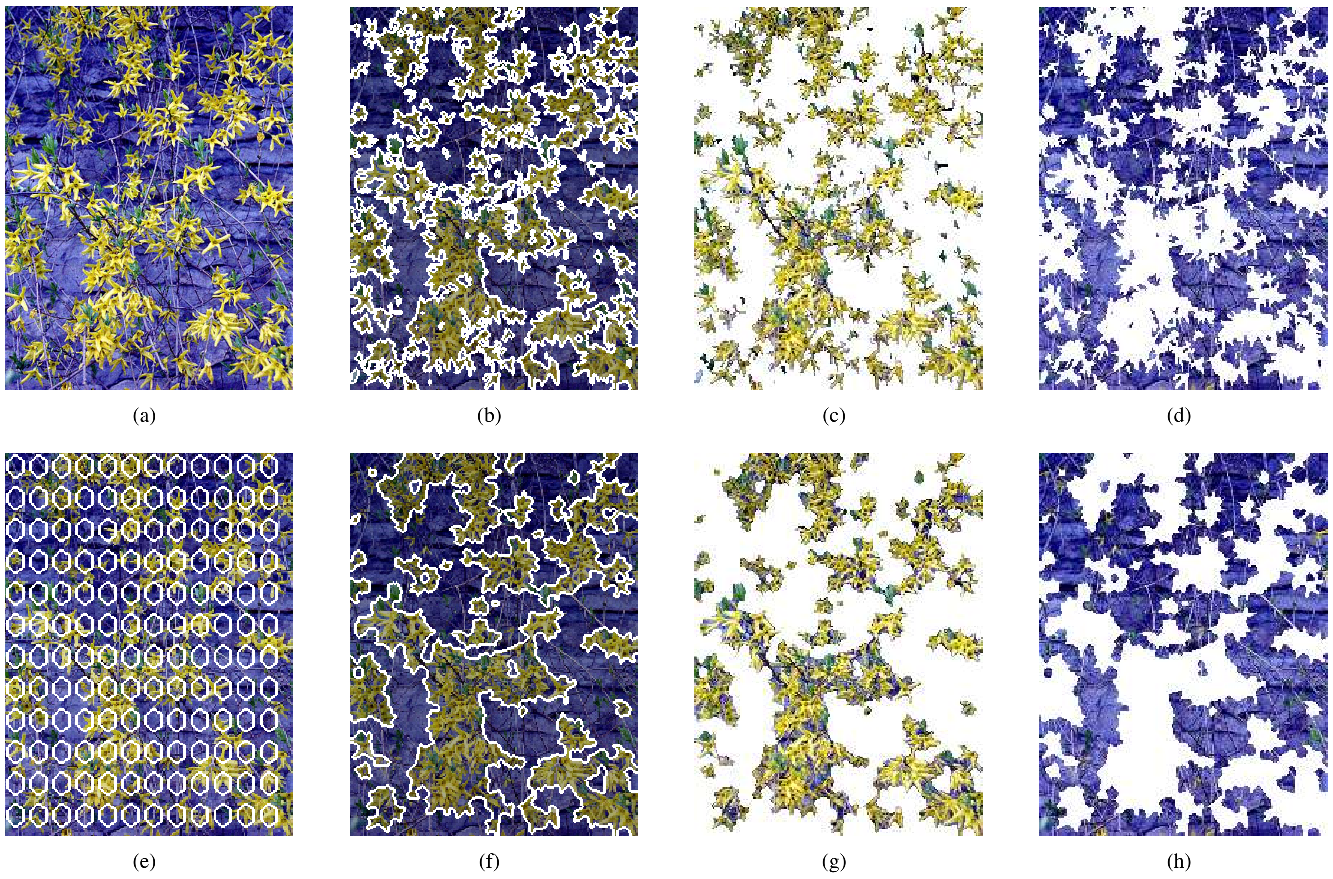


Fig. 5. a) Original image (266x200), e) initialization of a curve with 144 sub-parts, b-d) maximum cut segmentation with $\gamma = 0.005$, f-h) maximum cut segmentation with $\gamma = 0.1$.

also work with spatial terms in the dissimilarity metric. For dissimilarity measures for which this approach is not suitable, a random sampling or other representative features such as median can be used for the integral estimation.

Figures 9 and 10 visually show that the precision of the segmentation is not much affected by the approximation of W with W'. In all experiments in this paper, n and m are selected around 15 regardless of the image size. We choose the tiles as squares, where the dimensions of the tiles are $s_x = s_y = \lfloor Width/15 \rfloor + 1$. Then,

$$n = \left\lfloor \frac{Width - 1}{s_x} \right\rfloor + 1, m = \left\lfloor \frac{Height - 1}{s_y} \right\rfloor + 1 \quad (23)$$

In some rare cases, due to the large tile size in this approximation, the curve shrinks and disappears instead of converging to a segmentation boundary. This problem can be usually addressed by reducing the tile size.

It might seem surprising that even with quantizing one dimension of W, still a precise segmentation can be reached. This can be explained by observing the curve evolution equation (18). The energy functional given in (5) is symmetric for the dimensions of W. However, the curve evolution in (18) does not have the same symmetry. A single point c is compared to the rest of the points of the image. Coarsening the location of c would have a significant effect on the end result since even a neighbor of c might have a totally different feature value. On the other hand, approximating the integration over the rest of the points is more robust to errors, and allows us to reduce the resolution of one dimension of W without significantly affecting the end segmentation.

