Project Title: MIRACLE - Microscopic Image Processing Analysis Coding and Modelling Environment

Contract No : PIRSES-GA-2009-247091

Instrument : FP7 PEOPLE Marie Curie Actions IRSES

Thematic Priority : Medical image processing

Start of project : 1 May 2010

Duration : 36 months

Deliverable No : D7

Segmentation of follicular lymphoma images based on Markov Random Fields (MRF)

Due date of deliverable : May 2012

Actual Submission date : November 2012

Version : 2

1. Introduction

Segmentation of an image is an important task in many applications of computer vision. Image segmentation is a visual information processing and its main goal is the unsupervised partitioning of the visual entities into "coherent" parts, the so-called classes, by simultaneously localizing the boundaries of these entities in the image. "Good" data partitioning or clustering is a vague concept since there is no universal measure for the coherence of a cluster. Intuitively, coherence should reflect intra-cluster homogeneity and inter-cluster separability.

Over the last 20 years the computer vision community has produced a number of useful algorithms for image segmentation. There are gray-level thresholding [1], snakes [2][3], active contours [4], geodesic active contours [5][6], "shortest path" techniques [7] and many other examples of methods for partitioning an image into two segments: "object" and "background". Each method comes with its own set of features.

One emerging approach that has over the recent years gained more and more importance is the s-t graph cuts technique [8][9]. Greig et al. [10] were the first that applied graph cut algorithms to computer vision problems. Graph cut method combines both regional and boundary information to the segmentation of an image. It involves assigning a label (such as intensity or the result from a classification algorithm) to every pixel and is based on energy minimization. A common constraint is that labels should vary smoothly almost everywhere while preserving sharp discontinuities that may exist, e.g. at object boundaries. Global minimization is NP-hard even in the simplest discontinuity-preserving case. Due to the inefficiency of computing the global minimum, many authors have opted for a local minimum. However, in general, a local minimum can be arbitrarily far from the optimum [11]. A wide class of energies with various smoothness constraints has been proposed [12][13][14]. All these energy functions' objective is to find the optimal moves, meaning the optimal combinations of labeling of the pixels in the image. Assuming a labeling f, the labelings near to f are those that lie within a single move of f. However, many local optimization techniques use what is called standard moves, where only one pixel can change its label at a time. But this is indeed a very weak condition and therefore, low quality solutions are generated.

On the other hand, Boykov et al. [11] reported an efficient approximation algorithm of global minimization of these energy functions. They developed sampling algorithms for a well-known discontinuity preserving function, the so-called Potts model, which can make larger moves using MRF approach to assign the optimal labeling to the pixels. These moves are referred to as α-β-swaps and they were able to compute the best move among all possible ones while in [11][15], they randomly select connected subsets of pixels that change their label from α to β. Therefore, α-β-swaps relax the afore-mentioned constraint of standard moves.

The approach of Boykov et al. [11] is examined in this study in order to segment follicular lymphoma images. The goal is to find the five different regions that comprise any follicular lymphoma image. These regions can be considered as cytological components. There are five major cytological components in the FL tissue: nuclei, cytoplasm, extra-cellular material (ECM), red blood cells (RBC) and background regions. As the images have been dyed, nuclei and cytoplasm regions have a hue of blue and purple respectively, whereas extra-cellular materials are identified based on a hue of pink and RBCs based on a hue of red. The background, which does not correspond to any tissue component, is always white.

2. Data

The data used in the current study was acquired as following: Tissue biopsies of FL stained with H&E, from 17 different patients were scanned using high-resolution whole slide scanner (Aperio - Image Scope). Three board-certified hematopathologists selected 500 HPF images of follicular lymphoma out of the scanned tissue biopsies. Each image was of size 401-by-401 pixels.

Some variation in the texture of the cells exists, among images. Therefore, a smoothing operation on the grayscale version of every RGB image is applied, in order to reduce texture variations inside the cells. Two different filters were examined, Gaussian filter, as well as median filter. They both have texture preserving properties. Furthermore, each image was standardized by removing its mean value and dividing it then by its standard deviation.

3. Method

Graph-based image segmentation techniques generally represent the problem in terms of an undirected graph $G =< V, E >$ where each node $v_i \in V$ corresponds to a pixel in the image, and the edges in E connect certain pairs of neighboring pixels. A weight is associated with each edge based on some property of the pixels that it connects, such as their image intensities. Depending on the method, there may or may not be an edge connecting each pair of vertices.

In the current study, the segmentation examined is based on s-t graph cut algorithm [8] [11] . In this approach, there are two kinds of nodes; terminal nodes including s (source) and t (sink) and neighborhood nodes which correspond to the pixel in the image. The edge between two neighboring nodes is named as n-link, while the edge connecting terminal nodes with neighboring ones is described as t-link. In s-t graph, every edge is given a non-negative weight w. A cut C is a subset of edges E and the cost of a cut is the sum of the weights on edges in C, which is expressed as:

$$C_{\cos t} = \sum_{e \in C} w_e$$

A minimum cut is a cut that has the minimum cost among all of the cuts. For image segmentation, minimum cut should occur at a position that can efficiently separate the object from background. Min-cut is the method to obtain the minimum cut in a graph and it is implemented by finding the graph's maximum flow. Min-cut/max-flow algorithms [16][17][18] group the graph nodes into the two clusters, s (source) and t (sink). Conventionally, the cluster with s node represents the foreground while the other cluster denotes the background.

Regarding graph cut segmentation, the key step is to establish an energy function and assign weight to the two types of edges (t-links and n-links). The weight w is, in terms of an energy criterion, the following:

$$E(f) = E_{smooth}(f) + E_{data}(f) \qquad (1),$$

where E_{smooth} measures the extent to which f is not piecewise smooth, while E_{data} measures the disagreement between f and the observed data. The two energy terms are in the context of Bayesian labeling of first-order Markov Random Fields (MRF) [19]. MRFs are a kind of statistical models to model spatial constraints such as smoothness of image regions or spatial regularity of textures in a small region.

In this study, the Ising model [11] was used as the energy function E_{smooth}, whereas a data cost function which finds the likelihood of the pixel to belong to each cluster according to its RGB value was used for the E_{data} function. What one is looking for is the optimal labeling to each pixel that would result to a minimization of the Equation (1). This can be interpreted as a maximization of the Posterior estimation of Markov Random Field (MAP-MRF). In [20] they prove that the maximum a posteriori estimate of such an MRF can be obtained by solving a multiway minimum cut problem on a graph. To this end, this kind of graph cut segmentation algorithm is a probabilistic segmentation approach which model image properties as a random field whose probability density function is assumed to be Gibbsian.

4. Goal

Our goal was to efficiently segment an FL image in order to identify the cells in it. Specifically, we are interested in the regions of nuclei in the image, since the nucleus is the most suitable cell component to discriminate between centroblasts and centrocytes in FL images. By knowing the regions of nuclei, we can later apply cell splitting algorithm to find all the individual cells in the image. A classification method would then be applied in order to categorize each cell in a known class and grade the FL images based on the number of cells in a specific class of malignant cells.

5. Workflow and results using Graph Cuts

The RGB FL image used in this study and its grayscale version are shown in Figure 1. Besides the red–green–blue (RGB) color space, we also converted the image to the L*a*b color space, a perceptually uniform color space developed by Commission Internationale d'Eclairage (CIE). Perceptually uniform means that the same amount of change in color values produces the same amount of perceptual difference of visual importance. This property of the L*a*b color space allows us to use the Euclidean distance in comparing the colors [21]. Lab space is suitable when applying k-means, as we did in this work. Moreover, the L*a*b color space separates the luminance and the chrominance information such that L channels corresponds to illumination and a* and b* channels correspond to color opponent dimensions. Hence, the feature vector for each pixel contains intensity and color information separately. In Figures 2 and 3, the three channels in RGB and Lab space are presented.

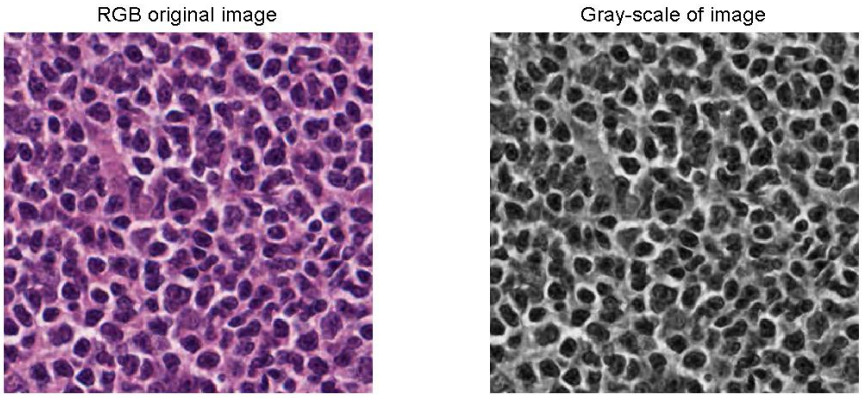

Figure 1.
RGB image (left) used in this study and its grayscale version(right).

