Efficient Segmentation Using Feature-based Graph Partitioning Active Contours

Filiz Bunyak, Kannappan Palaniappan
Department of Computer Science, University of Missouri-Columbia, Columbia, MO, USA

Abstract

Graph partitioning active contours (GPAC) is a recently introduced approach that elegantly embeds the graph-based image segmentation problem within a continuous optimization framework. GPAC can be used within parametric snake-based or implicit level set-based active contour continuous paradigms for image partitioning. However, GPAC similar to many other graph-based approaches has quadratic memory requirements which severely limits the scalability of the algorithm to practical problem domains. An $N \times N$ image requires $O(N^4)$ computation and memory to create and store the full graph of pixel inter-relationships even before the start of the contour optimization process. For example, an 1024x1024 grayscale image needs over one terabyte of memory. Approximations using tile/block-based or superpixel-based multiscale grouping of the pixels reduces this complexity by trading off accuracy. This paper describes a new algorithm that implements the exact GPAC algorithm using a constant memory requirement of a few kilobytes, independent of image size.

1. Introduction

Image segmentation using active contours and graph theoretic methods have been extensively explored. The relationship between and the combination of these two powerful techniques are current areas of active research. Active contours for region-based image segmentation, introduced in [9, 10, 18, 20, 28] has increasingly focused on the use of novel statistical region-based image energy functions [12]. Graph- or graph-cut based approaches for image segmentation have the underlying common theme of formulating the process as a weighted graph with each vertex corresponding to an image pixel or region and edges corresponding to similarity/dissimilarity between two corresponding pixels/regions [4, 11, 15, 16, 22, 25]. To discover an image segmentation this pixel graph is partitioned by minimizing some cost function. Graph-based approaches have strong connections to active contours and level sets. The segmentation energies optimized by graph-cuts combine boundary regularization with region-based properties in the same fashion as Mumford-Shah style functionals [4]. Segmentation methods combining active contours and graph cuts have led to approaches with encouraging results [5, 26].
Sumengen and Manjunath introduced [23] and extended [2] a new curve evolution framework called graph partitioning active contours (GPAC) that combines the advantages of pairwise pixel-similarity based cost functions (e.g. embedding heterogeneous information) with the flexibility of variational methods into a general and well-defined framework for image segmentation [2]. But high computational complexity prevents the direct application of GPAC even to common image sizes. Some *approximations* of the GPAC algorithm have been proposed in [23] and [2] to alleviate this problem by partitioning the input image into regular blocks or "superpixels" and calculating the dissimilarities at a coarse level. Block and superpixel-based approaches are approximations of the original GPAC.
In this paper, we propose FastGPAC, a novel implementation of the original GPAC algorithm, that dramatically reduces computational and memory requirements without the need to partition the image into blocks or superpixels. Segmentation scalability has enabled high-throughput, high resolution image analysis [13] and can be applied to tracking applications [7, 8]. In Section 2, we give an overview of the graph partitioning active contours (GPAC). In Section 3, we present our novel FastGPAC algorithm. In Section 4, we give a comparative analysis of the GPAC and FastGPAC algorithms. In Section 5, we describe implementation of FastGPAC using level sets. In Section 6, we discuss relationship between FastGPAC and some region-based level set approaches and explore impact of robust norms on segmentation. Results and conclusions are presented in Sections 7 and 8.

2. Graph Partitioning Active Contours

In [23] Sumengen and Manjunath introduce a new curve evolution framework called graph partitioning active contours (GPAC). This variational framework is based on pairwise similarities or dissimilarities between points and across-region cuts. In order to maximize the dissimilarities between regions, a variational cost function that is minimized for pairwise dissimilarities can be formulated as:

$$E_{CR}(\mathcal{C}) = - \iint_{p_2 \in R_i(\mathcal{C})} \iint_{p_1 \in R_o(\mathcal{C})} w(p_1, p_2) dp_1 dp_2 \quad (1)$$

2009 IEEE 12th International Conference on Computer Vision (ICCV)
978-1-4244-4419-9/09/$25.00 ©2009 IEEE

where $\mathcal{C}$ is a curve, $R_i(\mathcal{C})$ and $R_o(\mathcal{C})$ are the interior and exterior regions of $\mathcal{C}$, p_1 and p_2 are points such that $p_1 \in R_o$, $p_2 \in R_i$, and $w(p_1, p_2)$ is a dissimilarity measure between points p_1 and p_2. Double integrals reflect the integration over a 2D region defined by $R_r(x, y)$. Without any loss of generality $w(p_1, p_2)$ can be a similarity measure. In [2], Bertelli *et al.* reformulate this across-region cuts from [23], in terms of pairwise dissimilarity within the regions.

$$E_{WR}(\mathcal{C}) = \iint\limits_{p_2 \in R_i(\mathcal{C})} \iint\limits_{p_1 \in R_i(\mathcal{C})} w(p_1, p_2) dp_1 dp_2 + \iint\limits_{p_2 \in R_o(\mathcal{C})} \iint\limits_{p_1 \in R_o(\mathcal{C})} w(p_1, p_2) dp_1 dp_2 \tag{2}$$

As dissimilarity measure $w()$ in [23] and [1] L_p-norms L_1 and L_2 are used respectively on color features, but in the general case GPAC can use more complex dissimilarity measures that integrate spatial distance of pixels and domain knowledge as below [23]:

$$w(p_1, p_2) = \underbrace{||\vec{F}(p_1) - \vec{F}(p_2)||}_{\text{feature distance}} + \underbrace{\alpha ||p_1 - p_2||}_{\text{spatial distance}} + \underbrace{\beta ||m(p_1) - m(p_2)||}_{\text{domain knowledge}} \tag{3}$$

Theorem 2.1: Minimization of $E_{CR}(\mathcal{C})$ with respect to the curve $\mathcal{C}$ results in partitioning of the image which maximizes the dissimilarity between regions $R_i(\mathcal{C})$ and $R_o(\mathcal{C})$. The curve evolution equation that corresponds to the steepest descent minimization of Eq.1 is derived as [23]:

$$\frac{\partial C}{\partial t} = \underbrace{\left(\iint\limits_{R_o(\mathcal{C})} w(p_c, p) dp - \iint\limits_{R_i(\mathcal{C})} w(p_c, p) dp \right)}_{\text{Region Variability}} \vec{N} \tag{4}$$

where p_c is a point on the curve $\mathcal{C}$ and $\vec{N}$ is the outward normal of the curve. For proof see [23].