Consider a tile T_j of size N^2 located within the background R_o. Let T_j^s be the pixels inside this tile and c a point on the curve. Based on our simple dissimilarity measure, the normalized dissimilarity of c to T_j is $\frac{1}{N^2} \sum_s \|I(c) - I(T_j^s)\|$ without quantizing W. Using W', an approximation of the same dissimilarity can be written as:

$$\|I(c) - \frac{1}{N^2} \sum_s I(T_j^s)\| = \frac{1}{N^2} \left\| \sum_s I(c) - I(T_j^s) \right\| \quad (24)$$

As can be seen, if all $I(T_j^s)$ are smaller or larger than $I(c)$, both dissimilarities–at full resolution and after approximation–are equal. For most of the tiles, image features do not vary much within a tile (ignoring outlier points) if the tile is located inside an homogenous object. This is not the case for tiles overlapping an object boundary and this might introduce some inaccuracies into the dissimilarity calculation. We observe in our experiments that false movements of the curve because of approximations are usually corrected in the following iterations if c reaches an incorrect region as a result of these errors. Even though we do not pursue in this paper, size and geometry of the tiles can be selected more adaptively by analyzing the image features. Finally, a simple over-segmentation of the image can be used as the partitioning instead of tiles so that

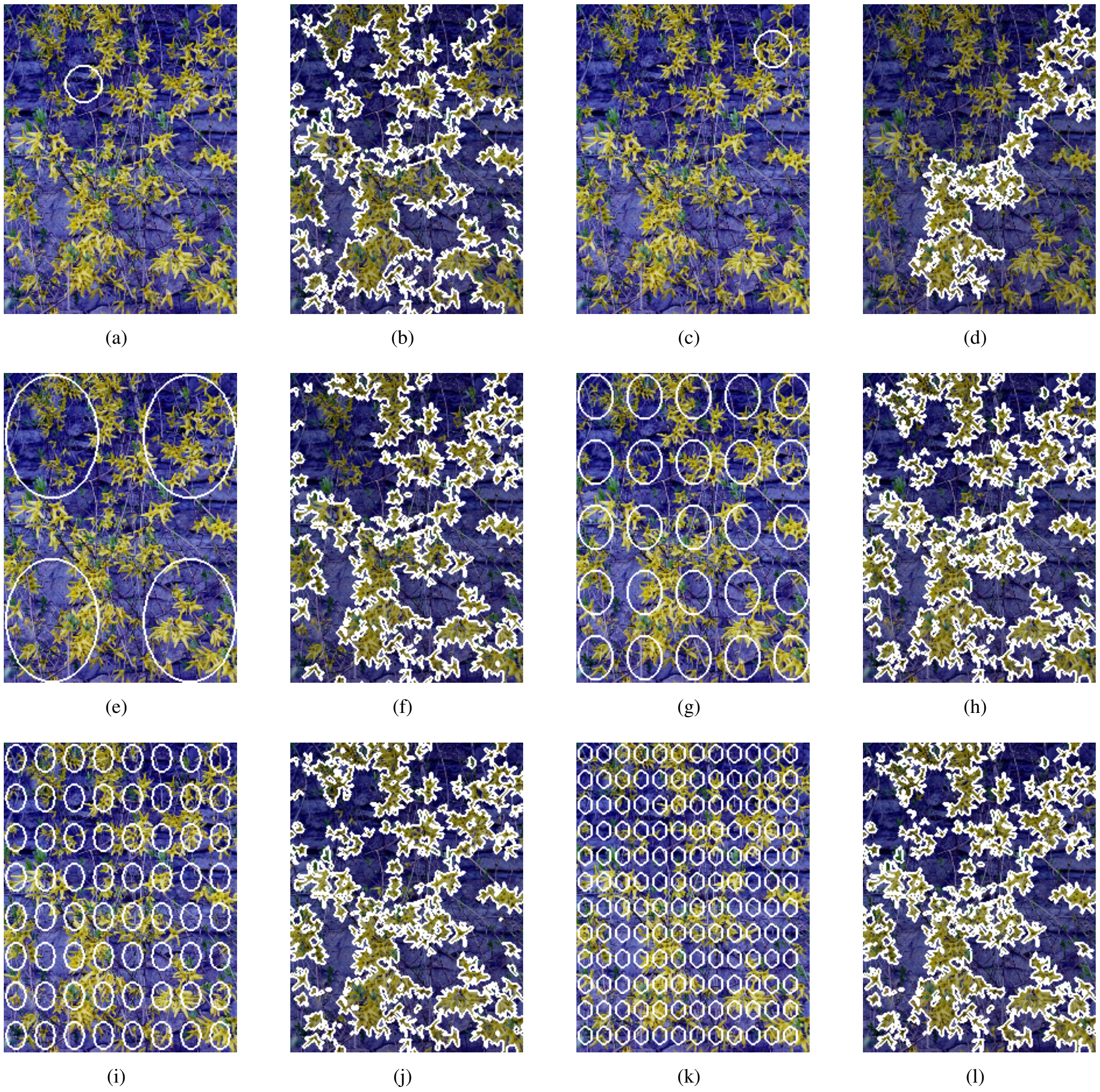


Fig. 6. Normalized maximum cut segmentation for various initializations of the curves with $\gamma = 0.005$.

image features are homogenous within each partition.

