GPU Enabled Parallel Touching Cell Segmentation Using Mean Shift Based Seed Detection and Repulsive Level Set

Xin Qi(1,2), Fuyong Xing(1,3,4), David J. Foran(1,2,4), Lin Yang(2,4)

1 Dept. of Pathology and Laboratory Medicine, UMDNJ-Robert Wood Johnson Medical School
2 Center for Biomedical Imaging & Informatics, The Cancer Institute of New Jersey, UMDNJ-Robert Wood Johnson Medical School
3 Dept. of Electrical and Computer Engineering, Rutgers University
4 Dept. of Radiology, UMDNJ-Robert Wood Johnson Medical School
Piscataway, NJ, 08854, U. S. A.

Abstract. Automated image analysis of histopathology specimens could potentially provide support for the early detection of breast cancer. Automated segmentation of cells in the digitized tissue microarray (TMA) is a prerequisite for quantitative analysis. However touching cells bring significant challenges for traditional segmentation algorithms. In this paper, we propose a novel algorithm to separate touching cells in hematoxylin stained breast TMA specimens which have been generated using a standard RGB camera. The algorithm starts with an accurate and fast object center localization approach using mean shift based seed detection. The final results are obtained using a multiphase level set with repulsive force. We compared our results with the most current literature. The segmentation results are evaluated by comparing the pixel wise accuracy between human experts’ annotation and the automatic segmentation algorithm. The method is tested using 100 image patches which containing more than 1000 touching cells. As the voting method of the seed detection is the most time consuming procedure, the algorithm is parallelized using graphic processing units (GPU) and 22 times speed up is achieved when compared with the C/C++ implementation.

1 Introduction

Breast cancer is one of most frequently diagnosed cancers in women. It is estimated that approximately 194,280 new cases of invasive breast cancer are expected to occur among women in the US during 2009. There is over a 98% five-year relative survival rate when localized breast cancer is caught before it spreads to other parts of the body [1]. Tissue microarrays (TMA) comprised of small histological sections (histospots) arranged in a matrix configuration on a recipient paraffin block [2]. TMAs provide an efficient method for preserving tissue while facilitating high-throughput of multiple tissue samples in parallel. Digital microscopy has become an extremely valuable tool for visualizing and archiving pathology specimens [3].
Segmenting the cells in the digitized histopathology specimens is usually the first step in automatic image analysis. An unsupervised clustering which utilizes both color and texture features to segment hematoxylin and eosin stained prostate cancer was recently proposed in [4]. A computationally efficient approach that uses color and differential invariants to assign class posterior probabilities to separate epithelial nuclei, stroma and background regions in breast microarray was reported in [5]. Tissue segmentation using color or multispectral images by combining spatial clustering with multiphase vector level set active contours was proposed for segmenting histological prostate cancer tissues in [6].

These segmentation methods produce good results on regions exhibiting little or no cell overlap. However, they usually fail to accurately segment touching cells. The watershed is the most commonly used segmentation method to handle touching objects. However, it often suffers from over segmentation. Some algorithms, such as a rule-based approach [7], were proposed to address this problem, by merging over-segmented regions. Voronoi diagram has been applied to handle the overlapping regions in [8]. However it is difficult to derive a generalized rule to merge the over-segmented regions across different image sets. Wen et. al. [9] reported their study on decomposing clumps of nuclei using high-level geometric constraints derived from maximum curvatures. This approach proved to be very effective in separating touching nuclei, however for some touching objects, the common connecting region may not exhibit local maximal curvatures. Kothari et al. [10] proposed a semi-automatic method for touching cell segmentation, in which they applied concavity detection at the edge of a cluster to find the points of overlap between two nuclei. An ellipse-fitting technique was then applied to segment the concavities between two nuclei of overlap region. However, the ellipse can not accommodate the shape of some cells, especially for cancerous cells. Yang et al. [11] proposed an approach to address touching cell segmentation using concave vertex graph. Another graph based method was proposed in [12] to segment touching stem cells within fluorescence microscopy images. These graph-based methods usually require that the images exhibit a high contrast at the edge, which is often not the case especially in the cancer regions.

