Project Assignment #4
Image Segmentation

Vineet Joshi (vineetj), Frank Lin (fjlin), Prateek Tandon (ptandon1)

Overview

The goal of this project was to study different image segmentation techniques. In Part 1, image segmentation techniques that are implemented in the ITK software package are tested. The techniques we tested specifically are the Confidence Connected Segmentation (SCC) and Shape Detection Level Set Filter (SSDLS) techniques. In the second part, several graph-cut and active-contour image segmentation algorithms are tested. The different image segmentation techniques are then compared with one another.

Part I: Image Segmentation using MAT-ITK

B.1 MAT-ITK and image data

The MAT-ITK package and images '60x_02.tif' and 'Blue0001.tif' were downloaded.

B.2.1 Segmentation of static images

The SCC and SSDLS techniques were applied to both test images ('60x_02.tif' and 'Blue0001.tif').

SCC: Confidence Connected Segmentation ('B2_1_SCC.m')

In the file 'B2_1_SCC.m', the SCC technique was applied to both test images. The different images files can be selected under "Choose image file". The inputs required for the MATITK SCC function are the parameters, the image, and seed. The parameters required include the multiplier, number of iterations, and replace value. For this part, we tested a variety of different parameters and decided that a multiplier of 4 was ideal with 5 iterations and a replace value of 1. The images were first converted to 3D for use in MATITK. Finally, we tested different seeds that decided that the middle seed for both images were ideal. The following are images generated.

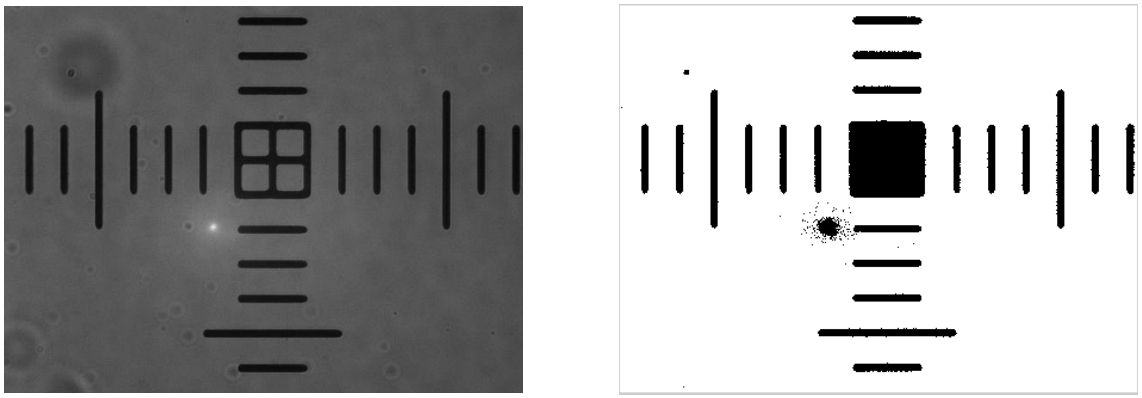


Figure 1A. ‘60x_02.tif’and Segmented ‘60x_02.tif’using SCC

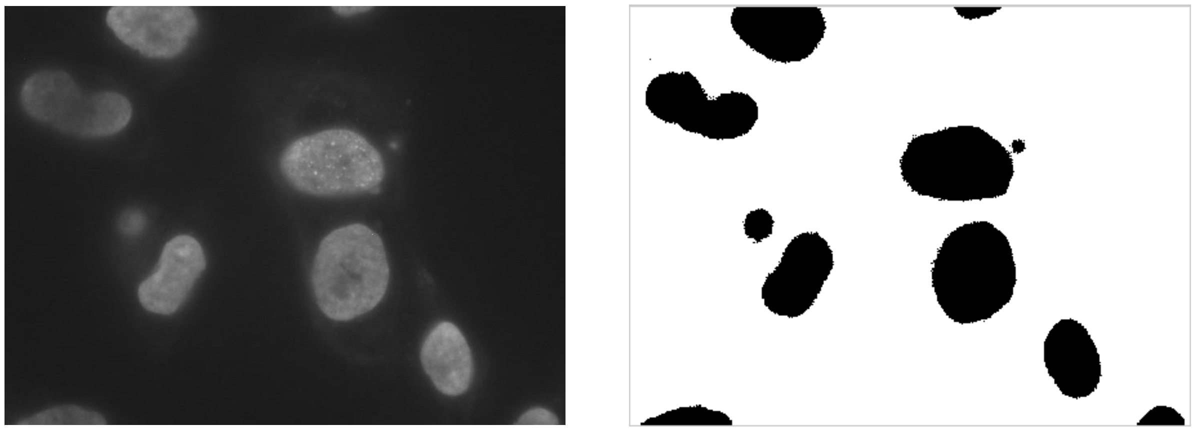


Figure 1B. ‘Blue0001.tif’and Segmented‘Blue0001.tif’using SCC

SSDLS: Shape Detection Level Set Filter (‘B2_1_SSDLS.m’)