V. Active Contour vs Graph Partitioning

Pairwise (dis)similarity-based cost functions we introduced in this paper have been commonly used within the graph partitioning framework. One of the common techniques for minimizing the cost functions in GPM is by finding the appropriate clusters (regions) of graph nodes (e.g. pixels). Others include finding cycles (closed contours) on which a certain cost function is minimized [16]. A popular technique for clustering graph nodes is using the graph spectrum as proposed by normalized cuts [11] and average cut [13] methods.

Several GPM methods [21, 16], which are edge-based, compare their methods to edge-based ACM and pointed out the advantages of their global minimization properties. Our technique (GPAC) can be classified as a region-based ACM. There are several advantages of region-based ACM compared to spectral GPM.

- Due to the computational efficiency reasons, spectral GPM limits the similarity neighborhood size. By limiting the neighborhood where the (dis)similarity measure is calculated, GPM makes assumptions and simplifications to the problem at hand and associated segmentation cost functions. This approach may lead to undesirable results such as over-segmentation of large homogenous regions. In GPAC, there is no need to impose neighborhood size limitations that might affect the segmentation quality.
- The evolution of curves creates an iterative process. During the evolution, various conditions can be checked and considered so that the curve's evolution is adjusted accordingly. This introduces added flexibility to our framework. We utilized this property of curve evolution (a) when we added the curvature-based smoothing term in (20), and (b) when we introduced area normalization in (21). We demonstrate in Section VIII-A that iterative nature of the curve evolution can be used to integrate

edge information. While difficult, it is also possible to integrate other types of domain knowledge originating from the problem at hand.

GPAC is a curve evolution method that incorporates the advantages of pairwise similarity metrics that are commonly used in GPM. The performance and characteristics of these cost functions are rigorously analyzed both analytically using statistical models and empirically on different classes of images [22] (see [22] for more references on this topic). In contrast, the performance of region-based variational and active contour methods have only been demonstrated on a limited number of images, and we are not aware of any detailed analysis of the corresponding cost functions.

A. Analysis of the Behavior of Normalized Cuts with Increasing Similarity Neighborhood Size

Normalized cuts method [11] is based on finding the second smallest eigenvalue and corresponding eigenvector of the matrix $D^{-1/2}(D - W)D^{-1/2}$, where W is the similarity matrix and D is a diagonal matrix with the diagonals $d_i = \sum_j w(i,j)$. Efficient solution of this problem requires that the matrix is sparse. For this purpose, normalized cuts method restrict the similarities within small neighborhoods such that $w(s_i, s_j) = 0$ if $\|s_i - s_j\| > R$, where $\|.\|$ is the spatial distance and R is a threshold, eg. 20 pixels. As we will show, this limitation, which is motivated by computational concerns, may lead to undesirable results and over-segmentation of large homogenous areas.

Consider the segmentation (bi-partitioning) results in Fig. 7. For normalized cuts implementation we use the binary-only software provided by its authors at [23]. Figures 7(c-e) show normalized cut bi-partitioning when R is equal to 20, 40, and 80 pixels and image size is 253×187. The default of the software is set to $R = 20$. Increasing the value of R would bring the implementation closer to the solution of the original cost function. Fig. 7(e) shows normalized cut solution for $R = 80$. We can see that the homogenous background (sky) and the foreground are better separated. This shows that restricting R to 20 changes the result and diverts from the original segmentation cost function. On the other other hand, increasing the neighborhood size R reduces the sparseness of the similarity matrix and implementations become unpractical. For $R = 80$, it takes over 1GB memory and 25 minutes at 100% CPU load on a pentium 4 2GHz workstation.

Our purpose in this section is not to compare GPAC and normalized cuts, but to point out an important behavior of normalized cuts. GPAC solution to the same problem is given in Fig. 7(b). Since GPAC does not restrict the (dis)similarities to a neighborhood, the result is similar to the normalized cut solution with $R = 80$ given in Fig. 7(e). GPAC implementation takes between 5 to 30 seconds depending on the initial location of the curves and allocate less than 50MB memory using our unoptimized implementation written in C#. We should note that there are more recent papers by Sharon et al. [24, 25] and Fowlkes et al. [26] that propose efficient techniques for the graph partitioning based segmentation.

VI. Curve Evolution for Other Pairwise Similarity Based Cost Functions

One of the advantages of our variational framework is that the theory developed so far (including the implementation issues that are discussed in Section IV) can be easily adapted and applied to other cost functions that are based on pairwise (dis)similarities. To demonstrate this aspect of our framework, we now derive curve evolution equations for various GPM-based cost functions that address several different problems. References to GPM such as average cut and normalized cut are used to indicate the cost functions associated with these methods, but not to refer to the GPM-based solutions of these problems. This section shows that the proposed curve evolution framework that we introduced in this paper is general and not specific to a certain cost function.

A. Curve Evolution based on Boundary-normalized Cut

Boundary-normalized cut that we will introduce here is a way of normalizing the cost function by the boundary length between R_i and R_o. Recall that the maximum cut criteria favors having a large number of connections across the cut, which corresponds to a longer boundary length. So, it might be of interest to investigate a boundary-normalized cost function. Boundary-length normalization is also used by several GPM [16, 14]. We can write the corresponding cost function as:

$$E = \frac{\iint_{R_i(C)} \iint_{R_o(C)} w(p_1, p_2) dp_1 dp_2}{\int_0^1 |C_q(q)| dq} \tag{25}$$