2 Touching Cell Segmentation

In this paper, we propose a novel algorithm to separate touching cells within breast TMA using standard RGB imaging. Hematoxylin stained breast TMA specimens were acquired at a high magnification objective (40×), at which the images show the cellular structure within the breast tissue. The touching cell segmentation algorithm for microscopic breast TMA image is composed of two main modules. The first module is to locate the center of each cell using a novel mean shift and voting based seed detection method, which is used as an initial position for the second module. The second module is the touching cell segmentation, which extracts each touching cell’s contour using a level set function with a repulsion force to penalize object overlap. In order to speed up
the whole segmentation procedure, the most computational expensive step has
been parallelized using a graphic processing unit (GPU).

2.1 Seed Detection

Since the number and location of cells is not known apriori, it is difficult to
directly segment cell from microscopic images especially when the cells are touch-
ing one another. The center of cells is considered as a basic perceptual cue that
supports the accurate separation of touching cells. Parvin et. al. [13] proposed
an iterative voting method which we have implemented to detect the center of
objects. This algorithm provides excellent results in detecting the centers in the
touching cells.

In our algorithm, we define $I(x, y)$ as the original image, the
image gradient $\nabla I(x, y)$ and its
magnitude $\|\nabla I(x, y)\|$ are sub-
sequently calculated. For each
pixel $(x, y)$, the voting direction
$\alpha(x, y)$ is defined as its negative
gradient direction $-\frac{\nabla I(x, y)}{\|\nabla I(x, y)\|} =
-(\cos(\theta(x, y)), \sin(\theta(x, y))$ where
$\theta$ is the angel of the gradient di-
rection with respect to $x$ axis. The
voting area $A(x, y; r_{min}, r_{max}, \Delta)$
of each pixel is defined by a cone-
shape with its vertex at $(x, y)$. We
used a cone-shape voting area
for two reasons. First the center
of the cell is far away from the
boundary of the cell, thus more voting points locate within the region close
to the center than within the region close to the edge of the cell. Second, a cone-
shape voting area will greatly reduce the time complexity of the algorithm for
calculating fewer voting points in total. The $r_{min}, r_{max}, \Delta$ and voting area of
pixel $(x, y)$ are illustrated in Figure 1. We define a 2D Gaussian kernel $g(x, y, \sigma)$
with its mean $\mu$ located at the center of the voting area $A$ and oriented in the
voting direction $\alpha(x, y)$. The center of the Gaussian kernel

$$g(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{(x - \mu_x)^2 + (y - \mu_y)^2}{2\sigma^2} \right)$$

(1)

where $\mu_x = x + \frac{(r_{max} - r_{min})\cos(\theta)}{2}$ and $\mu_y = y - \frac{(r_{max} - r_{min})\sin(\theta)}{2}$. We design the kernel
in this way so that the voting will be reinforced at the center of the object. We
define $V(x, y; r_{min}, r_{max}, \Delta)$ as the voting image, which has the same dimensions
as the original image $I(x, y)$. For each pixel $(x, y)$, we update the voting image
Algorithm 1: Mean shift based seed voting

1. Initialize the parameters: \( d = 40, r_{\text{min}} = 0.5d, r_{\text{max}} = 1.5d, \Delta = 30 \). The bandwidth of the mean shift is set to be 6.
2. Calculate the Gaussian blurred gradient image and the orientation of the gradient at each pixel \((x, y)\). Record the set of \((x, y)\) with large gradient magnitude as \(S\).
3. For each point \((x, y) \in S\), calculate the voting image \(V(x, y)\).
4. for \(R = 0.3, 0.4, ..., 0.9\) do
5. Record all the points \((x, y)\) in the voting image with voting number larger than \(\max(V(x, y)) \times R\).
6. end for
7. Sum all the voting image and run mean shift to generate the final list of the seeds.

as

\[
V = V(x, y; r_{\text{min}}, r_{\text{max}}, \Delta) + \sum_{(u, v) \in A} \|\nabla I(x + u, y + v)\| g(u, v, \sigma). 
\]

(2)

The centers of object are determined by running mean shift on the sum of the voting image after thresholding. The detail algorithm is listed in Algorithm 1.