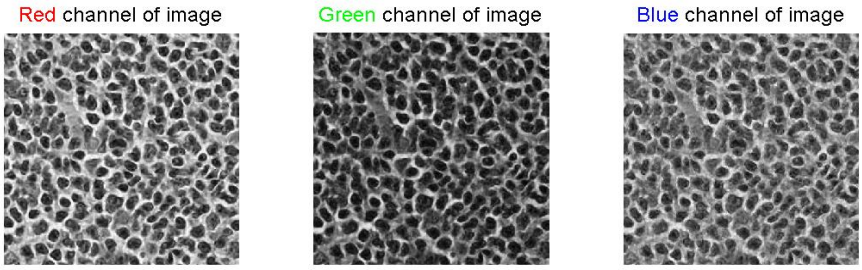

Figure 2.
The red channel (left), the green channel (middle) and the blue channel (right) of the original image.

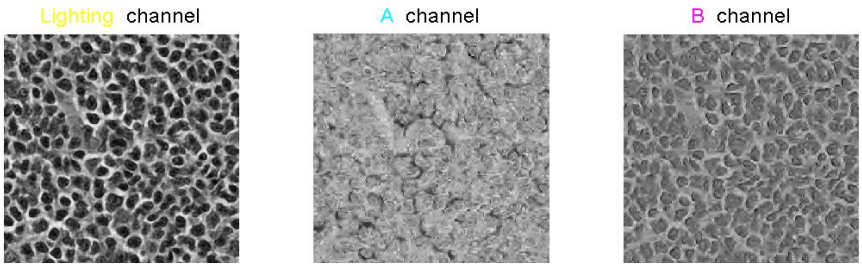

Figure 3.
The light channel (left), the A channel (middle) and the B channel (right) of the original image.

As mentioned in the Data section, it is necessary to pre-process the FL images in order to remove noise during acquisition process. We first applied histogram equalization and then filtered the images with two different filters in order to compare them and choose the optimal one. One tested filter was the median filter with a kernel window of 5-by-5 and the other one was the Gaussian filter with a corresponding window of 14-by-14. The sizes of both windows were chosen empirically. In Figures 4 and 5 we show the results in filtering and in applying histogram equalization. From these figures we concluded that the optimal filter is the median filter due to the good contrast of the resulted filtered image in this case.

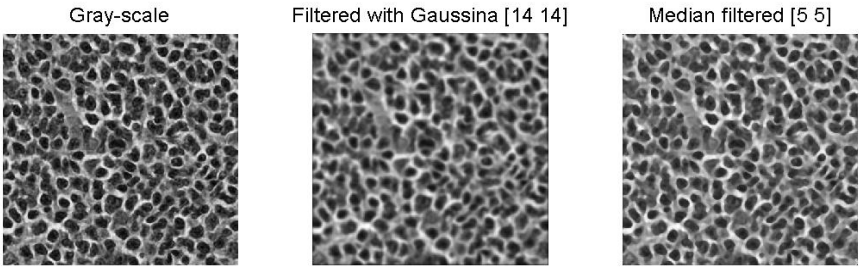

Figure 4.
Grayscale version of the original image (left), the result after applying Gaussian filter on the grayscale image with window [14 14] (middle) and the result after applying median filter on the grayscale image with window [5 5] (right).

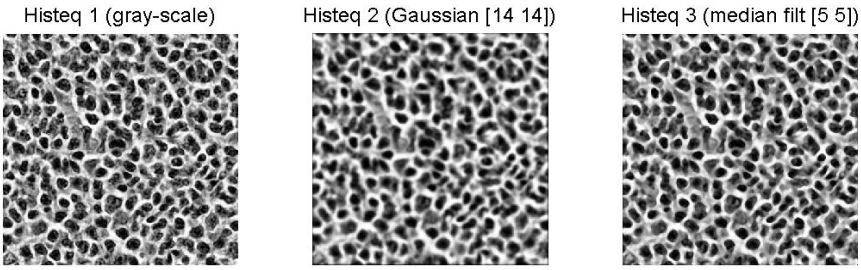

Figure 5.
Grayscale version of the original image after histogram equalization (left), the result after applying Gaussian filter and histogram equalization on the grayscale image with window [14 14] (middle) and the result after applying median filter and histogram equalization on the grayscale image with window [5 5] (right).

We therefore chose an image that is filtered with median filter, histogram equalized and transformed in the Lab space. As a first step in segmentation, regions of white background and red blood cells need to be identified since they do not correspond to any tissue component. RBCs and background regions show relatively uniform patterns; thus, they are segmented by thresholding the intensity values in the RGB color space. The segmentation of RBCs is relatively straightforward by using a simple threshold operation in the RGB color space. A pixel-wise thresholding is used to determine the RBC regions as follows:

$$RBC(i,j) = \{^{1if\ \frac{r(i,j)}{r(i,j)+b(i,j)+c(o,j)} \geq \tau}_{0,otherwise} \qquad (2)$$

where *i, j* indicate the pixel coordinates; and r, g and b are the corresponding red, green and blue color channels of the image in the RGB color space. The threshold τ is experimentally chosen as 0.45 after examining the histogram values of several representative regions. Similarly, the background pixels were identified using a threshold as follows:

$$bckgrnd(i,j)\{^{1,if\ (r(i,j)\geq\xi)\&(g(i,j)\geq\xi)\&(b(i,j)\geq\xi)}_{0,otherwise} \qquad (3)$$

where ξ was determined to be 200. In Figure 5 the original RGB image is shown, with the identified background regions in black (regions of white background and red blood cells). It

can be observed that by using the afore-mentioned threshold, the background regions are accurately identified.

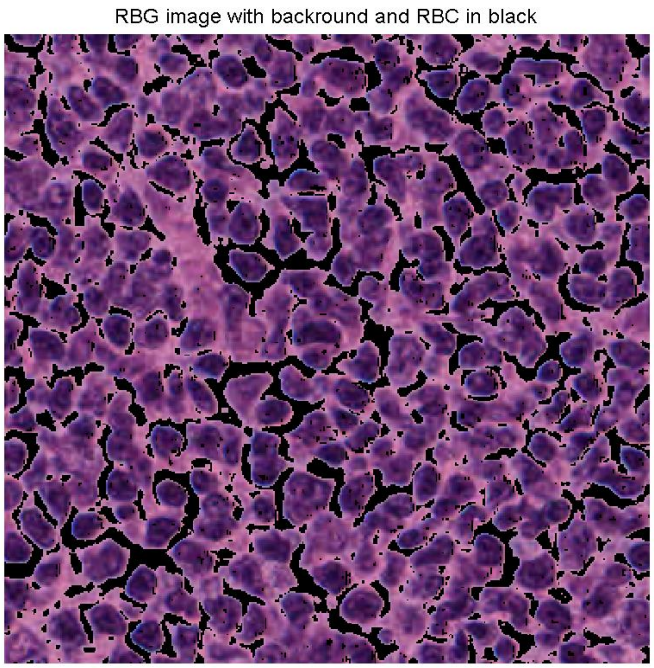

Figure 5.
The original RGB image with its background shown in black.

Graph Cuts (GC) are then applied for the first time to the filtered image in order to segment the foreground (which now is consisted of the ECM, the cytoplasm and the nuclei) from the identified background (consisting of white background regions and red blood cells). The reason we use GC even if we have identified the background regions is because GC will refine the result from thresholding. The result from the first application of GC in the image is shown in Figure 6.

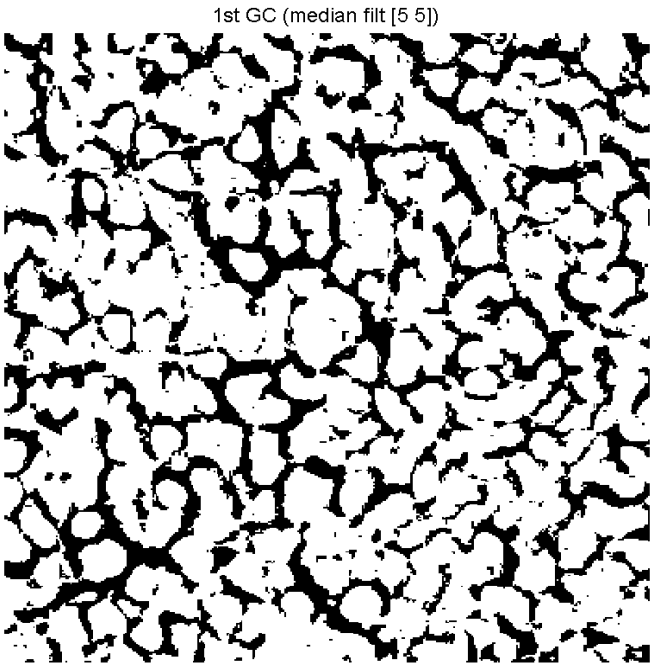

Figure 6.
Binary image in which foreground (ECM, cytoplasm and nuclei) is shown in white and background (RBC and white regions) in black. The binary image is the result of applying graph cuts in the grayscale version of the median filtered image.

Opening, a morphological operation is applied on the image in Figure 6 in order to remove small regions in both the foreground and background and the result from filling the holes is shown in Figure 7.