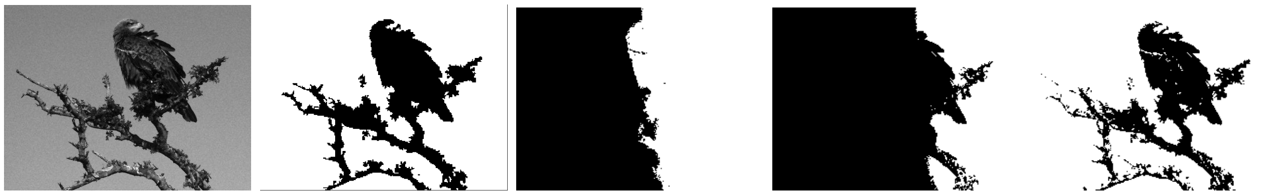
Let $K = \iint_{R_i(C)} \iint_{R_o(C)} w(p_1, p_2) dp_1 dp_2$ and $L = \int_0^1 |C_q(q)| dq$. The first variation of $E = K/L$ can be calculated as $E' = (K'L - KL')/L^2$. We already calculated K' in Section II. The solution of L' is previously studied [27] and it can be shown that $\partial L/\partial t = \oint_C \left\langle C_t, \kappa \vec{N} \right\rangle ds$. Based on these calculations, the evolution equation that maximizes (25) can be written as:

$$\frac{\partial C}{\partial t} = \frac{1}{L}\left[\left(\iint_{R_o} w(c,p)dp - \iint_{R_i} w(c,p)dp\right) - \frac{K}{L}\kappa\right]\vec{N} \tag{26}$$

The boundary length L is available during each reinitialization of the narrow band used in Level Set methods (See Section VII for more information). Calculating K can be expensive on the full image grid, but this value can be approximated on a low resolution grid for computational efficiency.

This result seems to be conceptually similar to the idea of introducing a curvature-based component as done in (20). On the other hand, the effects of the terms such as L and K are not clear. In the previous section, boundary-based constraint is introduced as an additional term in the cost function, whereas in (25) the cost function is normalized by the curve length. We can also write a version of (26) that is normalized by area:

$$\frac{\partial C}{\partial t} = \frac{1}{L}\left(\frac{1}{A_o}\iint_{R_o} w(c,p)dp - \frac{1}{A_i}\iint_{R_i} w(c,p)dp\right)\vec{N} \\ -\frac{K}{L^2}\kappa\vec{N} \tag{27}$$



(a) (b) GPAC (c) 20 pixel neighborhood (d) 40 pixel neighborhood (e) 80 pixel neighborhood

Fig. 7. a) Original image, b) GPAC bi-partitioning, c) normalized Cuts bi-partitioning when neighborhoods are limited to 20 pixels, d) normalized Cuts bi-partitioning when neighborhoods are limited to 40 pixels, e) normalized Cuts bi-partitioning when neighborhoods are limited to 80 pixels.

B. Curve Evolution based on Average Cut Criterion

In this section we find the descent equation for the average cut cost function:

$$Acut(A,B) = \frac{cut(A,B)}{|A|} + \frac{cut(A,B)}{|B|} \tag{28}$$

This cost function can be written in continuous domain as:

$$E = \frac{K}{\int_{R_i} dX} + \frac{K}{\int_{R_o} dX} = \frac{K}{A_i} + \frac{K}{A_o} \tag{29}$$

where K is defined in previous subsection and X is a point in 2-D. Note that the integrations in the denominator can also be done over a function $f(X)$, which might contain modeling information about the objects in the image.

Following a similar calculation as in Section VI-A, we can see that $E' = (K'A_i - KA_i')/A_i^2 + (K'A_o - KA_o')/A_o^2$. We calculated K' in Section II. A_i' and A_o' can also be calculated as special cases of (13) where $G(X) = 1$. This gives $\partial A_i/\partial t = \oint_C \left\langle C_t, \vec{N} \right\rangle ds$. For A_o the normal vector is in the opposite direction, which introduces a minus sign. The combined flow equation can then be written as:

$$\frac{\partial C}{\partial t} = \left[\left(\frac{1}{A_o} + \frac{1}{A_i} \right) H + \left(\frac{1}{A_o^2} - \frac{1}{A_i^2} \right) K \right] \vec{N} \tag{30}$$

where $H(c) = \iint_{R_o(C)} w(c,p)dp - \iint_{R_i(C)} w(c,p)dp$

C. Curve Evolution based on Normalized Cut Criterion

One of the popular ways to normalize the cost of a graph cut is the normalized cut framework. The graph theory version of this cost function is

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,V)} + \frac{cut(A,B)}{assoc(B,V)} \tag{31}$$

The continuous domain equivalent for this cost functions can be written as:

$$E = \frac{K}{B_i} + \frac{K}{B_o} \tag{32}$$

where $B_i = \iint_{R_i(C)} \iint_R w(p_1,p_2)dp_1dp_2$ and R corresponds to the full image domain. B_o is defined similarly. Calculation of the curve evolution equation is straight forward:

$$\frac{\partial C}{\partial t} = \left[\left(\frac{1}{B_o} + \frac{1}{B_i} \right) H(c) + \left(\frac{1}{B_o^2} - \frac{1}{B_i^2} \right) K \cdot Z(c) \right] \vec{N} \tag{33}$$

where $Z(c) = \iint_R w(c,p)dp$.

In this section, we have derived curve evolution equations for various pairwise similarity based cost functions, each of which could be useful for different kinds of applications. A possible future direction is an in depth comparison of these curve evolutions. This would help us understand the importance of the various terms that have emerged, namely L, K, Z, B_i and B_o.