Although our work was initially motivated by [13], the method we have developed is significantly different: For each point \((x, y)\) with high gradient, we define a convolution Gaussian kernel centered at the center of the voting area instead of \((x, y)\). This is a critical step which explains why our algorithm performs better. As the center of the object is usually far away from the boundary, the Gaussian kernel we designed can reinforce the voting to the center of the object. (We will revisit this issue in the experimental section). Meanwhile, instead of using iterative voting, we proposed to calculate the center of touching objects by running mean shift on the final set of voting images. This step dramatically reduces the computational time as we avoid many iteration steps.

2.2 Parallelization of the Seed Detection on the Graphic Processing Unit (GPU)

In our seed detection module, the voting algorithm is based on the voting pixels. Each voting pixel \((x, y) \in S\) has a cone-shape voting area \(A(x, y; r_{\text{min}}, r_{\text{max}}, \Delta)\). Each pixel in the final voting image \(V\) is updated by equation 2. Within the seed detection procedure, the most computational expensive part is the calculation of the voting image, therefore voting algorithm was accelerated by introducing parallelization on a graphics processing unit (GPU) to exploit its high performance computing power.

In Figure 2, the execution time profile is shown for each step of our proposed seed detection algorithm when analyzing a typical 2D pathology image \((1392 \times 1040)\). It is quite obvious from the figure that the bottleneck is the voting step. Because the voting algorithm is pixel based method which has an advantage of easy parallelization as a result of data independence. By fully utilizing this property, we utilized eight blocks for a GPU. Within each block, we create 128 threads. Each voting pixel was assigned to one thread to calculate its
corresponding voting image. In total 1024 threads were created simultaneously. In this way the GPU accelerated the voting image calculation dramatically and therefore increase the whole seed detection procedure.

2.3 Cell Segmentation

Given the accurately detected seeds, the touching cell segmentation was performed using level set based on an interaction model. The interaction model includes two types of mechanisms: a repulsion term prevents the contours of adjacent cells from overlapping and separates the touching cell boundaries; the competition term determines the membership of each pixel which is assigned to the cell producing the smallest difference. Considering an image $I$ that has $N$ cells, let $C_i (i = 1, \ldots, N)$ denote the contours that evolve towards the boundaries. Notice that each cell is represented by its own level set energy function. Instead of examining each contour independently, the interaction between neighboring contours was integrated into the level set energy function. The energy function $E$ for cell segmentation combines the repulsion and competition terms and can be expressed as:

$$E = \lambda_0 \sum_{i=1}^{N} \int_{\text{in}(C_i)} |I - c_i|^2 \, dxdy + \lambda_b \sum_{i=1}^{N} \int_{\Omega_b} |I - c_b|^2 \, dxdy + u \sum_{i=1}^{N} \int_{0}^{1} g\left(|\nabla I (C_i (q))|\right) \left|C'_i (q)\right| \, dq + \omega \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} A_i \cap A_j$$  \hspace{1cm} (3)

where $A_i$ denotes region of cell $\{A_i | i = 1, 2, \cdots, N \}$ and $\Omega_b$ is the background which represents the region outside all the cells $\text{out}(C_1) \cap \text{out}(C_2) \cap \cdots \cap \text{out}(C_N)$. The operator $\text{in}()$ and $\text{out}()$ represent the regions inside and outside of the cell, respectively. The $c_i$ and $c_b$ are the intensity means of the cell region and background region, respectively. The $\lambda_0$, $\lambda_b$, and $u$ are the fixed weighting parameters. Function $g$ is chosen to be a sigmoid function