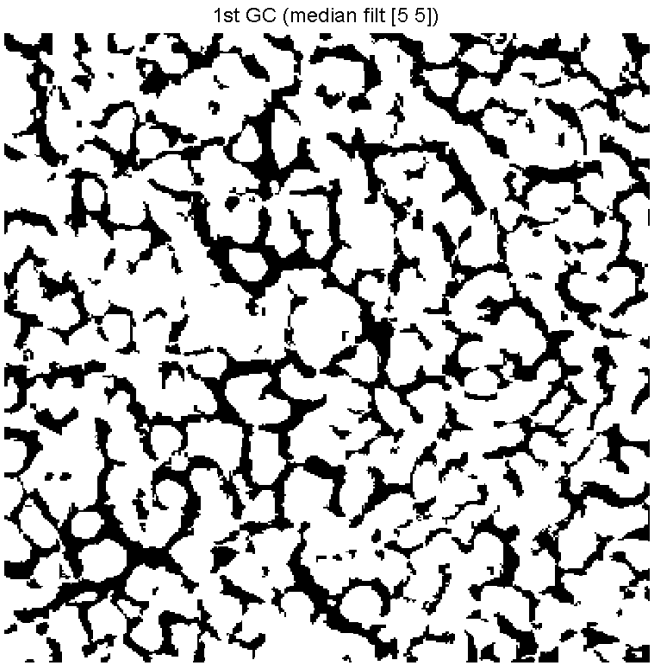

Figure 7.
Image of figure 6 after filling the small holes in both foreground and background.

K-means algorithm is then used in the resulting foreground regions of the image in the Lab space, so as to find the pixels belonging to ECM. The pixels belonging to the ECM are assigned the same label with the white regions and red blood cells, making a new background. Graph Cuts are then applied for the second time to segment the inner cell components (cytoplasm and nuclei) from the rest. The result from this second application of GC (after filling the holes again with the same technique as before) is shown in Figure 8.

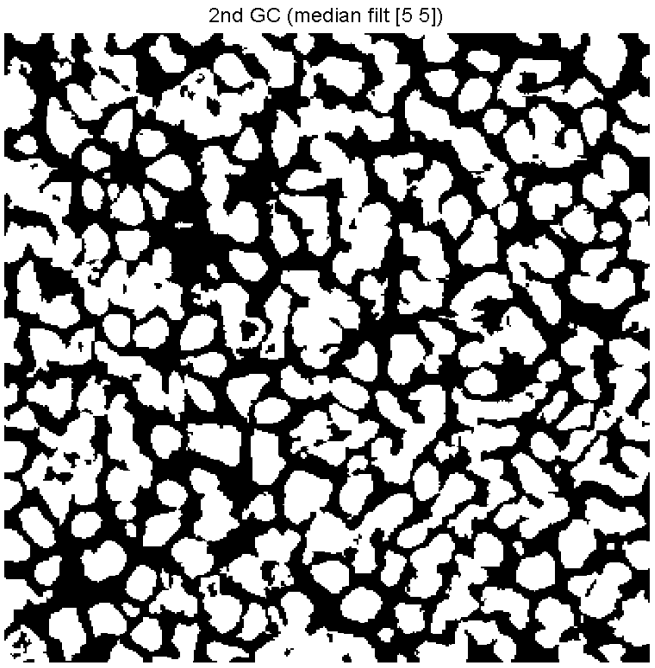

Figure 8.
Binary image in which foreground (cytoplasm and nuclei) is shown in white and background (RBC, white regions and ECM) in black. The binary image is the result of applying graph cuts for the second time in the grayscale version of the median filtered image. Holes have been filled.

.

K-means is then applied for the last time on the resulting foreground regions so as to find the cytoplasm in the pixels of the cells and then label them with the same value as the ECM, the white background and the red blood cells, resulting in a new background. Graph Cuts are applied for the last time to segment the nuclei from the rest. The result of this latter operation is presented in Figure 9.

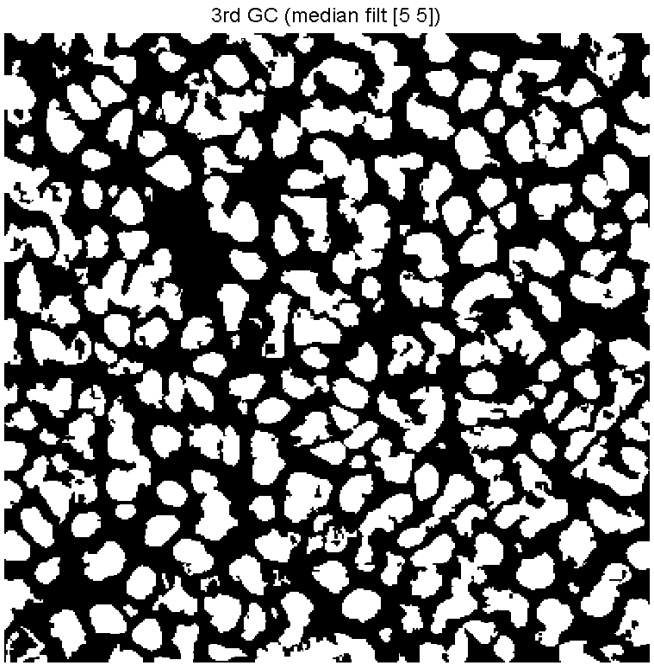

Figure 9.
Binary image in which foreground (nuclei) is shown in white and background in black. The binary image is the result of applying graph cuts for the third time in the grayscale version of the median filtered image. Holes have been filled.

Moreover, we can easily notice that some regions are considered as one single cell, whereas in reality there are more than one "touching" cells. These cells, due to the fact that they appear as one single big cell, are frequently erroneously considered as centroblasts (CB), a type of malignant cells. Therefore, the grading of the image is then biased. We need therefore to split this kind of cells. We applied a Watershed-based splitting algorithm that takes into account the average area of the cells. The result is presented in Figure 10.

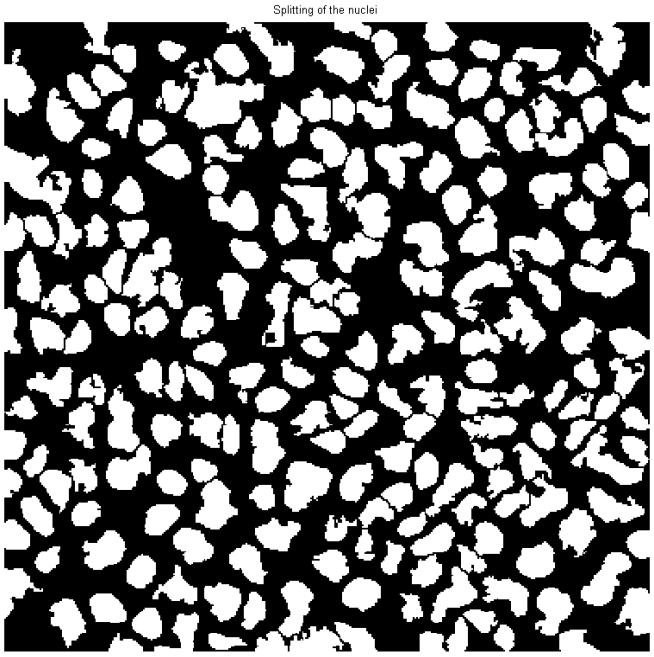

Figure 11.
Binary image in which the borders of the touching cells are easily observed.

6. Results using Otsu Thresholding

In order to assess the accuracy of segmentation using GC, we examined another, simple and commonly used segmentation algorithm, the Otsu thresholding [1]. We followed the same procedure in identifying the nuclei regions, as the one followed when using GC. A first step in segmentation, was the identification of the white background and the regions of the red blood cells, since they do not correspond to any tissue component. RBCs and background regions show relatively uniform patterns; thus they were segmented by thresholding the intensity values in the RGB color space (equations 2 and 3). Then we applied Otsu thresholding only on the regions that were considered as foreground in the first phase, meaning the regions that include the ECM, the cytoplasm and the nuclei. In the resulting binary image, the foreground is assumed to be the regions of the cytoplasm and the nuclei. The resulting background (assuming to be the ECM regions) is then mixed with the regions of RBC and white background, resulting in an expanded background that includes now three of the components, RBC, white background and ECM (shown in Figure 12 after filling the small holes).

We applied Otsu thresholding for a second time, in the region that was denoted as foreground in the previous step (the regions of cytoplasm and nuclei). The resulting background from this process was mixed with the pixels that were considered as background in the previous step. In other words, we identified the regions of cytoplasm by applying for the second time Otsu thresholding and then we mixed the pixels belonging in cytoplasm with those belonging to the RBC, the white background and the ECM. The resulting binary

image (shown in Figure 13 after filling the small holes) denotes the regions of nuclei (as foreground shown in white) and the rest of the components as background shown in black. Finally, Figure 14 depicts the result from splitting the “touching” nuclei after applying the same procedure used in the GC segmentation.