Algorithm 1 Original GPAC

```
Input  :  Image $\mathbf{I}(\boldsymbol{x})$, initial curve $\mathcal{C}$
Output :  Converged curve $\mathcal{C}$ and associated regions $R_i(\mathcal{C}), R_o(\mathcal{C})$
1: Initial Look-up table (LUT) computation:
   $\mathbf{W} \leftarrow$ pixel-to-pixel dissimilarity matrix // $\frac{NM \times NM}{2}$ computations
   or
   $\mathbf{W}' \leftarrow$ pixel-to-block dissimilarity matrix // $\frac{NM \times nm}{2}$ computations
2: while $((iterations < K) \& (curve\_convergence \neq true))$ do
3:    for each point $p_c(x, y) \in \mathcal{C}$ do
4:       Region Variability Term (RV) Calculation:
         $S_i(p_c) \leftarrow \sum_{p_i \in R_i} w(p_c, p_i)$; $S_o(p_c) \leftarrow \sum_{p_o \in R_o} w(p_c, p_o)$
         $RV(p_c) = S_o(p_c) - S_i(p_c)$
5:       Curve Evolution:
         Compute other terms such as $\vec{N}$ and $\mathcal{K}$ etc.
         Compute $\frac{\partial \mathcal{C}}{\partial t}$ and Update $\mathcal{C}, R_i, R_o$
6:    end for
7: end while
```

We refer to the term inside the brackets in Eq. 4 as *region variability* (RV). Every point p_c on the curve is compared to all the points in the interior and exterior of the curve and depending on the outcome, the curve is expanded or shrunk in the normal direction. A regularization term ($\mathcal{K}$) is added to the curve evolution expression to ensure curve smoothness. The curve evolution function in Eq. 4 requires computation of two integrals $\iint_{R_i(\mathcal{C})} w(p_c, p)dp$ and $\iint_{R_o(\mathcal{C})} w(p_c, p)dp$ that measure the (dis)similarity of each contour point p_c to the interior and exterior regions of the curve $R_i(\mathcal{C})$ and $R_o(\mathcal{C})$ respectively. These integrals are discretized into sums and the region variability term in Eq. 4 is discretely computed as:

$$RV(p_c) = \sum_{p \in R_o(\mathcal{C})} w(p_c, p) - \sum_{p \in R_i(\mathcal{C})} w(p_c, p) \tag{5}$$

The computational bottleneck in GPAC algorithm is the calculations of these sums. To speed-up calculation, in [23] pixel-to-pixel dissimilarities are pre-computed and kept in memory as a look-up table. For an image of size $N \times M$, this results in a symmetric dissimilarity matrix $\mathbf{W}$ with NM rows and NM columns, where an element $W(i, j)$ is $w(p_i, p_j)$, dissimilarity of pixels p_i and p_j. Number of such dissimilarities kept in this method is $N^2M^2/2$. Since even for small images $\mathbf{W}$ becomes quite large and hard to fit in memory (1TB for a $1K \times 1K$ image and 2 bytes per $w(p_1, p_2)$), approximations are proposed in [23] and in [2] by partitioning the input image into blocks or into "superpixels" respectively. In [23], the input image is divided into $n \times m$ equal size tiles T_i, and $\mathbf{W}$ is approximated by an $NM \times nm$ matrix $\mathbf{W}'$, where an element $W'(i, j)$ corresponds to the dissimilarity of pixel p_i to subregion/tile T_j of the image. The process of curve evolution, in the original GPAC framework [23] is summarized in Alg. 1.

3. Fast Graph Partitioning Active Contours (FastGPAC)

In this paper, we propose a FastGPAC implementation method for cases where the (dis)similarity measure $w(p_1, p_2)$ does not incorporate spatial distance between points p_1 and p_2 ($\alpha = 0$ in Eq. 3). Typically energy functions of active contour methods do not use spatial distance between points, since spatial coherence is ensured using an area regularization term. The proposed method (Alg. 2) speeds up the integral/sum computations in GPAC by maintaining two additional data structures, histograms $\mathbf{h}_i$ and $\mathbf{h}_o$ for $R_i(\mathcal{C})$, $R_o(\mathcal{C})$ interior and exterior regions of the curve $\mathcal{C}$. When the point to point (dis)similarity measure $w(p_1, p_2)$ does not incorporate spatial distance between p_1 and p_2, $w(p_1, p_2)$ can be rewritten as:

$$w(p_1, p_2) \equiv D(F(p_1), F(p_2)) \tag{6}$$

where $F(p)$ is a feature extracted at point $p(x, y)$, and D is a (dis)similarity measure defined on F (i.e. for absolute grayscale intensity difference, $w(p_1, p_2) = |I(p_1) - I(p_2)|$, feature $F(p)$ is grayscale intensity $I(p)$ and the dissimilarity measure D is L_1 metric.)