In the file ‘B2_1_SSDLS.m’, the SSDLS technique was applied to both test images. The different images files can be selected under “Choose image file”. The inputs required for the MATITK SSDLS function are the parameters, the image, and the gradient of the image. The parameters required include propagation scaling, curvature scaling, set maximum RMS error, and set number of iterations. For this part, we tested a variety of different parameters and decided that the default setting of (1, 1, 0.02, 1) was adequate as varying values didn’t change the result by much. The images were first converted to 3D for use in MATITK and the gradient was calculated by adding the gradient for x and gradient for y together. The following are images generated.

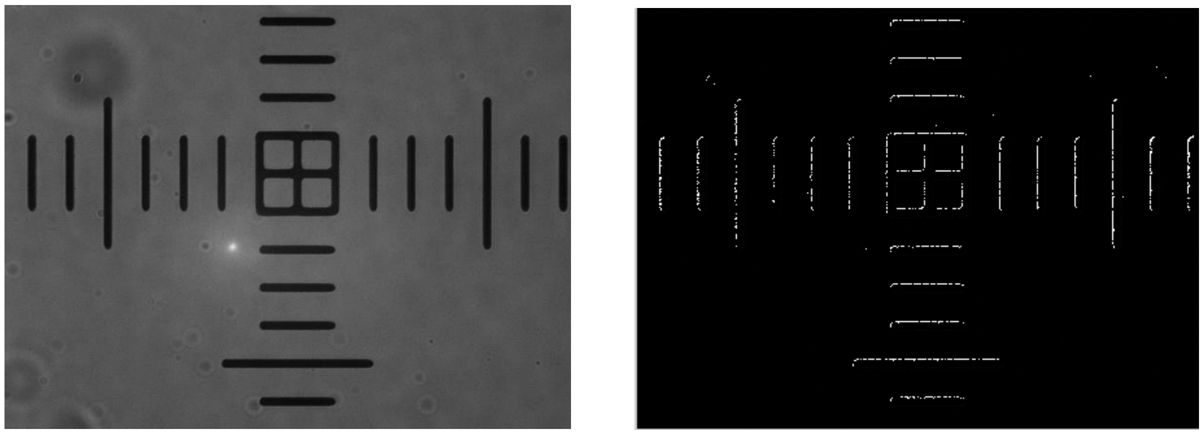


Figure 2A. ‘60x_02.tif’ and Segmented ‘60x_02.tif’using SSDLS

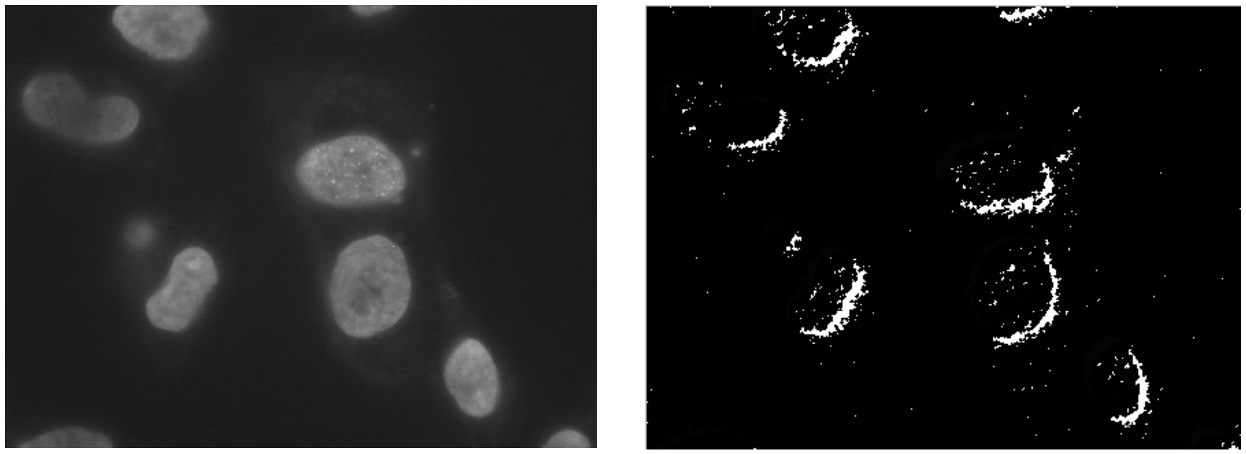


Figure 2B. ‘Blue0001.tif’ and Segmented‘Blue0001.tif’using SSDLS

B.2.2 Segmentation of image series

The segmentation techniques SCC and SSDLS were applied to a time-lapse image sequence (‘Mito_GFP_a01’) and AVI files were generated of the results.