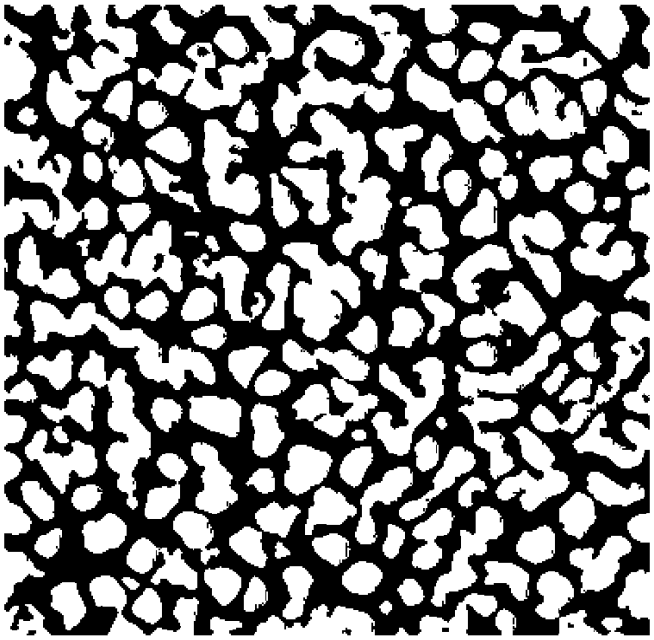

Figure 12.
Binary image after the first segmentation with Otsu. Foreground is shown in white (includes cytoplasm and nuclei) and background in black (includes RBC, white regions, ECM)

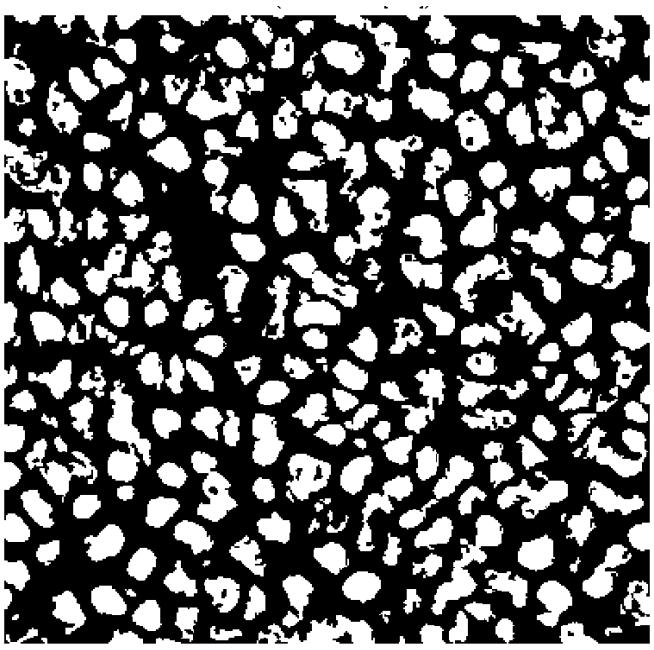

Figure 13.
Binary image after segmentation with Otsu for the second time.Foreground is shown in white (includes only nuclei) and background in black (includes RBC, white regions, ECM and cytoplasm)

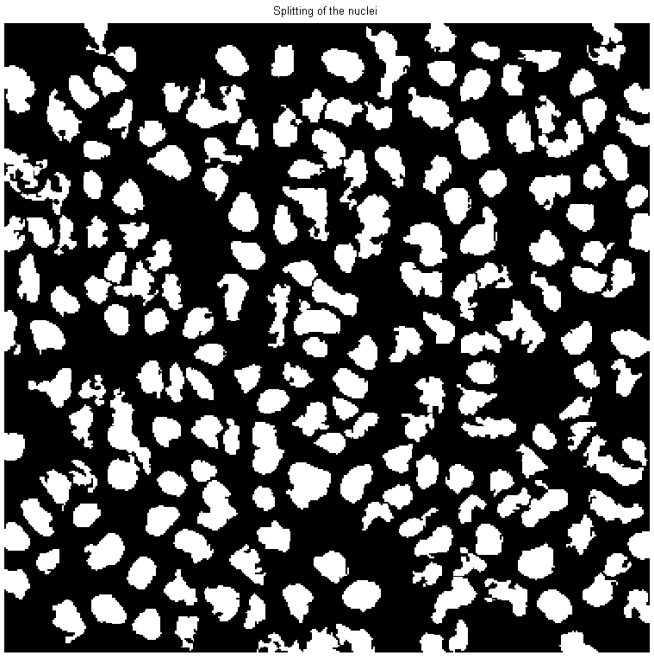

Figure 14.
Binary image where the splitted nuclei are shown as foreground in white. The borders of each nucleus are clearly seen.

7. Comparison of the two methods

In this section, the results from segmentation using the two pre-described methods are given. In order to evaluate the performance of the two methods, we used the following metrics: precision, recall and harmonic average.

Precision is defined as: the ratio of the number of correctly classified as foreground pixels to the total number of cells classified as foreground:

$$precision = tp/(tp+fp) \quad (16)$$

where *tp* corresponds to the number of the pixels that were correctly classified as foreground and *fp* corresponds to the number of the pixels that were erroneously classified as foreground.

Recall is defined as: the ratio of the number of pixels correctly classified as foreground to the number of pixels expected to be classified as foreground:

$$recall = tp/(tp+fn) \quad (17)$$

where *tp* corresponds to the number of the pixels that were correctly classified as foreground and *fn* corresponds to the number of the pixels that were expected to be classified as foreground but were erroneously classified as background.

Harmonic average (F-measure) is : a combination of recall and precision metrics. The formula is the following:

$$F = 2 * \frac{precision * recall}{precision + recall}$$

Whether a pixel is correctly or not classified as part of the foreground (nuclei) or the background is based on the available ground truth. Ground truth is a marked version of the original image in which the nuclei are marked in black by experienced pathologists (Figure 15).

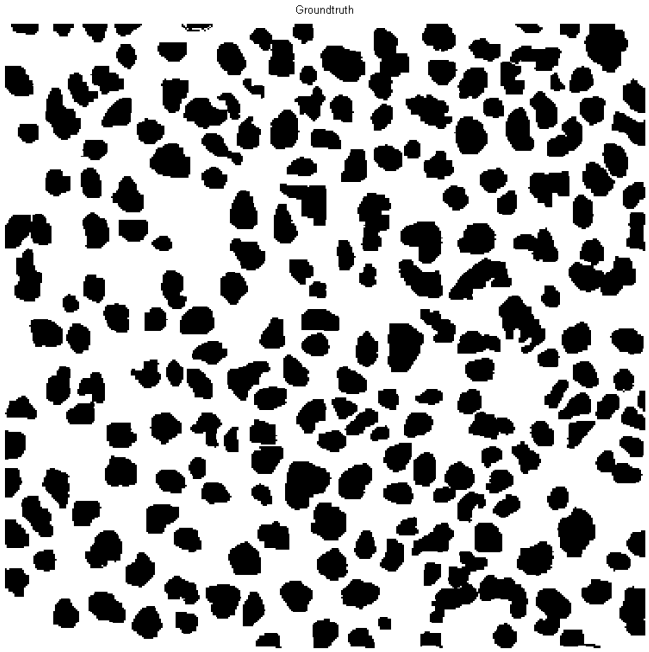

Figure 15.
Binary image called ground truth, where the nuclei (foreground) are marked in black and the background in white.

The results from segmentation expressed in the three pre-mentioned metrics are given in Table I.

Table I – Results of segmentation

		Recall	Precision	F-measure
GC	Image 1	0.898	0.370	0.524
	Image 2	0.933	0.365	0.524
	Image 3	0.886	0.397	0.549
	Average GC	***0.906***	***0.377***	***0.532***
Otsu	Image 1	0.799	0.322	0.459
	Image 2	0.837	0.317	0.460
	Image 3	0.774	0.346	0.478
	Average Otsu	***0.803***	***0.328***	***0.466***

8. Conclusion

This study presents an automated method for segmentation of FL images using graph cuts. The final goal in segmenting FL images is to identify the nuclei and therefore, the cells in the images. The regions of the nuclei which correspond to the regions of the cells should be included in the foreground of the final binary image. The reason the other cytological components (ECM and cytoplasm) are not taken into account while looking for

the center of a cell is due to the fact that the boarders of these regions are not clear due to the heat from stain.

For this reason, we needed to follow a path of applying sequentially GC on the grayscale images. In each iteration, one component was omitted from the foreground until the point where only the nuclei were considered as foreground. The final result presented in Figure 9 is satisfactory, since it is a clear image of cells with their boarders. GC can be applied to find the regions of nuclei, by using also k-means. GC should be applied on top of k-means, since it takes into account the locality of the pixels in relation to the whole image. Therefore, the image is better segmented than just applying k-means. However, we know that GC is still not able to split the touching cells that might be considered as one cell after segmentation. GC is an algorithm that can be efficiently applied on images with good contrast in colors. However, in case of the touching cells, the contrast is very poor. The conclusion is that GC is not suitable to find the individual cells among the touching ones in case of H&E stained FL images.

For this reason, a cell splitting algorithm should be applied on top of GC. In this case, a Watershed-based splitting algorithm was used. The result from this procedure, shown in Figure 11, is not optimal, since there are still some touching cells considered as a single one. Therefore, this issue needs to be further examined in the future.

When comparing the GC segmentation with simple Otsu thresholding, we see that GC segmentation gives more accurate results compared to Otsu thresholding (As seen in Table I the F-measure for GC is 0.532 while the F-measure for Otsu thresholding is 0.466). However, both fail in splitting the cells.

References

[1] R. Gonzalez, R.Woods, “Digital Imaging Processing”, Prentice Hall, New York, 2002.

[2] M.Kass,A.Witkin,D.Terzopoulos, “Snakes:Active Contour Models”,International Journal of Computer Vision,, 321-331,1988.

[3] D.Cohen,”On active contour models and balloons”,CVGIP: Image,1991.

[4] Blake, M. Israd, “Active Contours”, Springer, 1998.

[5] V.Caselles, R. Kimmel, G. Sapiro,“Geodesic Active Contours”, International Journal of Computer Vision 22(1), 61–79 (1997).