Theorem 3.1 (GPAC Region Sum Theorem): For cases where the (dis)similarity measure $w(p_1, p_2)$ does not incorporate spatial distance between points p_1 and p_2, the 2D regional sums, $\sum_{p \in R_r} w(p_c, p)$ can be reduced to 1D sums independent of the spatial size or shape of the regions $R_r(\mathcal{C})$.

$$\sum_{p \in R_r} w(p_c, p) \equiv \sum_{j=0}^{L-1} h_r(j) \times D(F(p_c), j) \tag{7}$$

where $\mathbf{h}_r$ is the histogram of the feature F in the r^{th} region R_r, $D()$ is a (dis)similarity measure, L is number of bins in $\mathbf{h}$, and $h_r(j) = \sum_{p \in R \wedge F(p)=j} 1$ is the j^{th} bin of $\mathbf{h}_r$ corresponding to the number of points $p \in R_r$ whose feature $F(p) = j$. In the more general case, $D(F(p_c), j)$ is replaced by $D(F(p_c), C(j))$ where $C(j)$ is a representative value for the histogram bin $h(j)$.

Proof: This equality is derived by grouping the points p into feature class bins $F(p) = j$, and by separating the original sum into two nested sums as follows:

$$\begin{aligned}
\sum_{p \in R_r} w(p_c, p) &\equiv \sum_{p \in R_r} D(F(p_c), F(p)) \qquad (8)\\
&= \sum_{j=0}^{L-1} \sum_{p \in R_r \wedge F(p)=j} D(F(p_c), F(p))\\
&= \sum_{j=0}^{L-1} \sum_{p \in R_r \wedge F(p)=j} \{D(F(p_c), j) \times 1\}\\
&= \sum_{j=0}^{L-1} D(F(p_c), j) \times \underbrace{\sum_{p \in R_r \wedge F(p)=j} 1}_{h(j)}\\
&= \sum_{j=0}^{L-1} D(F(p_c), j) \times h(j)
\end{aligned}$$

In this case the discretized region variability term of Eq. 4 transforms into:

$$\begin{aligned}
&\sum_{p \in R_o(\mathcal{C})} w(p_c, p) - \sum_{p \in R_i(\mathcal{C})} w(p_c, p)\\
&= \sum_{j=0}^{L-1} h_o(j) \times D(F(p_c), j) - \sum_{j=0}^{L-1} h_i(j) \times D(F(p_c), j)\\
&= \sum_{j=0}^{L-1} [h_o(j) - h_i(j)] \times D(F(p_c), j)
\end{aligned}$$

Proposed curve evolution process is described in Alg. 2.

Algorithm 2 Fast GPAC

```
Input  : Image or feature $F(\boldsymbol{x})$, Initial curve $\mathcal{C}$
Output : Converged curve $\mathcal{C}$ and associated regions $R_i(\mathcal{C}), R_o(\mathcal{C})$
1: Initial histogram computation:
   $\mathbf{h}_i \leftarrow Histogram(R_i)$
   $\mathbf{h}_o \leftarrow Histogram(R_o)$

2: while $((iterations < K)\&(curve\_convergence \neq true))$ do
3:   $R_i^- = R_i$; $R_o^- = R_o$
4:   for each point $p_c(x, y) \in \mathcal{C}$ do
5:     Region Variability Term (RV) Calculation:
       $RV(p_c) \leftarrow \sum_{j=0}^{L-1} [h_o(j) - h_i(j)] \times D(F(p_c), j)$
6:     Curve Evolution:
       Compute other terms such as $\mathcal{K}$, $\vec{N}$
       Compute $\frac{\partial \mathcal{C}}{\partial t}$ and Update $\mathcal{C}, R_i, R_o$
7:   end for
8:   Histogram Update:
     $R_{io} \leftarrow R_i^- \cap R_o$; $R_{oi} \leftarrow R_o^- \cap R_i$
     Update $\mathbf{h}_i$ // Add pixels in $R_{oi}$, subtract pixels in $R_{io}$
     Update $\mathbf{h}_o$ // Add pixels in $R_{io}$, subtract pixels in $R_{oi}$
9: end while
```

4. Complexity Analysis

Complexity analysis will focus on using the across-region cut formulation in Eq. 1 and the histogram sum decomposition described in Section 3 in order to directly compare computational cost with [23].

4.1. Storage Requirements

Original GPAC: In the original GPAC [23], (dis)similarity between any pairing points in space $\mathbf{W}$ or between any point and tile in space $\mathbf{W}'$, are precomputed once and reused for the rest of the curve evolution as a look-up table. The storage requirements for $\mathbf{W}$ and $\mathbf{W}'$ are $(MN \times MN)$ and $(MN \times mn)$ respectively where MN is the image size and mn is the number of tiles.

FastGPAC: The proposed method maintains two histograms $\mathbf{h}_i$ and $\mathbf{h}_o$, for $R_i(\mathcal{C})$ and $R_o(\mathcal{C})$, interior and extrerior regions of the curve $\mathcal{C}$. For scalar features (i.e. intensity) storage requirement is $2 \times L$, where L is the number of bins in each histogram. For non-scalar features (i.e. multi-channel images, texture measures etc.) storage requirement is $2 \times BL$ for separable cases (where B 1-dimensional histograms are used) and $2 \times L^B$ for un-separable cases (where single B-dimensional histograms are used), where B is the feature vector size or number of channels (Table 1).

4.2. Computational Cost

Original GPAC: Total computational cost of the Original GPAC approach can be separated into two as initial cost and operation cost.