$$g(x) = \left(1 + e^{\alpha x} \right)^{-1}$$  \hspace{1cm} (4)
where $\alpha$ is used to control the slope the output curve and $\beta$ controls the window size. By penalizing the union of the overlapped region $\{A_i | i = 1, \cdots, N\}$ enclosed by contours $C_i (i = 1, \cdots, N)$, the last item in Energy function $E$ is the repulsion term which is used to represent the repulsion force between each adjacent touching object and the $\omega$ is the regulation parameter.

The segmentation is achieved by minimizing the energy function $E$ using evolution of level set. In order to express the energy function using level set, we bring in the regularized Heaviside function $H$ [14]

$$H_\varepsilon (z) = \frac{1}{2} \left( 1 + \frac{2}{\pi} \arctan \left( \frac{z}{\varepsilon} \right) \right)$$

where $\varepsilon$ is the regulation parameter of the Heaviside function and Delta function is defined as

$$\delta_\varepsilon (z) = \frac{d}{dz} H_\varepsilon (z).$$

The energy function can be minimized iteratively employing the gradient descent method. The evolution equation for each energy function $\Psi_i (t, x, y)$ is then obtained by deducing the associated Euler-Lagrange equation as

$$\frac{\partial \Psi_i}{\partial t} = \delta (\Psi_i) \left\{ \lambda_0 |I - c_i|^2 - \lambda_b |I - c_b|^2 \prod_{j=1, j \neq i}^N H (\Psi_j) ight. \\
+ \mu \nabla g \cdot \frac{\nabla \Psi_i}{|\nabla \Psi_i|} + \gamma \text{div} \left( \frac{\nabla \Psi_i}{|\nabla \Psi_i|} \right) + \omega \sum_{j=1, j \neq i}^M (1 - H (\Psi_j)) \right\}.$$  

After evolving the level sets, the means $c_i$ and $c_b$ of the cell and background regions are iteratively updated. This method has proven to be quite effective and accurate for RNAi fluorescent cellular image segmentation in [15]. Throughout the experiments, the parameters that we selected were: $\lambda_0 = 1, \lambda_b = 0.3, \mu = 0.5, \gamma = 0.2, \omega = 0.6, \epsilon = 1, \alpha = 1, \beta = 7.$
Fig. 4. Seed detection results using real dataset where the errors on the overlapping part are marked with yellow rectangles. (a) The original Hematoxylin stained TMA image. (b) The ground truth annotation by human experts. (c) The seed detection results using the iterative voting method in [13] (d) The seed detection results using our mean-shift based seed voting method.

3 Experimental Results

Hematoxylin stained breast TMA specimens were used throughout these experiments. The images were captured at a high magnification objective (40×). In total we have 100 image patches containing more than 1000 imaged cells.

Figure 3 shows the seed detection results on a synthetic image with five touching cells. From this experiment, it can be seen that the iterative voting method tends to put the seed at the overlaid region when several cells are touching together. This is because the voting number calculated in [13] is the convolution result of a Gaussian centered at the boundary point \((x, y)\), which is the location with the higher gradient. However, the center of the object should be far away from the boundary. In Figure 4 we show the results using real data. The seed detection errors using the method in [13] are marked with yellow rectangles. In our algorithm, we utilize a Gaussian kernel defined at the center of the voting area for each voting point \((x, y)\) calculated by thresholding the gradient image. In this way we penalize the voting towards the boundary of the object and enforce the voting towards the center of the object. This is the major reason that our method can provide better results for cells with overlapping regions.