[6] A. Yezzi, S. Kichenassamy, A. Kumar, P. Olver, and A. Tannenbaum, “A geometric snake model for segmentation of medical imagery,” IEEE Trans. Med. Imag., vol. 16, pp. 199–209, Apr. 1997.

[7] E.N. Mortensen, W.A. Barrett, “Interactive segmentation with intelligent scissors”, Graph. Models Image Process, 60 (1998), pp. 349–384.

[8] Y.Boykov, G.Funka-Lea, “Graph Cuts and Efficient N-D Image Segmentation”, International Journal of Computer Vision 70(2), 109-131, 2006.

[9] Y.Gdalyahu, D.Weinshall, M.Werman,”Self organization in vision : stochastic clustering for image segmentation, perceptual grouping, and image database organization”, IEEE Trans. on Pattern Analysis and Machine Intelligence, 23(10):1053-1074,2001.

[10] D. Greig, B. Porteous, A.Seheult, Exact Maximum A Posteriori Estimation for Binary Images, J. Royal Statistical Soc., Series B, vol. 51, no. 2, pp. 271-279, 1989.

[11] Y.Boykov, O.Veksler,R.Zabih, “Fast approximate energy minimization via graph cuts”, IEEE transactions on pattern analysis and machine intelligence, vol.23, no.11, November 2001.

[12] S.Geman, D..Geman, ”Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images”, IEEE Trans.Pattern Analysis and Machine Intelligence, vol.6, pp.721-741, 1984.

[13] W.E.L. Grimson and T.Pavlidis, “Discontinuity Detection for Visual Surface Reconstruction”, Computer Vision, Grpahics and Image Processing, vol.30, pp.316-330, 1985.

[14] D.Terzopoulos, “Regularization of Inverse Visual Problems Involving Discontinuities”, IEEE Trans.Pattern and Analysis and Machine Intelligence, vol.8, pp.413-426,1986.

[15] R.H. Swendson and J. Wang, “Nonuniversal Critical Dynamics in Monte Carlo Simulations,” Physical Rev. Letters, vol. 58, no. 2, pp. 86-88, 1987.

[16] Y. Yuri and G. Lea, "Graph Cuts and Efficient N-D Image Segmentation," Journal of Computer Vision, 70, 109-131 (2006).

[17] Y. Yuri and M. Jolly, "Interactive Graph Cuts for optimal Boundary & Region Segmentation of Objects in N-D Images," International conference on computer vision, I, 105-112 (2001).

[18] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithm for energy minimization in vision," IEEE trans on PAMI, 26, 1124-1137 (2004).

[19] S. Li, Markov Random Field Modeling in Computer Vision. Springer-Verlag, 1995.

[20]Y.Boykov, O.Veksler, R.Zabih, “Markov Random Fields with Efficient Approximations”, Proceedings of IEEE conference on “Computer Vision and Pattern Recognition” (CVPR), 1998.

[21]G. Paschos, “Perceptually uniform color spaces for color texture analysis: an empirical evaluation”, IEEE Transaction on Image Processing, vol.10, pp.932-937, 2001.