Initial cost is the cost associated with the distance computations to form the lookup tables $\mathbf{W}$ or $\mathbf{W}'$ (for pixel-to-

	Original GPAC			FastGPAC		
Step	**Alg. 1**	**Pixel − to − Pixel**	**Pixel − to − Block**	**Alg. 2**	**Non − separable**	**Separable**
MEMORY REQ.		$\frac{1}{2}M^2N^2$ (for **W**)	$\frac{1}{2}MN \times mn$ (for **W'**)		$2L^B$ (for $\mathbf{h_i}, \mathbf{h_o}$)	$2BL$ (for $\mathbf{h_i}, \mathbf{h_o}$)
1. Initial Cost	1	$B \times \frac{1}{2}M^2N^2$	$B \times \frac{1}{2}MN \times mn$	1	$2 \times B \times MN$	$2 \times B \times MN$
2. Operation Cost (K iterations)	2-7	$K \times (MN \times \mathcal{C}_L + n_b \times \mathcal{C}_L)$	$K \times (mn \times \mathcal{C}_L + n_b \times \mathcal{C}_L)$	2-9	$K \times (L^B \mathcal{C}_L + n_b \times \mathcal{C}_L)$	$K \times (BL\mathcal{C}_L + n_b \times \mathcal{C}_L)$
2.1 Integral Computation	4	$K \times (MN \times \mathcal{C}_L)$	$K \times (mn \times \mathcal{C}_L)$	5	$K \times (L^B \mathcal{C}_L)$	$K \times (BL\mathcal{C}_L)$
2.2 Curve Evolution	5	$K \times (n_b \times \mathcal{C}_L)$	$K \times (n_b \times \mathcal{C}_L)$	6	$K \times (n_b \times \mathcal{C}_L)$	$K \times (n_b \times \mathcal{C}_L)$
3. Update Cost	-	-	-	8	$2B\Delta R$	$2B\Delta R$
TOTAL COST		$B \times \frac{1}{2}M^2N^2 + K \times (MN \times \mathcal{C}_L + n_b \times \mathcal{C}_L)$	$B \times \frac{1}{2}MN \times mn + K \times (mn \times \mathcal{C}_L + n_b \times \mathcal{C}_L)$		$2 \times B \times MN + K \times (L^B\mathcal{C}_L + n_b \times \mathcal{C}_L) + 2B\Delta R$	$2 \times B \times MN + K \times (BL\mathcal{C}_L + n_b \times \mathcal{C}_L) + 2B\Delta R$

Table 1: Memory requirements and computational complexity for different steps of GPAC and FastGPAC implementations. MN: image size, L: number of bins in the histogram, B: number of channels/bands or feature vector size, K: number of iterations, $\mathcal{C}_L$: curve length.

pixel or pixel-to-block respectively) (Alg. 1 Step 1). It is $O(\frac{1}{2}MN \times MN)$ for pixel-to-pixel GPAC and $O(\frac{1}{2}MN \times mn)$ for pixel-to-block approximate block GPAC, where MN is image size and mn is number of blocks/tiles.
Operation cost is the cost associated with curve update for points $p_c \in \mathcal{C}$. Operation cost for a single iteration includes integral calculations and curve evolution (Alg. 1 Steps 3-6). Complexity of integral computation for all the points $p_c \in \mathcal{C}$ is $O(MN \times \mathcal{C}_L)$ for pixel-to-pixel GPAC case and $O(mn \times \mathcal{C}_L)$ for pixel-to-block approximate block GPAC case, where $\mathcal{C}_L$ is the curve length. Complexity of curve evolution which includes computation of the additional terms such as curvature $\mathcal{K}$, normal vector $\vec{N}$, etc., speed function $\frac{\partial \mathcal{C}}{\partial t}$, curve and region updates is $O(n_b \times \mathcal{C}_L)$ for both GPAC and approximate block GPAC, where n_b is the width of the narrow band around $\mathcal{C}$.
Itemized cost analysis for pixel-to-pixel and pixel-to-block GPACs are given in columns 3&4 of Table 1 respectively. If we assume that the curve length $\mathcal{C}_L$ is $O(NM)$ then, for $M = N$ and $m = n$, computational complexities become:

$$Cost_{GPAC} = \underset{Initial}{O(N^4)} + \underset{Operation}{O(N^4)} \quad (9)$$

$$Cost_{BlockGPAC} = \underset{Initial}{O(n^2N^2)} + \underset{Operation}{O(n^2N^2)} \quad (10)$$

FastGPAC: Total computational cost of the FastGPAC approach can be separated into three as: (1) initial cost, (2) operation cost, and (3) update cost.
Initial cost is the cost associated with the initial computation of the histograms $\mathbf{h}_i$ and $\mathbf{h}_o$ (Alg.:2, step 1). Its complexity is $O(MN)$ for an $M \times N$ image.
Operation cost is the cost associated with curve update. Operation cost for a single iteration includes integral calculations and curve evolution for points $p_c \in \mathcal{C}$ (Alg.:2, steps 4-7). Complexity of integral computation for points $p_c \in \mathcal{C}$ is $O(BL \times \mathcal{C}_L)$ for separable cases and $O(L^B \times \mathcal{C}_L)$ for non-separable cases, where L is number of bins in the histograms, B is number of channels/bands or feature vector size, and $\mathcal{C}_L$ is the curve length. Complexity of curve evolution which includes computation of the additional terms such as curvature $\mathcal{K}$, normal vector $\vec{N}$, etc., speed function $\frac{\partial \mathcal{C}}{\partial t}$, curve and region updates is $O(n_b \times \mathcal{C}_L)$ where n_b is the width of the narrow band around $\mathcal{C}$.
Update cost is the cost associated with update of the histograms $\mathbf{h}_i$ and $\mathbf{h}_o$ (Alg.:2, step 8). Complexity is $O(2 \times B \times \Delta R_k)$ where the changed region $\Delta R_k = R_{io} + R_{oi}$ is sum of pixels that changed from foreground to background and from background to foreground during iteration k. If the total number of pixels that changed from one class to the other is $\Delta R = \sum_{k=1}^{K} \Delta R_k$, the the total cost for K iterations is $O(\Delta R)$.
Itemized cost analysis for non-separable and separable FastGPACs are given in columns 6 & 7 of Table 1 respectively. If we assume that the curve length $\mathcal{C}_L$ is $O(NM)$ then, for $M = N$ and $m = n$, computational complexities become:

$$Cost_{FastGPAC_1} = \underset{Initial}{O(N^2)} + \underset{Operation}{O(BLN^2)} + \underset{Update}{O(N^2)} \quad (11)$$

$$Cost_{FastGPAC_2} = \underset{Initial}{O(N^2)} + \underset{Operation}{O(L^BN^2)} + \underset{Update}{O(N^2)} \quad (12)$$

for separable and non-separable cases where the histograms used for B-channel inputs are B separate 1-D histograms or single B-D histograms respectively.