VII. Experimental Results

In this section we present experimental results regarding the theory developed in the previous sections. Our main target domain is natural images, which are rich in color and texture. Unless otherwise stated, normalized maximum cut framework given in (21) and color features are used in the experiments for the segmentation of the images. We use opponent color space in our implementation. The reason for choosing color features is that it is easier to visualize and evaluate color segmentation than texture segmentation. We also demonstrate how utilizing Gabor texture features improves the results compared to using color features alone. For efficiency purposes, the distances between feature vectors are calculated using L_1 distance metric.

Our implementation of Level Set Methods uses the narrow band approach, which is explained in [20, Chapter 7]. The evolving curve C is embedded into a 2-D function $u(x,y)$, such that $C = \{(x,y)|u(x,y) = 0\}$. This function u is generated and evolved only on a narrow band around the curve instead of the full image domain for efficiency purposes. We select the size of the narrow band as 2 pixels wide and the size of the land mine area as 1 pixels wide at both sides of the curve. When the curve reaches the land mine area, narrow band is regenerated from the current curve and the values of u are recalculated on the new narrow band. Even though the specific choice of u mostly does not effect the curve evolution, a popular choice for u is the signed distance function, where each point in the narrow band is assigned a value based on their signed distance from the curve (negative if inside the curve, positive otherwise). Since we are using a very narrow narrow band, we use a different and simpler strategy for the reinitialization of u. We start with the curve and grow the narrow band layer by layer by using the 4-neighbors. First layer consists of the 4-neighbors of the curve points. The second layer is the 4-neighbors of the first layer, etc. Moving from one layer to the next, we increment (decrement) the value of u by 1. The convergence criterion for the curve evolution is also connected to the narrow band approach. Convergence is achieved if the narrow band is not reinitialized for n iterations. This means that the curve moved only within the narrow

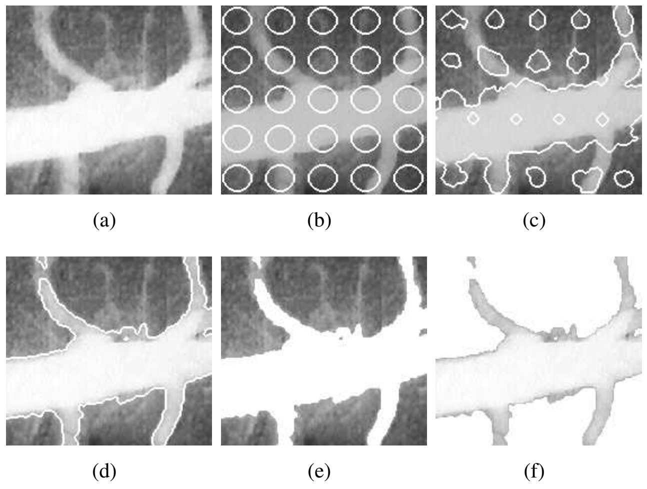


Fig. 8. Segmentation of an arterial tree image. a) Original image, b) curve initialization, c) curve is evolving, d) segmentation boundary, e) segmented background, f) segmented foreground.

band during these iterations. In our experiments, n is chosen between 30 and 50.

Fig. 8 shows segmentation of a gray scale intensity image of an arterial tree. Segmentation of this image of size 183×163 takes less than 5 seconds on a Pentium 4 2.0 GHertz computer using our unoptimized code written in C#.

Fig. 9 and 10 show foreground/background segmentation on variety of images. Image resolutions are reduced to 300×200 from their original sizes before segmentation. As can be seen from the figures, regions are precisely segmented despite the fact that we are using the fast scheme from Section IV. In these experiments and in the following ones, all parameters are fixed and images are segmented automatically without any manual intervention. Even though this kind of bi-partitioning shouldn't be thought of as a full segmentation, it has many applications in various fields ranging from content-based image retrieval to segmentation of biological or medical images.

Color features by themselves might not be able to segment natural images properly. These type of images are usually rich in texture and using texture features will improve segmentation results. Fig. 11 shows a comparison of segmentation results using only color features, only texture features and a combination of color and texture features on a bear image. Color features by themselves are not able to extract the boundaries due to the color changes within the head area. On the other hand, the texture of the fur helps finding the precise boundary if texture features are utilized. To calculate the texture features, we filter the image by Gabor filters [28] tuned to certain scales and orientations. In Fig. 11, features are calculated at 3 scales and 6 orientations. In Fig. 11(d), color and texture features are combined for the segmentation. We do this by creating a feature vector $[\alpha\vec{T}\ (1-\alpha)\vec{C}]^T$, where $\vec{T}$ is the texture feature vector normalized by the average of all the texture feature values and $\vec{C}$ is the color feature vector normalized similarly. α is the weighting between color and texture features, which is selected as 0.7 in Fig. 11(d).