Appendix - Code for segmenting an FL image using previously described methods

```
%% Code for segment the cancer cells in follicular lymphoma
images (H&E stained) using Graph Cuts
%%% Images acquired from both APTH medical center's database
and OSU Medical Center's database,as well as from
%%% OSU, Bioinformatics department's database

%%% General input: rgb color image
%%% General output: regions of centroblast (CB) cells in
follicular
%%% lymphoma image
%%% originally by Evgenios Kornaropoulos, EKETA-ITI & BMI-OSU
%%% Columbus, Ohio, The United States, 01/03/12 - 30/07/12

clear all;
clc;
close all;
pack;

%% Load an image (input) and check its channels

n = 5; % number of image in the folder
eval(['load cropped_images img_cropped_' num2str(n) ';']);
eval(['I_rgb_ui = img_cropped_' num2str(n) ';']);
I_rgb = double(I_rgb_ui);
eval(['clear img_cropped_' num2str(n) ';']);
f1 = figure(n);
imshow(I_rgb./255,[]);
maximize(f1);

%%% Checking input image
szrgb = size(I_rgb);
ht = szrgb(1);
wd = szrgb(2);
dp = szrgb(3);

total_pixels = ht*wd;

if (total_pixels <= 0)
    disp(strcat(fname, 'has no image data'));
end

if (dp ~= 3)
    disp(strcat(fname, 'is not a color image'));
end

% Get the gray-scale image
I_gray = rgb2gray(I_rgb_ui);

% Figure 1 - Original image and its gray-scale
```

```
f1 = figure(1);
subplot(1,2,1)
imshow(I_rgb./255,[]);
title(['\fontsize{18}RGB original image'])
% title('RGB original cropped image');
subplot(1,2,2)
imshow(I_gray,[]);
title('\fontsize{18}Gray-scale of image');
maximize(f1);

% check the RGB channels
I_red = I_rgb(:,:,1);
I_green = I_rgb(:,:,2);
I_blue = I_rgb(:,:,3);

% Figure 2 - R,G,B channels
f2 = figure(2);
subplot(1,3,1)
imshow(I_red,[]);
title('\fontsize{18}\color{red}Red \color{black}channel of
image');
subplot(1,3,2)
imshow(I_green,[]);
title('\fontsize{18}\color{green}Green \color{black}channel of
image');
subplot(1,3,3)
imshow(I_blue,[]);
title('\fontsize{18}\color{blue}Blue \color{black}channel of
image');
maximize(f2);

% check the Lab channels
cform = makecform('srgb2lab');
lab = applycform(I_rgb_ui,cform);
Lighting  = lab(:,:,1);
La = lab(:,:,2);
Lb = lab(:,:,3);

% Figure 3 - L,A,B channels
f3 = figure(3);
subplot(1,3,1)
imshow(Lighting,[]);
title('\fontsize{18}\color{yellow}Lighting \color{black}
channel');
subplot(1,3,2)
imshow(La,[]);
title('\fontsize{18}\color{cyan}A \color{black} channel');
subplot(1,3,3)
imshow(Lb,[]);
title('\fontsize{18}\color{magenta}B \color{black} channel');
maximize(f3);

% check how the values of the RGB channesl lie on 3D space to
see if
```

```
% k-means would be applicable (if they resemble as a cirle it
will succeed,
% otherwise it will fail)
sz = size(I_rgb_ui);
v_rgb = reshape(I_rgb, [prod(sz(1:2)) 3]);

% do the same for the Lab channels
v_lab = reshape(lab, [prod(sz(1:2)) 3]);

% Figure 4
% f4= figure(4);
% subplot(1,2,1)
% plot3(v_rgb(:,1),v_rgb(:,2),v_rgb(:,3));
% subplot(1,2,2)
% plot3(v_lab(:,1),v_lab(:,2),v_lab(:,3));
% maximize(f4);

%% Apply filters to the original image in order to remove
noise (preprocessing phase)

% 1st try Gaussian filtering
G1 = fspecial('gaussian',[14 14],2);
If1 = imfilter(I_gray,G1);

% 2nd try median filtering
If2 = medfilt2(I_gray,[5 5]);

% check the result
% Figure 5 - Checking result of filtering
f5 = figure(5);
subplot(1,3,1)
imshow(I_gray,[]);
title(['\fontsize{18}Gray-scale'])
% title('RGB original cropped image');
subplot(1,3,2)
imshow(If1,[]);
title('\fontsize{18}Filtered with Gaussina [14 14]');
subplot(1,3,3)
imshow(If2,[]);
title('\fontsize{18}Median filtered [5 5]');
maximize(f5);

% they all need some histrogram equalization
adhist_gray = adapthisteq(I_gray);
adhist_If1 = adapthisteq(If1);
adhist_If2 = adapthisteq(If2);

% check the result
% Figure 6 - Checking result of adapthisteq
f6 = figure(6);
subplot(1,3,1)
imshow(adhist_gray,[]);
title(['\fontsize{18}Histeq 1 (gray-scale)'])
% title('RGB original cropped image');
```

```
subplot(1,3,2)
imshow(adhist_If1,[]);
title('\fontsize{18}Histeq 2 (Gaussian [14 14])');
subplot(1,3,3)
imshow(adhist_If2,[]);
title('\fontsize{18}Histeq 3 (median filt [5 5])');
maximize(f6);

% !! The winner is median filtering !!

%% Remove the background (white regions), as well as the RBC

ch_red = I_rgb(:,:,1);
ch_green = I_rgb(:,:,2);
ch_blue = I_rgb(:,:,3);

%-- eliminate RBC
mask = (ch_red./(ch_red+ch_green+ch_blue))
RBC = mask>=0.45;
clear mask;
ind_RBC = find(RBC==1);

BACKGROUND = (ch_red>140 & ch_green>140 & ch_blue>140);

rbc_and_bckgr = BACKGROUND; % create a mix matrix of both
background and RBC
rbc_and_bckgr(ind_RBC) = 1;

mask_image = ~rbc_and_bckgr;

% figure 7
f7=figure(7);
imshow(((I_rgb.*double(repmat(mask_image,[1 1 3]))))./255),[]);
title('\fontsize{18}RBG image with backround and RBC in
black');
maximize(f7);

%-- now look on the rest(interested in) pixels
ind_good_pixels = find(mask_image);
ind_bad_pixels = find(mask_image==0);
ind_1st_good_pixels = ind_good_pixels;
ind_1st_bad_pixels = ind_bad_pixels;
mask_image_1st  = mask_image;

%% Apply 1st Graph Cut to segment foreground and background
using k-means (in Lab space)

% segment the image into 2 different regions (foreground and
background)
k = 2;

% color space distance
distance = 'sqEuclidean';
```

```
v_lab = double(v_lab);

% cluster the image colors into k regions
idx = mask_image(:); % assign label=1 to foreground pixels
idx = double(idx);
idx(ind_bad_pixels) = 2; % assign label=2 to background pixels
% create cluster centers by calculating the avg of each class
c(1,:) = mean(v_lab(ind_good_pixels,:),1);
c(2,:) = mean(v_lab(ind_bad_pixels,:),1);

% calculate the data cost per cluster center
Dc = zeros([sz(1:2) k],'single');

for ci=1:k
%     use covariance matrix per cluster
    icv = inv(cov(v_lab(idx==ci,:)));
    dif = v_lab - repmat(c(ci,:), [size(v_lab,1) 1]);

%     data cost is minus log likelihood of the pixel to belong
to each
%     cluster according to its RGB value
    Dc(:,:,ci) = reshape(sum((dif*icv).*dif./2,2),sz(1:2));

end

% cut the graph

% smoothness term:
% constant part
Sc = ones(k) - eye(k);

% spatialy varying part - do it once for every case
% Option 1 : use of gradient
% [Hc1 Vc1] =
gradient(imfilter(double(adhist_gray),fspecial('gauss',[5
5]),'symmetric'));
% [Hc2 Vc2] =
gradient(imfilter(double(adhist_If1),fspecial('gauss',[5
5]),'symmetric'));
% [Hc3 Vc3] =
gradient(imfilter(double(adhist_If2),fspecial('gauss',[5
5]),'symmetric'));

% Option 2 : use of Gaussian filter
g = fspecial('gauss', [14 14], sqrt(14));
dy = fspecial('sobel');
vf = conv2(g, dy, 'valid');

% Vc1 = zeros(sz(1:2));
% Hc1 = Vc1;
% Vc2 = zeros(sz(1:2));
% Hc2 = Vc2;
Vc3 = zeros(sz(1:2));
```

```
Hc3 = Vc3;

% for b=1:size(I_rgb,3)
%     Vc1 = max(Vc1, abs(imfilter(double(adhist_gray), vf,
'symmetric')));
%     Hc1 = max(Hc1, abs(imfilter(double(adhist_gray), vf',
'symmetric')));
%     Vc2 = max(Vc2, abs(imfilter(double(adhist_If1), vf,
'symmetric')));
%     Hc2 = max(Hc2, abs(imfilter(double(adhist_If1), vf',
'symmetric')));
    Vc3 = max(Vc3, abs(imfilter(double(adhist_If2), vf,
'symmetric')));
    Hc3 = max(Hc3, abs(imfilter(double(adhist_If2), vf',
'symmetric')));
% end

% gch1 = GraphCut('open', Dc, 10*Sc, exp(-Vc1*5), exp(-
Hc1*5));
% gch2 = GraphCut('open', Dc, 10*Sc, exp(-Vc2*5), exp(-
Hc2*5));
gch3 = GraphCut('open', Dc, 10*Sc, exp(-Vc3*5), exp(-Hc3*5));

% [gch1 L1] = GraphCut('expand',gch1);
% [gch2 L2] = GraphCut('expand',gch2);
[gch3 L3] = GraphCut('expand',gch3);

% gch1 = GraphCut('close', gch1);
% gch2 = GraphCut('close', gch2);
gch3 = GraphCut('close', gch3);

% L1 = ~L1;
% L2 = ~L2;
L3 = ~L3;

% fg_pixels1 = find(L1==1);
% bg_pixels1 = find(L1==0);
% fg_pixels2 = find(L2==1);
% bg_pixels2 = find(L2==0);
fg_pixels3 = find(L3==1);
bg_pixels3 = find(L3==0);

% check the result
% Figure 8 - Checking result of 1st graph cut
% f8 = figure(8);
% subplot(1,3,1)
% imshow(double(L1),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3,[]);
```

```
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f8);

% figure 8
f8 = figure(8);
imshow(L3,[]);
title('\fontsize{18}1st GC (median filt [5 5])');
maximize(f8);

% nowget rid of the small artifacts in the semgented image
% L1_new = bwareaopen(L1,9);
% L1_new = ~L1_new;
% L1_new = bwareaopen(L1_new,9);
% L1_new = ~L1_new;
%
% L2_new = bwareaopen(L1,9);
% L2_new = ~L2_new;
% L2_new = bwareaopen(L2_new,9);
% L2_new = ~L2_new;

L3_new = bwareaopen(L3,10);
L3_new = ~L3_new;
L3_new = bwareaopen(L3_new,10);
L3_new = ~L3_new;

% check the result
% Figure 9 -
% f9 = figure(9);
% subplot(1,3,1)
% imshow(double(L1_new),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2_new,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3_new,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f9);