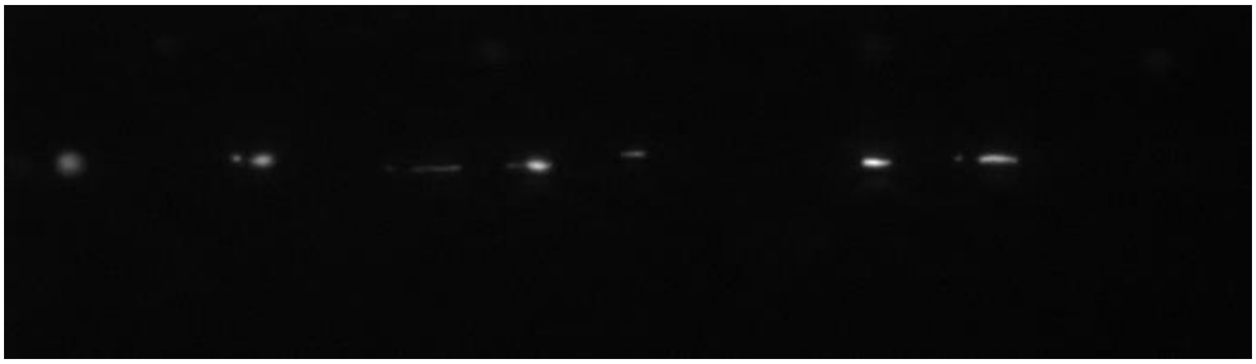


Figure 3A. ‘MitoGFP_LgtGal4_a01r01s02001.tiff’

SCC: Confidence Connected Segmentation (‘B2_2_SCC.m’)

In the file “B2_2_SCC.m”, the SCC technique was applied to the time-lapse image. Parameters of (10, 5,1) were used along with a seed of (40,90,1). The results are saved in the folder titled ‘B2_2_SCC_Results”. An AVI file is also generated titled ‘B2_2_SCC_Results.avi’.

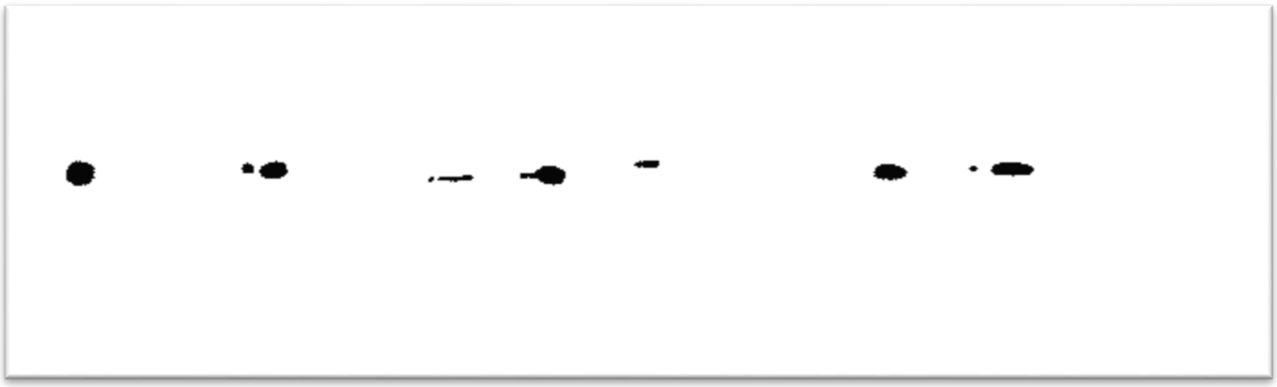


Figure 3B. ‘SCC_Result_1.tiff”

SSDLS: Shape Detection Level Set Filter (‘B2_2_SSDLS.m’)

In the file “B2_2_SSDLS.m”, the SSDLS technique was applied to the time-lapse image. Parameters of (1.4, 1,2 ,20) were used and gradient was calculated the same way as in the previous section. The results are saved in the folder titled ‘B2_2_SSDLS_Results”. An AVI file is also generated titled ‘B2_2_SSDLS_Results.avi’.