In Figure 5, the performance of our proposed touching cell segmentation method (Figure 5d) is compared with the marker based watershed segmentation algorithm [16] (Figure 5c). Each row represents one testing sample. The contours are presented using red lines. Figure 5a shows the original RGB images. Figure 5b represents the ground truth annotations. Figure 5c represents the segmentation results using marker based watershed, where the markers are provided by our
Seed detection method. Figure 5 represents the original image overlaid with the segmentation result using mean shift based seed detection and repulsive level set.

In order to quantitatively measure the accuracy, precision ($P$) and recall ($R$) are calculated for our touching cell segmentation algorithm and compared with the ground truth annotations. The $P$ is defined as the union between the segmentation results and the manually annotation results divided by the segmentation results. The $R$ is defined as the union between the segmentation results and manually annotation results divided by the manually annotation results. The mean and standard deviation of $P$ and $R$ for our touching cell segmentation algorithm is $0.90 \pm 0.02$ and $0.78 \pm 0.03$, respectively, which indicates a good agreement between the manual and ground truth annotations. In Table 1, we compared the quantitative segmentation results of our algorithm compared with the ground truth annotation. The 80% column in Table 1 represents the sorted 80% accuracy among all the segmentation results.

The parallelization code was developed using Graphic Processing Unit (GPU). GPU is a massively parallel multiple cores chip that can execute thousands of concurrent threads. The GPU cores (also called stream processors) are grouped...
Table 1. The segmentation accuracy compared with ground truth annotation

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>Variance</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
<th>80%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segmentation accuracy (Precision $P$)</td>
<td>0.90</td>
<td>0.02</td>
<td>0.96</td>
<td>0.15</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Segmentation accuracy (Recall $R$)</td>
<td>0.78</td>
<td>0.03</td>
<td>0.70</td>
<td>0.14</td>
<td>0.97</td>
<td>0.81</td>
</tr>
</tbody>
</table>

into several streaming multiprocessors. They are managed by the thread manager. Our GPU is the NVIDIA Quadro FX5800 which has 240 cores and 30 streaming multiprocessors, each of which contains 8 GPU cores. It supports both single and double float point precision, offers 933 GFlops single precision, and has memory bandwidth with 102 GB/sec. The streaming multiprocessor is a multithread instruction unit which allows 32 Single Instruction, multiple Data (SIMD) warps of 32 threads and provides independent Multiple Instruction, Multiple Data (MIMD) thread execution and hardware thread scheduling. Unlike CPU threads that are heavy weight, GPU threads are light weight with little creation overhead, instant switching, and instruction and memory latency hiding. The Compute Unified Device Architecture (CUDA) environment from NVIDIA Corporate is utilized in our parallel implementations.

The experimental results have shown significant speed up (22 times faster compared with the sequential implementation on CPU and thousands of times faster than the original Matlab implementation). In the parallel version of the algorithm, we can complete the seed detection procedure for an image with dimensionality $1392 \times 1040$ in 197 millisecond.

4 Conclusion

In this paper we presented a touching cell segmentation algorithm on a large clinical dataset. We demonstrated that using mean shift based voting algorithm we can accurately detect the center of each touching cell. Given these estimated seeds, a level set active contour based on the interaction model can effectively separate each of the touching cell. The improved segmentation results are achieved by accurately estimating the seed (number of cells) and utilizing a repulsion term in the level set energy function to separate the touching boundaries. The method is automatic and not limited to domain specific prior knowledge. Therefore it can be extended to other multichannel touching object segmentation applications. The GPU implementation of our algorithm can handle one 2D image in less than 0.2 second. The experimental results show that GPU is an efficient parallel platform for touching cell segmentation.

5 Acknowledgement

This research was funded, in part, by grants from the NIH through contract 5R01EB003587-04 from the National Institute of Biomedical Imaging and Bio-engineering and contracts 5R01LM009239-03 and 3R01LM009239-03S2 from the National Library of Medicine. Additional funds were provided by IBM through a Shared University Research Award. This research is also partially supported by UMDNJ Foundation grant #66-09.
References