% figure 9
f9 = figure(9);
imshow(L3_new,[]);
title('\fontsize{18}1st GC (median filt [5 5])');
maximize(f9);

ind_good_pixels = find(L3_new);
ind_bad_pixels = find(L3_new==0);
mask_image = L3_new;

%% Apply 2nd Graph Cut to segment extracellular and background
to cell regions, using k-means (in Lab space)

% first apply k-means to find the extracellular membrane
```

```
lab = applycform(I_rgb_ui,cform);
L  = lab(:,:,1);
La = lab(:,:,2);
Lb = lab(:,:,3);
L_new = L(ind_good_pixels);
La_new = La(ind_good_pixels);
Lb_new = Lb(ind_good_pixels);
I_new = [L_new La_new Lb_new];
k=3; % number of classes
distance = 'sqEuclidean'; % color space distance
[idx_kmeans c_kmeans] = kmeans(double(I_new), k, 'distance',
distance,'maxiter',200);
I_labelled = double(mask_image);
I_labelled(ind_good_pixels) = idx_kmeans;
c_s = (sum(c_kmeans,2))./3;
[val ind] = sort(c_s);
class1 = find(I_labelled==ind(3,1));%brighter
color==ECM(extra-cellular membrane)
ecm_and_rbc_and_bckgr = [ind_bad_pixels;class1];
ecm_and_rbc_and_bckgr = sort(ecm_and_rbc_and_bckgr);
mask_image(ecm_and_rbc_and_bckgr) = 0;
ind_good_pixels = find(mask_image);
ind_bad_pixels = ecm_and_rbc_and_bckgr;
idx2(ind_good_pixels) = 1;
idx2(ind_bad_pixels) = 2;

% segment the image into 2 different regions (foreground and
background)
k = 2;

% color space distance
distance = 'sqEuclidean';

% create cluster centers by calculating the avg of each class
c(1,:) = mean(v_lab(ind_good_pixels,:),1);
c(2,:) = mean(v_lab(ind_bad_pixels,:),1);

% calculate the data cost per cluster center
Dc = zeros([sz(1:2) k],'single');

for ci=1:k
%     use covariance matrix per cluster
    icv = inv(cov(v_lab(idx==ci,:)));
    dif = v_lab - repmat(c(ci,:), [size(v_lab,1) 1]);

%     data cost is minus log likelihood of the pixel to belong
to each
%     cluster according to its RGB value
    Dc(:,:,ci) = reshape(sum((dif*icv).*dif./2,2),sz(1:2));

end

% cut the graph
```

```
% gch1 = GraphCut('open', Dc, 10*Sc, exp(-Vc1*5), exp(-
Hc1*5));
% gch2 = GraphCut('open', Dc, 10*Sc, exp(-Vc2*5), exp(-
Hc2*5));
gch3 = GraphCut('open', Dc, 10*Sc, exp(-Vc3*5), exp(-Hc3*5));

% [gch1 L1] = GraphCut('expand',gch1);
% [gch2 L2] = GraphCut('expand',gch2);
[gch3 L3] = GraphCut('expand',gch3);

% gch1 = GraphCut('close', gch1);
% gch2 = GraphCut('close', gch2);
gch3 = GraphCut('close', gch3);

% L1 = ~L1;
% L2 = ~L2;
L3 = ~L3;

% fg_pixels1 = find(L1==1);
% bg_pixels1 = find(L1==0);
% fg_pixels2 = find(L2==1);
% bg_pixels2 = find(L2==0);
fg_pixels3 = find(L3==1);
bg_pixels3 = find(L3==0);

% check the result
% Figure 8 - Checking result of 1st graph cut
% f10 = figure(10);
% subplot(1,3,1)
% imshow(double(L1),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f10);

% figure 10
f10 = figure(10);
imshow(L3,[]);
title('\fontsize{18}2nd GC (median filt [5 5])');
maximize(f10);

% nowget rid of the small artifacts in the semgented image
% L1_new = bwareaopen(L1,9);
% L1_new = ~L1_new;
% L1_new = bwareaopen(L1_new,9);
% L1_new = ~L1_new;
%
% L2_new = bwareaopen(L1,9);
% L2_new = ~L2_new;
```

```
% L2_new = bwareaopen(L2_new,9);
% L2_new = ~L2_new;

L3_new = bwareaopen(L3,10);
L3_new = ~L3_new;
L3_new = bwareaopen(L3_new,10);
L3_new = ~L3_new;

% check the result
% Figure 11 -
% f11 = figure(11);
% subplot(1,3,1)
% imshow(double(L1_new),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2_new,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3_new,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f11);

% figure 11
f11 = figure(11);
imshow(L3_new,[]);
title('\fontsize{18}2nd GC (median filt [5 5])');
maximize(f11);

ind_good_pixels = find(L3_new);
ind_bad_pixels = find(L3_new==0);
mask_image = L3_new;

%% Apply Graph Cut for 3rd (and last) time to segment the
nuclei of the cells, using k-means (in Lab space)

% first apply k-means to find the extracelluar membrane
lab = applycform(I_rgb_ui,cform);
L  = lab(:,:,1);
La = lab(:,:,2);
Lb = lab(:,:,3);
L_new = L(ind_good_pixels);
La_new = La(ind_good_pixels);
Lb_new = Lb(ind_good_pixels);
I_new = [L_new La_new Lb_new];
k=2; % number of classes
distance = 'sqEuclidean'; % color space distance
[idx_kmeans c_kmeans] = kmeans(double(I_new), k, 'distance',
distance,'maxiter',200);
I_labelled = double(mask_image);
I_labelled(ind_good_pixels) = idx_kmeans;
c_s = (sum(c_kmeans,2))./3;
[val ind] = sort(c_s);
class1 = find(I_labelled==ind(2,1));%brighter color==cytoplasm
```

```
cytoplasm_and_ecm_and_rbc_and_bckgr = [ind_bad_pixels;class1];
cytoplasm_and_ecm_and_rbc_and_bckgr =
sort(cytoplasm_and_ecm_and_rbc_and_bckgr);
mask_image(cytoplasm_and_ecm_and_rbc_and_bckgr) = 0;
ind_good_pixels = find(mask_image);
ind_bad_pixels = cytoplasm_and_ecm_and_rbc_and_bckgr;
idx3(ind_good_pixels) = 1;
idx3(ind_bad_pixels) = 2;

% segment the image into 2 different regions (foreground and
background)
k = 2;

% color space distance
distance = 'sqEuclidean';

% create cluster centers by calculating the avg of each class
c(1,:) = mean(v_lab(ind_good_pixels,:),1);
c(2,:) = mean(v_lab(ind_bad_pixels,:),1);

% calculate the data cost per cluster center
Dc = zeros([sz(1:2) k],'single');

for ci=1:k
%     use covariance matrix per cluster
    icv = inv(cov(v_lab(idx==ci,:)));
    dif = v_lab - repmat(c(ci,:), [size(v_lab,1) 1]);

%     data cost is minus log likelihood of the pixel to belong
to each
%     cluster according to its RGB value
    Dc(:,:,ci) = reshape(sum((dif*icv).*dif./2,2),sz(1:2));

end

% cut the graph

% gch1 = GraphCut('open', Dc, 10*Sc, exp(-Vc1*5), exp(-
Hc1*5));
% gch2 = GraphCut('open', Dc, 10*Sc, exp(-Vc2*5), exp(-
Hc2*5));
gch3 = GraphCut('open', Dc, 10*Sc, exp(-Vc3*5), exp(-Hc3*5));

% [gch1 L1] = GraphCut('expand',gch1);
% [gch2 L2] = GraphCut('expand',gch2);
[gch3 L3] = GraphCut('expand',gch3);

% gch1 = GraphCut('close', gch1);
% gch2 = GraphCut('close', gch2);
gch3 = GraphCut('close', gch3);

% L1 = ~L1;
% L2 = ~L2;
```

```
L3 = ~L3;

% fg_pixels1 = find(L1==1);
% bg_pixels1 = find(L1==0);
% fg_pixels2 = find(L2==1);
% bg_pixels2 = find(L2==0);
fg_pixels3 = find(L3==1);
bg_pixels3 = find(L3==0);

% check the result
% Figure 12 - Checking result of 1st graph cut
% f12 = figure(12);
% subplot(1,3,1)
% imshow(double(L1),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f12);

% figure 12
f12 = figure(12);
imshow(L3,[]);
title('\fontsize{18}3rd GC (median filt [5 5])');
maximize(f12);

% nowget rid of the small artifacts in the segmented image
% L1_new = bwareaopen(L1,9);
% L1_new = ~L1_new;
% L1_new = bwareaopen(L1_new,9);
% L1_new = ~L1_new;
%
% L2_new = bwareaopen(L1,9);
% L2_new = ~L2_new;
% L2_new = bwareaopen(L2_new,9);
% L2_new = ~L2_new;

L3_new = bwareaopen(L3,15);
L3_new = ~L3_new;
L3_new = bwareaopen(L3_new,15);
L3_new = ~L3_new;

% check the result
% Figure 13 -
% f13 = figure(13);
% subplot(1,3,1)
% imshow(double(L1_new),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
```

```
% imshow(L2_new,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3_new,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f13);

% figure 13
f13 = figure(13);
imshow(L3_new,[]);
title('\fontsize{18}3rd GC (median filt [5 5])');
maximize(f13);

% overlay_images(I_gray, L3_new)

%% Compare the final result with the available ground truth

L3_inverse = ~L3_new;
% figure 12
f13=figure(13);
imshow(L3_inverse);
title('\fontsize{18}Nuclei after GC');
maximize(f13);

GT = imread('gt_im5.tif');
gt_red = GT(:,:,1);
gt_green = GT(:,:,2);
gt_blue = GT(:,:,3);

%-- eliminate RBC
mask_gt = (gt_red==255 & gt_green==255 & gt_blue==255);

% figure 14
f14=figure(14);
imshow(mask_gt);
title('\fontsize{18}Ground truth');
maximize(f14);

L3_pixels = L3_inverse(:);
mask_gt_pixels = mask_gt(:);

num_of_real_cell_pixels = find(mask_gt==0);
num_of_real_bkgrd_pixels = find(mask_gt==1);

num_of_found_cell_pixels = find(L3_pixels==0);
num_of_found_bkgrd_pixels = find(L3_pixels==1);

tp = find(L3_pixels(num_of_real_cell_pixels)==0);
fp = find(L3_pixels(num_of_real_bkgrd_pixels)==1);
tn = find(mask_gt_pixels(num_of_found_cell_pixels)==1);
fn = find(mask_gt_pixels(num_of_found_bkgrd_pixels)==0);

recall = (size(tp,1)/(size(tp,1)+size(fn,1)));
```

```
precision = (size(tp,1)/(size(tp,1)+size(fp,1)));
f_measure = 2*((precision*recall)/(precision+recall));

disp(strcat('Recall of GC segmentation : ', num2str(recall)));
disp(strcat('Precision accuracy of GC segmentation : ',
num2str(precision)));
disp(strcat('F-measure with GC segmentation : ',
num2str(f_measure)));

%% Since the nuclei regions are identified, split the nuclei

cellNuclear = L3_new;
areaThresh = 150;
cellNuclear = bwareaopen(cellNuclear, areaThresh, 8);
cellNuclear = imfill(cellNuclear, 'holes');

bwTmp = cellNuclear;
bwTmp = imfill(cellNuclear, 'holes');
% apply watershed transformation