5. Level set Implementation

In order to make a fair comparison regardless of coding optimizations, GPAC code by Luca Bertelli [1] is used as a base for the FastGPAC code. This code has some differences from the implementation described in [23] where the variational cost is formulated as across-region cuts (Eq.1) and partitioning of an image is defined as direct minimization with respect to the explicit representation of the curve $\mathcal{C}$ (see Eq. 4 in Section 2). In Bertelli's code [1] and our code variational cost is formulated as within region dissimilarity Eq. 2 instead of across-region cuts Eq. 1 and the curve $\mathcal{C}$

evolution is done implicitly using level sets.
In level set-based active contour methods, a curve $\mathcal{C}$ is represented implicitly via a Lipschitz function $\phi : \Omega \mapsto \mathbb{R}$ by $\mathcal{C} = \{(x, y) | \phi(x, y) = 0\}$, and the evolution of the curve is given by the zero-level curve of the function $\phi(t, x, y)$ [9]. The function ϕ is positive inside and negative outside of the curve $\mathcal{C}$. Heaviside function ($\mathcal{H}(\phi) = 1$ if $\phi > 0$, 0 elsewhere) is used as indicator function for the points inside and outside regions. In [2], Bertelli et al. prove that minimizing Eq. 1 is equivalent to minimizing Eq. 13:

$$E = \iint_\Omega \iint_\Omega w(p_1, p_2)\mathcal{H}(\phi(p_1))\mathcal{H}(\phi(p_2))dp_1 dp_2 \\ + \iint_\Omega \iint_\Omega w(p_1, p_2)\Big(1 - \mathcal{H}(\phi(p_1))\Big)\Big(1 - \mathcal{H}(\phi(p_2))\Big)dp_1 dp_2 \tag{13}$$

Steepest descent minimization of Eq. 13 leads to the curve evolution equation Eq. 14 which is shown to be equivalent to Eq. 4 (see Section 2).

$$\frac{\partial\phi(p_2)}{\partial t} = \delta(\phi(p_2))[\iint_\Omega w(p_1, p_2)\Big(1 - \mathcal{H}(\phi(p_1))\Big)dp_1 \\ - \iint_\Omega w(p_1, p_2)\mathcal{H}(\phi(p_1))] \tag{14}$$

Using Theorem 3.1 (GPAC Region Sum Theorem) FastGPAC transforms the discretized curve evolution function corresponding to Eq. 14 into:

$$\frac{\Delta\phi(p_2)}{\Delta t} = \delta_\epsilon(\phi(p_2))\Big[\sum_{p_1 \in R_o(\mathcal{C})} w(p_2, p_1) - \sum_{p_1 \in R_i(\mathcal{C})} w(p_2, p_1)\Big] \\ = \delta_\epsilon(\phi(p_2))\Big[\sum_{j=0}^{L-1} [h_o(j) - h_i(j)] \times D(F(p_2), j)\Big] \tag{15}$$

where δ_ϵ is the discretized delta function, $\mathbf{h_i}$ and $\mathbf{h_o}$ are histograms of feature F for inside, outside regions $R_i(\mathcal{C})$ and $R_o(\mathcal{C})$, and D is the dissimilarity measure. Additional geometric properties or constraints can be introduced into curve evolution using normalization factors α, β [2,23], and weights $\lambda_1, \lambda_2, \mu$. Smoothness of the curve is ensured with a regularization term. Complete FastGPAC curve evolution then becomes:

$$\frac{\Delta\phi(p_2)}{\Delta t} = \delta_\epsilon(\phi(p_2))[\sum_{j=0}^{L-1} [\lambda_2\beta h_o(j) - \lambda_1\alpha h_i(j)] \\ \times D(F(p_2), j) + \mu \operatorname{div}(\frac{\nabla\phi(p_2)}{|\nabla\phi(p_2)|})] \tag{16}$$

To further speed-up calculations of the sums a $1 \times L$ look-up-table T_k is pre-computed at each iteration where

$$T_k(i) = \sum_{j=0}^{L-1} [h_o(j) - h_i(j)] \times D(i, j) \tag{17}$$

which denote the region variability of the feature value i to the regions R_o and R_i, whose feature distributions are represented by the histograms $\mathbf{h}_o, \mathbf{h}_i$. $RV(p_2, k)$ is then obtained by using T_k as: $RV(p_2, k) \leftarrow T_k(F(p_2))$
Use of look-up-table reduces $N \times M$ point to region dissimilarity computation to L feature values to regions dissimilarity computation and $N \times M$ look-up-table access, where $N \times M$ is image size and L is number of histogram bins.