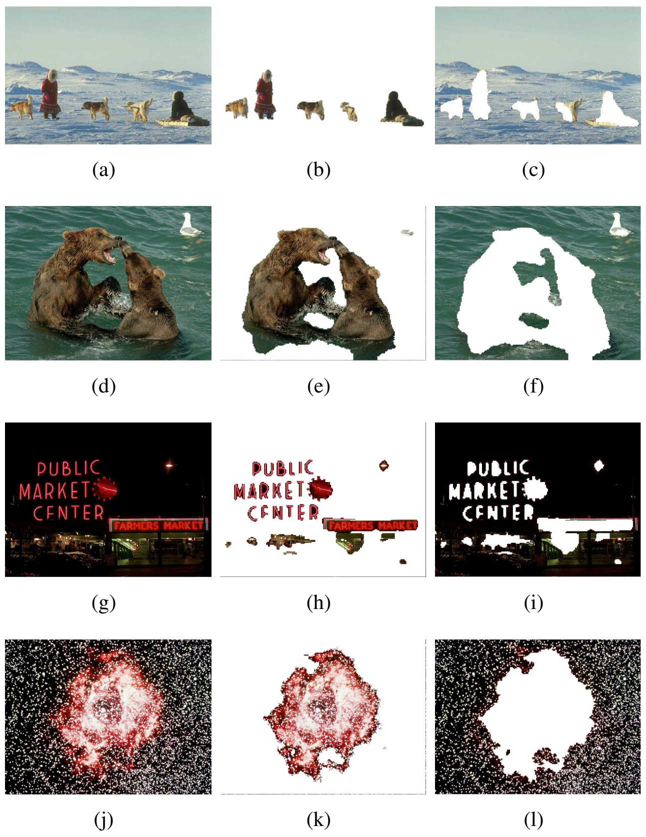


Fig. 9. Foreground/Background segmentation. First column shows the original images. Second and third columns correspond to foreground and background.

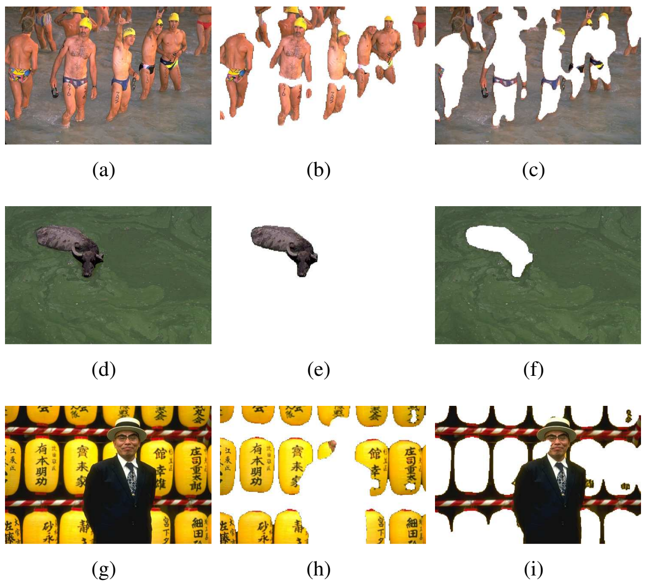


Fig. 10. Foreground/Background segmentation. First column shows the original images. Second and third columns correspond to foreground and background.

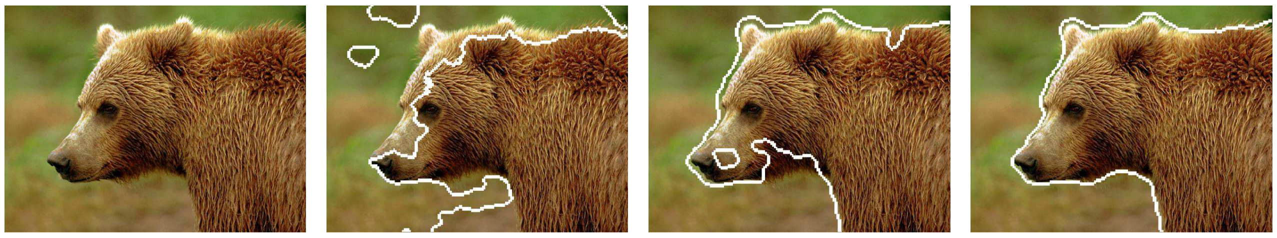


Fig. 11. Demonstration of improvements using texture feature vectors. a) Original image, b) color segmentation, c) texture segmentation using Gabor texture features at 3 scales and 6 orientations, d) segmentation using texture and color features. Texture features weighted 70% and color features 30%.

VIII. Discussions and Conclusions

A. Strategies for Integration of Edge Information

We have discussed region-based strategies for segmentation. It is often desirable to integrate the edge information to extract more precise boundaries. GPAC offers a flexible framework where edge information can be integrated in three different ways.

1) *Integrating edge information to the curve evolution framework:* ACM most commonly utilize an edge function $g = 1/(1 + |\nabla \hat{I}_\sigma|)$, where $\hat{I}_\sigma$ is the Gaussian smoothed image. This edge function is generated from the image through filtering and derivative operators. Another possibility is to create an edge function from an edge vector field $\vec{S}$ that is derived from the image [29]. By design, $\vec{S}$ would create a flow towards the edges. The general characteristic of an edge function is that it is a decreasing function of the edge strength. One way of integrating edge information is the following:

$$\frac{\partial C}{\partial t} = g\left(\iint_{R_o} w(c,p)dp - \iint_{R_i} w(c,p)dp\right)\vec{N} \\ -\gamma g\kappa\vec{N} + \alpha(\vec{S}\cdot\vec{N})\vec{N} \tag{34}$$

where α and γ are constants. The effect of multiplying with g is that the evolution of the curve will slow down or stop when the curve is aligned with an edge.