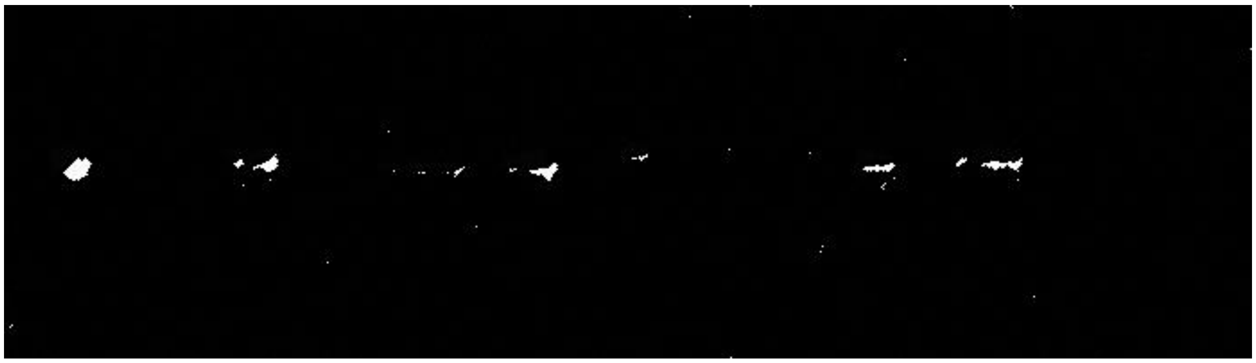


Figure 3C. ‘SSDLS_Result_1.tiff"

B.2.3 Theoretical background

A theoretical background of the SCC and SSDLS techniques are given in this section.

<u>SCC Theory</u>

SCC or the Confidence Connected Segmentation uses connectivity to segment image pixels. Prior to the start of this segmentation process, we must remove noise in the image using a filter as it interferes with the quality of segmentation process. Typically, any edge preserving smoothing filter can be used for this purpose.

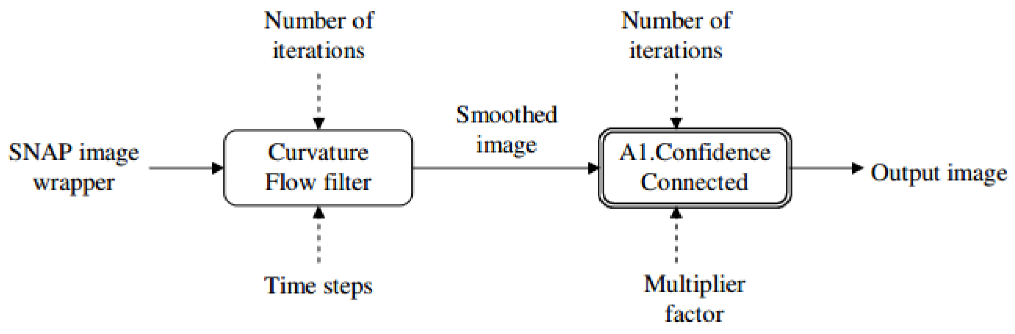
The algorithm extracts a set of pixels that are connected based on pixel intensities that are statistically consistent with a seed point. The mean and variance are calculated across the neighborhood of every seed point (8-connected, 26-connected, etc.). Then only the neighboring pixels whose values lie within the confidence interval for this seed are grouped together. Confidence interval is defined as Mean $\pm$ Multiplier*S.D.

$$I(x,y) \in (\mu - M * \sigma, \mu + M * \sigma)$$

The Multiplier (M) controls the confidence interval's width and hence size of the grouping. When the intensity values across a segment follow a Gaussian distribution, a Multiplier value of 2.5 would give a 99% confidence interval (group with 99% of neighbors that were earlier connected). Smaller M will result in smaller upper and lower intensity bounds which may restrict the inclusion of pixels with slightly different intensities lying in the same segment. A large M will result in a large intensity interval which may result in the inclusion of pixels that are part of different segments.

The above grouping is the initial phase of the segmentation. After this phase, the mean and variance (S.D.) are re-calculated, this time using the pixels from the grouping in the previous segmentation. This is in contrast with using the pixels in the neighborhood of the seed point like in the previous step. The segmentation (grouping) is then recalculated using these refined estimates for the mean and variance. The entire process is repeated for a fixed number of iterations that are controlled by the user. If an image had low homogeneity then it requires a higher number of iterations while image with higher homogeneity requires lesser iterations. At the end the final segmentation is returned as the final output.

There is a restriction on the lower and upper thresholds of the confidence interval during every iteration. They must lie within the valid numeric limits of the input data pixel type. The limits may also be adjusted to contain the intensity of the initial seed.