6. Discussion

In this section, we explore the relationship of FastGPAC segmentation to other region-based active contour methods. A widely used method is the Chan & Vese region-based active contour model. In the Chan and Vese model presented in [9] segmentation is done using the contour evolution function:

$$\frac{\partial\phi}{\partial t} = \delta(\phi)\Big[\underbrace{\nu \operatorname{div}\frac{\nabla\phi}{|\nabla\phi|}}_{\text{Regularization}}\underbrace{-\mu_1(u(\mathbf{x}) - c_i)^2 + \mu_2(u(\mathbf{x}) - c_o)^2}_{\text{Region Variability}}\Big] \tag{18}$$

where $\phi(\mathbf{y})$ is the 2D level set function, $u(\mathbf{y})$ is the intensity image, and c_i, c_o are the mean gray values of the two phases used to define the segmentation. The first curve *regularization* term aims at minimizing the length of the boundary separating these two phases; the last two terms *region variability* (RV), aim at maximizing the gray value homogeneity of the two phases; μ_1, μ_2 and ν are adjustable constants associated with this functional. Describing regions using only mean values results in poor representation and blending effects. For better description other parametric models have been introduced such as the Gaussian model in [21] by Rousson and Deriche and mixture distributions of Gaussian components in [19] by Paragios and Deriche. Parametric methods can impose serious restrictions on the input data statistics and applications since the performance can be negatively affected when the parametric model does not match the underlying data distribution [14]. Nonparametric statistical methods offer robustness and better modeling capabilities, and have been used in different active contour frameworks such as [6, 14, 24, 27] and in FastGPAC. Two recent active contour approaches that have some similarities to FastGPAC are Brox & Weickert [6] and Weiler et. al. [24] that describe nonparametric level set methods along with histogram based region descriptors. They use a reformulation of the energy functional in Zhu and Yuille [28], which results in the contour evolution function:

$$\frac{\partial\phi}{\partial t} = \delta(\phi)\Big[\nu \operatorname{div}\big(\frac{\nabla\phi}{|\nabla\phi|}\big) + \log\frac{p_1}{p_2}\Big] \tag{19}$$

The first term is the regularization component and the second is the region variability term where p_1, p_2 are pixel probabilities assigned to inner and outer regions which has been extended to multiple regions in [6] and to motion pattern segmentation in [24]. Regions inside and outside of the contour are described using normalized histograms $\mathbf{hn}_{1,b}$ and $\mathbf{hn}_{2,b}$ for each feature channel b [24]. Region assignment probability is then computed as: $p_i(x) = \prod_{b=1}^{B} hn_{i,b}(j^*)$ where $hn_{i,b}(j^*)$ is the histogram entry that corresponds to the bin index j^* indicated by b^{th} channel of the feature vector $\vec{F}(\mathbf{x})$.

Well-known problems with histogram-based segmentation include sensitivity to size, number, and distribution of the histogram bins. Single bin histogram differences [14, 24, 27], lack of summation over the entire probability density function and variable quantization significantly effect performance. In FastGPAC dissimilarity to the foreground/background distributions is a weighted aggregation of all bin-to-bin differences. This makes the dissimilarity computation robust to size and number of the histogram bins and to non-uniformities of the region features distributions. This approach is also flexible since desired weighting scheme and influence region can be obtained by changing the distance function. The level set formulation in [24] can be considered a special case of FastGPAC where the distance/weighting function D in Eq. 16 is an impulse function and where the contour is evolved by maximizing similarity instead of minimizing dissimilarity as in FastGPAC.

The most important advantages of GPAC and FastGPAC are their region description capabilities enabled by use of region histograms and their flexibility enabled by possibility of using different image features and dissimilarity measures. Of particular interest are robust norms [3] such as Geman-McClure, Tukey's biweight, or saturated L_1 and L_2 distances. Robust norms used with region histograms are robust to outliers. Section 7 shows sample results of FastGPAC using robust norms.

7. Experimental Results

In this section we present comparison of the original GPAC [1] and our FastGPAC results in terms of speed, memory usage and segmentation mask similarity. Both programs are written as a mixture of Matlab and C. For Figure 1 and Tables 2, 3, both GPAC and FastGPAC convert the input images from RGB into YC_bC_r color space and scale the luma component Y into [0-100], chroma components C_b and C_r into [0-200] ranges. The original GPAC program uses L_2 (Euclidean distance) distance metric, the FastGPAC program uses L_2-squared distance metric, because of its separability property. Parameters used in the presented test cases are $\epsilon = 0.001$, $\mu = 4000$, $\lambda_1 = \lambda_2 = 1$ for the original GPAC and $\epsilon = 0.001$, $\mu = 4000$, $\lambda_1 = \lambda_2 = 1/200$ for FastGPAC (λ_1 and λ_2 values are different to compensate for L_2 versus L_2-square distance metric). Both programs were tested on Intel Xeon 5140 dual core with dual processors and 10GB RAM.

Segmentation results obtained for sample test cases from original GPAC and FastGPAC codes and the differences are shown in Figure 1. Quantitative comparison of the segmentation masks and speed-ups obtained with FastGPAC are given in Table 2. For the last three images on the table, comparison statistics are not available (NA) since for these larger images the GPAC code could not run on the current system because of the large memory requirements of the original GPAC algorithm. FastGPAC timing results for these images can be found in Table 3. These results illustrate that even for quite small images there is a significant timing difference between original GPAC and FastGPAC algorithms, which increases dramatically for larger images.

Segmentation similarity is measured with $F - measure$ defined as: $\text{F-measure} = \frac{2\times(\text{precision} \times \text{recall})}{\text{precision} + \text{recall}}$ and $\text{recall} = \frac{|G \cap F|}{|G|}$, $\text{precision} = \frac{|G \cap F|}{|F|}$ where G, F are GPAC and FastGPAC segmentation masks respectively. The few differences seen are caused by the effects of regularization on region variability terms with different distance metrics.

Further details on timing and memory usage are presented in Table 3. Due to excessive memory requirements of the original GPAC code, the images shelter-gardens-flowers, Hawaii_Antenna, and people-surfing are reduced from the original size of 1152×864 to the sizes indicated in Table 3, but still exceed the available memory. For larger images, two sets of FastGPAC timings have been presented in Table 3 with and without look-up-table described in Section 5. While not as effective for small images, look-up-table provides further speed-up for larger images, by reducing point to region dissimilarity computation to feature to region dissimilarity computation and look-up-table access. During the performance tests done on the Berkeley Segmentation Dataset [17], FastGPAC results in an average of 226$\times$ speed-up for 100 test images reduced to 241×146 pixels (because of original GPAC's memory requirements). Mean precision and recall of the FastGPAC masks compared to corresponding GPAC masks are 99% and 98.3% respectively.