2) *Integrating edge information to the cost function:* Another way to integrate edges is by modifying the cost function for the boundary-normalized cut introduced in (25). Instead of just normalizing by the boundary length, we can also normalize by the boundary length weighted by the edge function (a similar normalization is also proposed in [16]).

$$E = \frac{\iint_{R_i(C)}\iint_{R_o(C)} w(p_1,p_2)dp_1dp_2}{\int_0^1 g(C)|C_q(q)|dq} \tag{35}$$

Let $L_2 = \int_0^1 g(C)|C_q(q)|dq$. The solution of this cost function is similar to the solution of (25). The only difference is that L is replaced by L_2 and the first variation of L is replaced by L_2'. First variation of L_2 has been derived in [1, Appendix B] and is equal to $\partial L_2/\partial t = \oint_C \left\langle C_t, (\nabla g\cdot\vec{N} + g\kappa)\vec{N}\right\rangle ds$. Based on these calculations, the corresponding evolution equation can be written as:

$$\frac{\partial C}{\partial t} = \frac{1}{L_2}\left(\iint_{R_o(C)} w(c,p)dp - \iint_{R_i(C)} w(c,p)dp\right)\vec{N} \\ -\frac{K}{L_2^2}\left(g\kappa + \nabla g\cdot\vec{N}\right)\vec{N} \tag{36}$$

3) *Integrating edge information to the (dis)similarity measure:* A third way of using edge information is by utilizing it when defining the similarities between pixels, $w(i,j)$. This approach can be easily applied to our framework considering that the curve evolution is independent of the (dis)similarity measure. In [30] it has been proposed that, if there are strong edges along a line connecting two pixels (intervening contour), these pixels probably belong to different regions and should be labeled as dissimilar. So, edge information can be integrated by reducing the pairwise similarity of such pixels.

B. Conclusion

In this paper we presented a generic region-based segmentation method, graph partitioning active contours (GPAC), using color and texture features. This method is based on defining a dissimilarity metric at the pixel level and finding the optimum partitioning of the image that maximizes the dissimilarities across the boundary. We have shown connections to graph partitioning methods (GPM) and derived curve evolution equations for various complex region-based segmentation cost functions. These segmentation cost functions include minimum and maximum cut frameworks and area and boundary normalized versions of these.

Our proposed method, GPAC, is based on pairwise dissimilarities between pixels. The method is quite flexible since the dissimilarity metric can be adapted to the application at hand or to the domain knowledge. This is different than other region based variational methods where analysis is done at the region level.

Unfortunately, direct computations of dissimilarities and region integrals in (21) bring unreasonable computational complexity and memory requirements. To address this issue, we proposed a fast method for implementing the curve evolution. This method is based on reducing the resolution of one dimension of the dissimilarity matrix W. Despite the approximation, we observe that precision of the segmentation is not lost.

We have shown promising experimental results for the bi-partitioning of the image into foreground and background regions. The experimental results support the theory we developed in this paper. We also discussed different strategies for integrating edges both from ACM and GPM point of views.

As future work, we plan to evaluate normalized maximum cut, various region-based active contour methods, and the methods we introduced in Sections VI and their GPM equivalents.

Acknowledgements

We would like to thank the reviewers for their constructive comments. We would also like to thank Dr. Charles Kenney and Prof. Shiv Chandrasekaran for fruitful discussions. This work is supported by the Office of Naval Research under grant ONR #N00014-04-1-0121, and the National Science Foundation award NSF ITR-0331697.

Appendix

In this section we calculate the first variation for the following functional with respect to t:

$$M = \oint_C \left\langle \vec{S}, \vec{N} \right\rangle ds \tag{37}$$

where $\nabla \cdot \vec{S} = G$ as defined in Section II. Let $\vec{S} = [S^1 \; S^2]^T$. M is then equal to:

$$M = \int_0^1 \left\langle \begin{bmatrix} S^1 \\ S^2 \end{bmatrix}, \|\vec{C}_p\|\vec{N} \right\rangle dp \tag{38}$$

where p is a parametrization of the closed curve $\vec{C}$. Remember that $\vec{C}_t = [x_t \; y_t]^T$, $\vec{C}_p = [x_p \; y_p]^T$ and $\|\vec{C}_p\|\vec{N} = [-y_p \; x_p]^T$. The first variation can be written as:

$$\begin{aligned}\frac{\partial M}{\partial t} &= \int_0^1 \left\langle \begin{bmatrix} \nabla S^1 \cdot \vec{C}_t \\ \nabla S^2 \cdot \vec{C}_t \end{bmatrix}, \begin{bmatrix} -y_p \\ x_p \end{bmatrix} \right\rangle dp \\ &+ \int_0^1 \left\langle \begin{bmatrix} S^1 \\ S^2 \end{bmatrix}, \begin{bmatrix} -y_{pt} \\ x_{pt} \end{bmatrix} \right\rangle dp\end{aligned}$$

Integrating by parts:

$$\begin{aligned}\frac{\partial M}{\partial t} &= \int_0^1 \left\langle \begin{bmatrix} \nabla S^1 \cdot \vec{C}_t \\ \nabla S^2 \cdot \vec{C}_t \end{bmatrix}, \begin{bmatrix} -y_p \\ x_p \end{bmatrix} \right\rangle dp \\ &- \int_0^1 \left\langle \begin{bmatrix} S^1_p \\ S^2_p \end{bmatrix}, \begin{bmatrix} -y_t \\ x_t \end{bmatrix} \right\rangle dp\end{aligned}$$

We rewrite the scalar product within the second integral:

$$\begin{aligned}\frac{\partial M}{\partial t} &= \int_0^1 \left\langle \begin{bmatrix} \nabla S^1 \cdot \vec{C}_t \\ \nabla S^2 \cdot \vec{C}_t \end{bmatrix}, \begin{bmatrix} -y_p \\ x_p \end{bmatrix} \right\rangle dp \\ &- \int_0^1 \left\langle \begin{bmatrix} \nabla S^2 \cdot \vec{C}_p \\ -\nabla S^1 \cdot \vec{C}_p \end{bmatrix}, \vec{C}_t \right\rangle dp\end{aligned}$$