D = imcomplement(bwdist(~bwTmp));
% suppress insignificant minima
D = imhmin(D, 2);
mask = ones(size(D));
% watershed lines in a matrix
mask(watershed(D)==0)=0;
% separate the regions
cellNuclear = mask & cellNuclear;

cellNuclear = bwareaopen(cellNuclear, areaThresh,8);
cellNuclear = imfill(cellNuclear, 'holes'); %%% cell nuclear
regions after watershed transform and

% figure 15
f15=figure(15);
imshow(cellNuclear);
title('\fontsize{18}Splitted nuclei after GC');
maximize(f15);

%% Compare again the final result with the available ground
truth

cellNuclear_inv = ~cellNuclear;
% figure 16
f16=figure(16);
imshow(cellNuclear_inv);
title('\fontsize{18}Splitted nuclei after GC');
maximize(f16);

cellNuclear_pixels = cellNuclear_inv(:);

new_num_of_found_cell_pixels = find(cellNuclear_pixels==0);
new_num_of_found_bkgrd_pixels = find(cellNuclear_pixels==1);

tp_new = find(cellNuclear_pixels(num_of_real_cell_pixels)==0);
```

```
fp_new =
find(cellNuclear_pixels(num_of_real_bkgrd_pixels)==1);
tn_new =
find(mask_gt_pixels(new_num_of_found_cell_pixels)==1);
fn_new =
find(mask_gt_pixels(new_num_of_found_bkgrd_pixels)==0);

recall_new = (size(tp_new,1)/(size(tp_new,1)+size(fn_new,1)));
precision_new =
(size(tp_new,1)/(size(tp_new,1)+size(fp_new,1)));
f_measure_new =
2*((precision_new*recall_new)/(precision_new+recall_new));

disp(strcat('New recall of GC segmentation : ',
num2str(recall_new)));
disp(strcat('New precision accuracy of GC segmentation : ',
num2str(precision_new)));
disp(strcat('New F-measure with GC segmentation : ',
num2str(f_measure_new)));

% clearvars -except I_rgb I_rgb_ui lab I_gray adhist_If2
mask_image_1st ind_1st_good_pixels ind_1st_bad_pixels v_rgb
v_lab cform
clearex('I_rgb','I_rgb_ui','lab
I_gray','adhist_If2','mask_image_1st','ind_1st_good_pixels','i
nd_1st_bad_pixels','v_rgb','v_lab','cform');
ind_good_pixels = ind_1st_good_pixels;
ind_bad_pixels = ind_1st_bad_pixels;
mask_image = mask_image_1st;

%% Apply 1st Otsu to segment foreground (cytoplasm,nuclei) and
background(ECM,RBC,white regions)

otsu_threshold = graythresh(adhist_If2(ind_good_pixels));
L3_s = adhist_If2(ind_good_pixels)/255 > otsu_threshold;
L3 = ~mask_image;
L3(ind_good_pixels)=L3_s;

% L1 = ~L1;
% L2 = ~L2;
L3 = ~L3;

% fg_pixels1 = find(L1==1);
% bg_pixels1 = find(L1==0);
% fg_pixels2 = find(L2==1);
% bg_pixels2 = find(L2==0);
fg_pixels3 = find(L3==1);
bg_pixels3 = find(L3==0);

% check the result
% Figure 8 - Checking result of 1st graph cut
% f8 = figure(8);
% subplot(1,3,1)
% imshow(double(L1),[]);
```

```
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f8);

% figure 17
f17 = figure(17);
imshow(L3,[]);
title('\fontsize{18}1st Otsu (median filt [5 5])');
maximize(f17);

% now get rid of the small artifacts in the segmented image
% L1_new = bwareaopen(L1,9);
% L1_new = ~L1_new;
% L1_new = bwareaopen(L1_new,9);
% L1_new = ~L1_new;
%
% L2_new = bwareaopen(L1,9);
% L2_new = ~L2_new;
% L2_new = bwareaopen(L2_new,9);
% L2_new = ~L2_new;

L3_new = bwareaopen(L3,10);
L3_new = ~L3_new;
L3_new = bwareaopen(L3_new,10);
L3_new = ~L3_new;

% check the result
% Figure 9 -
% f9 = figure(9);
% subplot(1,3,1)
% imshow(double(L1_new),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2_new,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3_new,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f9);

% figure 18
f18 = figure(18);
imshow(L3_new,[]);
title('\fontsize{18}1st Otsu (median filt [5 5])');
maximize(f18);

ind_good_pixels = find(L3_new);
```

```
ind_bad_pixels = find(L3_new==0);
mask_image = L3_new;

%% Apply 2nd Otsu to segment ECM,cytoplasm and background to
cell regions

% first apply k-means to find the extracellular membrane
lab = applycform(I_rgb_ui,cform);
L  = lab(:,:,1);
La = lab(:,:,2);
Lb = lab(:,:,3);
L_new = L(ind_good_pixels);
La_new = La(ind_good_pixels);
Lb_new = Lb(ind_good_pixels);
I_new = [L_new La_new Lb_new];
k=3; % number of classes
distance = 'sqEuclidean'; % color space distance
[idx_kmeans c_kmeans] = kmeans(double(I_new), k, 'distance',
distance,'maxiter',200);
I_labelled = double(mask_image);
I_labelled(ind_good_pixels) = idx_kmeans;
c_s = (sum(c_kmeans,2))./3;
[val ind] = sort(c_s);
class1 = find(I_labelled==ind(3,1));%brighter
color==ECM(extra-cellular membrane)
ecm_and_rbc_and_bckgr = [ind_bad_pixels;class1];
ecm_and_rbc_and_bckgr = sort(ecm_and_rbc_and_bckgr);
mask_image(ecm_and_rbc_and_bckgr) = 0;
ind_good_pixels = find(mask_image);
ind_bad_pixels = ecm_and_rbc_and_bckgr;
idx2(ind_good_pixels) = 1;
idx2(ind_bad_pixels) = 2;

otsu_threshold = graythresh(adhist_If2(ind_good_pixels));
L3_s = adhist_If2(ind_good_pixels)/255 > otsu_threshold;
L3 = ~mask_image;
L3(ind_good_pixels)=L3_s;

% L1 = ~L1;
% L2 = ~L2;
L3 = ~L3;

% fg_pixels1 = find(L1==1);
% bg_pixels1 = find(L1==0);
% fg_pixels2 = find(L2==1);
% bg_pixels2 = find(L2==0);
fg_pixels3 = find(L3==1);
bg_pixels3 = find(L3==0);

% check the result
% Figure 8 - Checking result of 1st graph cut
% f10 = figure(10);
% subplot(1,3,1)
% imshow(double(L1),[]);
```

```
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f10);

% figure 19
f19 = figure(19);
imshow(L3,[]);
title('\fontsize{18}2nd Otsu (median filt [5 5])');
maximize(f19);

% now get rid of the small artifacts in the segmented image
% L1_new = bwareaopen(L1,9);
% L1_new = ~L1_new;
% L1_new = bwareaopen(L1_new,9);
% L1_new = ~L1_new;
%
% L2_new = bwareaopen(L1,9);
% L2_new = ~L2_new;
% L2_new = bwareaopen(L2_new,9);
% L2_new = ~L2_new;

L3_new = bwareaopen(L3,10);
L3_new = ~L3_new;
L3_new = bwareaopen(L3_new,10);
L3_new = ~L3_new;

% check the result
% Figure 11 -
% f11 = figure(11);
% subplot(1,3,1)
% imshow(double(L1_new),[]);
% title(['\fontsize{18}GC 1 (gray-scale)'])
% % title('RGB original cropped image');
% subplot(1,3,2)
% imshow(L2_new,[]);
% title('\fontsize{18}GC 2 (Gaussian [14 14])');
% subplot(1,3,3)
% imshow(L3_new,[]);
% title('\fontsize{18}GC 3 (median filt [5 5])');
% maximize(f11);

% figure 20
f20 = figure(20);
imshow(L3_new,[]);
title('\fontsize{18}2nd Otsu (median filt [5 5])');
maximize(f20);

ind_good_pixels = find(L3_new);
```

```
ind_bad_pixels = find(L3_new==0);
mask_image = L3_new;

% overlay_images(I_gray, L3_new)

%% Since the nuclei regions are identified, split the nuclei

cellNuclear = L3_new;
areaThresh = 150;
cellNuclear = bwareaopen(cellNuclear, areaThresh, 8);
cellNuclear = imfill(cellNuclear, 'holes');

bwTmp = cellNuclear;
bwTmp = imfill(cellNuclear, 'holes');
% apply watershed transformation
D = imcomplement(bwdist(~bwTmp));
% suppress insignificant minima
D = imhmin(D, 2);
mask = ones(size(D));
% watershed lines in a matrix
mask(watershed(D)==0)=0;
% separate the regions
cellNuclear = mask & cellNuclear;

cellNuclear = bwareaopen(cellNuclear, areaThresh,8);
cellNuclear = imfill(cellNuclear, 'holes'); %%% cell nuclear
regions after watershed transform and

% figure 21
f21=figure(21);
imshow(cellNuclear);
title('\fontsize{18}Splitted nuclei after Otsu');
maximize(f21);

%% Compare the final result with the available ground truth

L3_inverse = ~L3_new;
% figure 22
f22=figure(22);
imshow(L3_inverse);
title('\fontsize{18}Splitted nuclei after Otsu');
maximize(f22);

GT = imread('gt_im5.tif');
gt_red = GT(:,:,1);
gt_green = GT(:,:,2);
gt_blue = GT(:,:,3);

%-- eliminate RBC
mask_gt = (gt_red==255 & gt_green==255 & gt_blue==255);

num_of_cell_pixels = find(mask_gt==0);
num_of_bkgrd_pixels = find(mask_gt==1);
```

```
L3_pixels = L3_inverse(:);
mask_gt_pixels = mask_gt(:);
cells_found = find(L3_pixels==0);
bkgrd_found = find(L3_pixels==1);
tp = find(L3_pixels(num_of_cell_pixels)==0);
fp = find(L3_pixels(num_of_bkgrd_pixels)==1);
tn = find(mask_gt_pixels(cells_found)==1);
fn = find(mask_gt_pixels(bkgrd_found)==0);

% COMP = L3_inverse-mask_gt;
% m1= find(COMP==-1);
% m2= find(COMP==0);
% m3= find(COMP==1);
%
% COMP_d = I_rgb;
% sz=size(COMP_d);
% COMP_d = reshape(COMP_d, [sz(1)*sz(2) 3]);
% COMP_d(m1,1) = 1;
% COMP_d(m1,2) = 0;
% COMP_d(m1,3) = 0;
% COMP_d(m2,1) = 0;
% COMP_d(m2,2) = 0;
% COMP_d(m2,3) = 0;
% COMP_d(m3,1) = 1;
% COMP_d(m3,2) = 1;
% COMP_d(m3,3) = 1;
% COMP_d = reshape(COMP_d,[sz(1:2) 3]);
% f15=figure(15);
% imshow(COMP_d);
% title('Fig11 after filling the holes');
% maximize(f15);
%
% clc;
% ACCURACY =
(size(m2,1)/(size(m1,1)+size(m3,1)+size(m2,1)))*100;
% disp(strcat('Overall accuracy of Otsu segmentation % ',
num2str(ACCURACY)));

recall = (size(tp,1)/(size(tp,1)+size(fn,1)));
precision = (size(tp,1)/(size(tp,1)+size(fp,1)));
f_measure = 2*((precision*recall)/(precision+recall));
disp(strcat('Recall of Otsu segmentation : ',
num2str(recall)));
disp(strcat('Precision accuracy of Otsu segmentation :  ',
num2str(precision)));
disp(strcat('F-measure with Otsu segmentation :  ',
num2str(f_measure)));
```