Figure 2 demonstrates multi-modal capabilities of FastGPAC combined with robust norms. As can be seen in the figure, using FastGPAC any 2 out of 4 regions/distributions can be extracted with appropriate initializations, without blending problems or need for multi-phase level sets which is not possible with two phase Chan&Vese model. Not shown but any single region also can be extracted.

8. Conclusion

We presented a novel efficient graph partitioning active contours algorithm FastGPAC, which without the need of partitioning the input image space into blocks or superpixels, reduces both computational and memory requirements

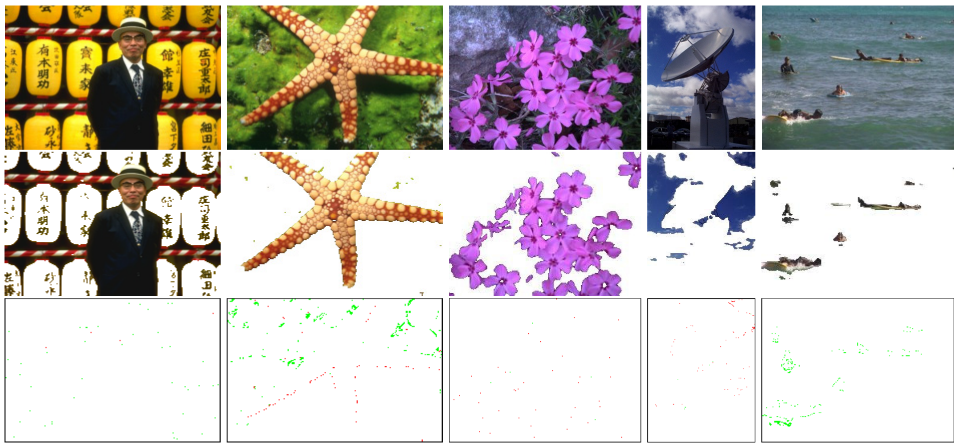


Figure 1: Segmentation result comparison for original GPAC and FastGPAC methods on images (left to right) 65019, 12003 small-Shelter-gardens, small-Hawaii-Antenna, small-people-surfing. First row: original image, second row: FastGPAC segmentation result, third row: difference between GPAC and FastGPAC results, green: foreground in GPAC and background in FastGPAC, red: background in GPAC and foreground in FastGPAC.

of graph partitioning active contours substantially (few orders of magnitude). We demonstrated the conformance of the proposed algorithm to the original graph partitioning active contours and its computational efficiency both theoretically and experimentally. Experiments on the various image types, natural, biomedical etc. show promising segmentation results with substantially reduced computational requirements. FastGPAC algorithm provide flexibility of the GPAC algorithm enabled by possibility of using different image features and different dissimilarity measures while being computationally feasible for large images, high spatial resolutions, and/or long image sequences. We have successfully used FastGPAC on biomedical imagery of different modalities [13]. We are developing an extended FastGPAC and exploring its use on biomedical image sequences.

Acknowledgements Research was partially supported by NIH award R33-EB00573, Air Force Research Laboratory under agreement number FA8750-09-2-0198 and by the Leonard Wood Institute (LWI) in cooperation with the U.S. Army Research Laboratory (ARL) under Cooperative Agreement Number W911NF-07-2-0062. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of LWI, ARL or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

Image	Size	F-measure	Speed-up
65019	144 × 214	0.9983	83×
12003	144 × 214	0.9746	36×
small-shelter-gardens	240 × 180	0.9991	260×
small-Hawaii-Antenna	243 × 182	0.9957	238×
small-people-surfing	242 × 181	0.9409	223×
shelter-gardens	864 × 1152	NA	NA
Hawaii-Antenna	1280 × 960	NA	NA
people-surfing	864 × 1152	NA	NA

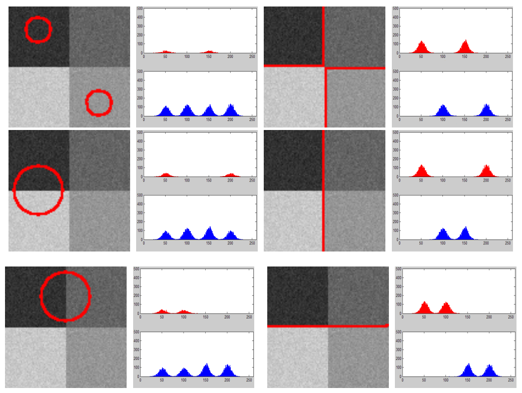
Table 2: Comparison of GPAC and FastGPAC segmentation results for images shown in Figure 1. For the three larger images comparison is not available (NA) since the original GPAC code could not run on the current system because of its the large memory requirements.

References

[1] L. Bertelli. GPAC source code. `http://vision.ece.ucsb.edu/~lbertelli/soft_GPAC.html`.

Image	# iter.	Max Memory	CPU Time
GPAC			
65019 (Man)	114	7477 MB	326 sec
12003 (StarFish)	400	7478 MB	335 sec
small-Shelter-gardens	48	11882 MB	911 sec
small-Hawaii-Antenna	132	11878 MB	1329 sec
small-people-surfing	57	11892 MB	1282 sec
Shelter-gardens	NA	out-of-memory	NA
Hawaii-Antenna	NA	out-of-memory	NA
people-surfing	NA	out-of-memory	NA
FastGPAC			
65019 (Man)	43	2 MB	3.93 sec
12003 (StarFish)	221	2 MB	9.40 sec
small-Shelter-gardens	24	2 MB	3.50 sec
small-Hawaii-Antenna	86	2 MB	5.58 sec
small-people-surfing	84	2 MB	5.74 sec
Shelter-gardens	400	313 MB	502 sec
Hawaii-Antenna	400	364 MB	1164 sec
people-surfing	400	308 MB	920 sec
FastGPAC-LUT			
Shelter-gardens	400	313 MB	401 sec
Hawaii-Antenna	400	364 MB	504 sec
people-surfing	400	314 MB	406 sec

Table 3: Timing and memory usage results for images in Figure 1.