After opening the scalar products and rearranging terms:

$$\frac{\partial M}{\partial t} = \int_0^1 \left\langle \begin{bmatrix} -y_p S^1_x + \cancel{x_p S^2_x} - \cancel{x_p S^2_x} - y_p S^2_y \\ -\cancel{y_p S^1_y} + x_p S^2_y + x_p S^1_x + \cancel{y_p S^1_y} \end{bmatrix}, \vec{C}_t \right\rangle dp$$

This is equal to:

$$\begin{aligned}\frac{\partial M}{\partial t} &= \int_0^1 \left\langle (S^1_x + S^2_y) \begin{bmatrix} -y_p \\ x_p \end{bmatrix}, \vec{C}_t \right\rangle dp \\ &= \int_0^1 \left\langle (\nabla \cdot \vec{S}) \|\vec{C}_p\| \vec{N}, \vec{C}_t \right\rangle dp \\ &= \oint_C \left\langle G\vec{N}, \vec{C}_t \right\rangle ds\end{aligned}$$

References

[1] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *International Journal of Computer Vision*, pp. 61–79, February 1997.

[2] N. Xu, R. Bansal, and N. Ahuja, "Object segmentation using graph cuts based active contours," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2003, pp. 46–53.

[3] Y. Boykov and V. Kolmogorov, "Computing geodesics and minimal surfaces via graph cuts," in *IEEE International Conference on Computer Vision (ICCV)*, October 2003, pp. 26–33.

[4] D. Mumford and J. Shah, "Boundary detection by minimizing functionals," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1985, pp. 22–6.

[5] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on Image Processing*, pp. 266–77, February 2001.

[6] L. D. Cohen, "On active contour models and balloons," *Computer Vision, Graphics, and Image Processing. Image Understanding*, 53(2):211–218, 1991.

[7] A. Tsai, *Curve Evolution and Estimation-Theoretic Techniques for Image Processing*, Ph.D. thesis, Harvard-MIT Division of Health Sciences and Technology, August 2000.

[8] A. Tsai, A. J. Yezzi, and A. S. Willsky, "Curve evolution implementation of the mumford-shah functional for image segmentation, denoising, interpolation, and magnification," *IEEE Transactions on Image Processing*, pp. 1169–86, August 2001.

[9] A. J. Yezzi, A. Tsai, and A. Willsky, "A statistical approach to snakes for bimodal and trimodal imagery," in *International Conference on Computer Vision (ICCV)*, 1999, pp. 898–903.

[10] N. Paragios and R. Deriche, "Geodesic active regions: a new framework to deal with frame partition problems in computer vision," *Journal of Visual Communication and Image Representation*, pp. 249–68, March 2002.

[11] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 888–905, August 2000.

[12] Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: theory and its application to image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1101–1113, Nov 1993.

[13] S. Sarkar and P. Soundararajan, "Supervised learning of large perceptual organization: graph spectral partitioning and learning automata," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 504–525, May 2000.

[14] S. Wang and J. M. Siskind, "Image segmentation with ratio cut," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 675–690, Jun 2003.

[15] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1222–1239, Nov 2001.

[16] I. H. Jermyn and H. Ishikawa, "Globally optimal regions and boundaries as minimum ratio weight cycles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1075–1088, Oct 2001.

[17] S. C. Zhu and A. Yuille, "Region competition: unifying snakes, region growing, and bayes/mdl for multiband image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 884–900, September 1996.

[18] A. Vasilevskiy and K. Siddiqi, "Flux maximizing geometric flows," in *IEEE International Conference on Computer Vision (ICCV)*, July 2001, pp. 7–14.

[19] B. Sumengen and B. S. Manjunath, "Category pruning in image databases using segmentation and distance maps," in *European Signal Processing Conference (EUSIPCO)*, September 2005.

[20] J. A. Sethian, *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, Cambridge University Press, 1999.

[21] I. J. Cox, S. B. Rao, and Y. Zhong, “Ratio regions: a technique for image segmentation,” in *International Conference on Pattern Recognition (ICPR)*, August 1996, pp. 557–564.

[22] P. Soundararajan and S. Sarkar, “An in-depth study of graph partitioning measures for perceptual organization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 642–660, June 2003.

[23] http://www.hid.ri.cmu.edu/Hid/software_ncutPublic.html

[24] E. Sharon, A. Brandt, and R. Basri, “Fast multiscale image segmentation,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2000, pp. 70–77.

[25] E. Sharon, A. Brandt, and R. Basri, “Segmentation and boundary detection using multiscale intensity measurements,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, December 2001, pp. 469–476.

[26] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, “Spectral grouping using the nystrom method,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, pp. 214–225, February 2004.

[27] G. Sapiro, *Geometric Partial Differential Equations and Image Analysis*, Cambridge University Press, January 2001.

[28] B. S. Manjunath and W. Y. Ma, “Texture features for browsing and retrieval of image data,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 837–42, August 1996.

[29] B. Sumengen and B. S. Manjunath, “Edgeflow-driven variational image segmentation: Theory and performance evaluation,” Tech. Rep., August 2005.

[30] J. Malik, S. Belongie, J. Shi, and T. Leung, “Textons, contours and regions: cue integration in image segmentation,” in *IEEE International Conference on Computer Vision (ICCV)*, September 1999, pp. 918–925.