(a) (b)

Figure 2: Contours and R_i,R_o inside, outside intensity distributions on +junction image for three initializations (rows 1-3). (a) Initial state (b) final state. The segmentations shown are obtained using FastGPAC with saturated L_2-square distance and normalization factors $\alpha = \beta = 1$. On each distribution plot red(top) plot:R_i, blue(bottom) plot: R_o. + junction image consists of four regions (R_1, R_2, R_3, R_4) with means: 50/255, 100/255, 150/255, 200/255 respectively and $\mathcal{N}(0, 0.032)$ noise added.

[2] L. Bertelli, B. Sumengen, B. Manjunath, and F. Gibou. A variational framework for multi-region pairwise similarity-based image segmentation. *IEEE Trans. Patt. Anal. Machine Intell.*, 30(8):1400–1414, 2008.

[3] M. Black, G. Sapiro, D. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Trans. Image Proces.*, 7(3):421–432, 1998.

[4] Y. Boykov and G. Funka-Lea. Graph cuts and efficient N-D image segmentation. *Int. J. Comp. Vision*, 70(2):109–131, 2006.

[5] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Proc. IEEE Int. Conf. Computer Vision*, volume 1, pages 26–33, Oct. 2003.

[6] T. Brox and J. Weickert. Level set segmentation with multiple regions. *IEEE Trans. Image Process.*, 15(10):3213–3218, 2006.

[7] F. Bunyak, K. Palaniappan, S. Nath, T. Baskin, and G. Dong. Quantitative cell motility for *in vitro* wound healing using level set-based active contour tracking. In *Proc. IEEE Int. Symp. Biomedical Imaging*, pages 1040–1043, Arlington, VA, April 2006.

[8] F. Bunyak, K. Palaniappan, S. K. Nath, and G. Seetharaman. Flux tensor constrained geodesic active contours with sensor fusion for persistent object tracking. *J. Multimedia*, 2:20–33, August 2007.

[9] T. Chan and L. Vese. Active contours without edges. *IEEE Trans. Image Proc.*, 10(2):266–277, Feb. 2001.

[10] L. Cohen. On active contour models and balloons. *Comput. Vis., Graphics, Image Processing*, 53:211–218, 1991.

[11] I. Cox, S. Rao, and Y. Zhong. Ratio regions: A technique for image segmentation. In *Proc. Intl Conf. Pattern Recognition*, pages 557–564, Aug. 1996.

[12] D. Cremers, M. Rousson, and R. Deriche. A review of statistical approaches to level set segmentation: Integrating color, texture, motion and shape. *Int. J. Comp. Vision*, 72(2):195–215, 2007.

[13] I. Ersoy, F. Bunyak, V. Chagin, M. Cardoso, and K. Palaniappan. Segmentation and classification of cell cycle phases in fluorescence imaging. *Lecture Notes in Computer Science (MICCAI)*, Sep. 2009.

[14] J. Kim, J. Fisher, A. Yezzi, M. etin, and A. Willsky. A non-parametric statistical method for image segmentation using information theory and curve evolution. *IEEE Trans. Image Process*, 14(10), 2005.

[15] V. Kolmogorov and R. Zabin. What energy functions can be minimized via graph cuts? *IEEE Trans. Patt. Anal. Machine Intell.*, 26(2):147–159, 2004.

[16] J. Malcolm, Y. Rathi, and A. Tannenbaum. A graph cut approach to image segmentation in tensor space. In *IEEE Conf. Comp. Vision and Patt. Recog.*, pages 1–8, 2007.

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

[18] N. Paragios and R. Deriche. Geodesic active regions for motion estimation and tracking. In *Proc. Int. Conf. Computer Vision*, pages 688–694, Corfu, Greece, 1999.

[19] N. Paragios and R. Deriche. Geodesic active regions and level set methods for supervised texture segmentation. *Int. J. Com. Vision*, 46(3):223–247, 2002.

[20] R. Ronfard. Region-based strategies for active contour models. *Int.J. Comput. Vision*, 13:229–251, 1994.

[21] M. Rousson and R. Deriche. A variational framework for active and adaptive segmentation of vector-valued images. In *IEEE Workshop Motion Video Comp.*, pages 56–62, 2002.

[22] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Patt. Anal. Machine Intell.*, 22(8):888–905, 2000.

[23] B. Sumengen and B. S. Manjunath. Graph partitioning active contours (gpac) for image segmentation. *IEEE Trans. Patt. Anal. Machine Intell.*, pages 509–521, Apr 2006.

[24] D. Weiler, V. Willert, J. Eggert, and E. Korner. A probabilistic method for motion pattern segmentation. In *Int. Joint Conf. on Neural Networks*, pages 1645–1650, Aug. 2007.

[25] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. Patt. Anal. Machine Intell.*, 15(11):1101–1113, 1993.

[26] N. Xu, N. Ahuja, and R. Bansal. Objcet segmentation using graph cuts based active contours. *Computer Vision and Image Understanding*, 107:210–224, 2007.

[27] T. Zhang and D. Freedman. Improving performance of distribution tracking through background mismatch. *IEEE Trans. Patt. Anal. Machine Intell.*, 27(2):282– 287, 2005.

[28] S. Zhu and A. Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Trans. Patt. Anal. Mach. Intell.*, 18:884–900, 1996.