**Source: http://undergraduate.csse.uwa.edu.au/year4/Current/Students/Files/2004/PeterEnnis/CorrectedDissertation.pdf*

SSDLS Theory

SSDLS or ‘Shape Detection Level Set Filter’ uses a level set method segmentation filter. An initial contour is propagated outwards or inwards until it "sticks" to the shape boundaries. This involves the use of a level set speed function that is derived from a user supplied edge potential map.

The filter takes in two inputs, an ‘initial level set’ and ‘feature image’. The initial level set consists of a real image. The feature image consists of an edge potential map or edge features of the image. The edge potential map has values close to zero in regions near the edges (high image gradient) and values close to one the inside the shapes (regions with relatively constant intensity).

The edge potential is derived from the image gradient of the real image using the formula,

$$g(I) = \frac{1}{1 + |(\nabla * G)(I)|}$$

$$g(I) \cong exp - |(\nabla * G)(I)|$$

where I is image intensity and $(\nabla * G)$ is the derivative of Gaussian operator. This is known as the Shape Detection Level Set Function and it is a subclass of the generic Level Set Function.

Parameter tuning is important to get the right results. For example, the ‘Propagation Scaling’ parameter can switch propagation from outward direction (positive parameter value) to inward direction (negative parameter value). Propagation Scaling and Curvature Scaling parameters can be can be used to adjust the smoothness of the resulting contour. Curvature Scaling parameter should be assigned a positive value for proper function of the algorithm.

The algorithm outputs a single, scalar, real-valued image. The insides of the segmentation regions are represented by negative values in the output image and outsides of the segmentation regions are represented by positive values. The zero crossovers in the output image correspond to the position of the propagation front.

Part II: Image segmentation using graph-cut and active contour

C.1.1 Graph cut based image segmentation

Two graph-cut algorithms (______________) are identified, the codes were downloaded, and the algorithms were tested on the images from section B (‘60x_02.tif’ and ‘Blue0001.tif’).

Normalized Cut

Efficient graph-based segmentation

C.1.2 Active contour based image segmentation

Two active contour based image segmentation algorithms (______________) are identified, the codes were downloaded, and the algorithms were tested on the images from section B (‘60x_02.tif’ and ‘Blue0001.tif’).

Active Contour based on level set method-DRLSE
Active Contour based on Gradient Vector Flow

C.1.3 Theoretical background

In this section, the principles of the algorithms from C.1.1 and C.1.2 are discussed. The algorithms used in the projects are compared and contrasted.

Normalized Cut

The Normalized Cut Algorithm extracts the global impression of an image rather than focus on local features and their consistencies. Image segmentation is treated as a graph partitioning problem and a criterion (normalized cut) is used for segmenting the graph. This criterion measures the total dissimilarity between groups and the total similarity within groups.

First, construct a graph G is by taking each pixel as a node and connecting them with an edge. The weight set on an edge connecting two nodes is a measure of the similarity between them. $(D - W)x = \lambda Dx$ is solved for the eigenvectors with the smallest eigenvalues. Then, the eigenvector with the second smallest eigenvalue is used to bipartition the graph. Finally, based on the information, it is decided whether the current partition should be subdivided and the segmented pairs are recursively repartitioned if necessary.

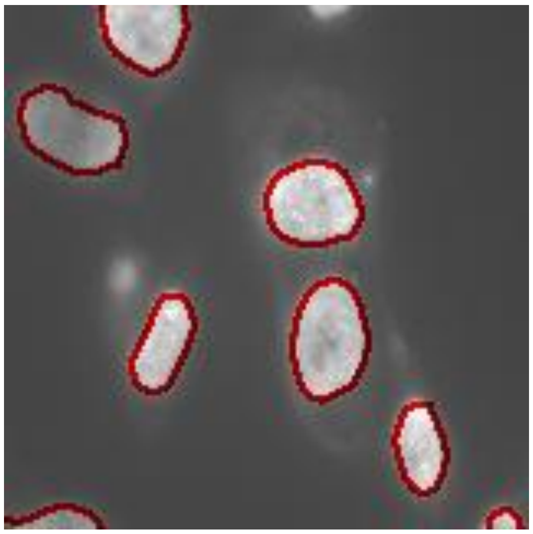
The Normalized Cut Algorithm is based on the fact that a graph, G = (V, E), can be partitioned into two disjoint sets A, B by removing edges between the two sets. The degree of dissimilarity can be computed as total weight of edges that have been removed, also known as a cut. The cut of A, B is defined as:

$$cut(A,B) = \sum_{w \epsilon A, v \epsilon B} w(u,v)$$

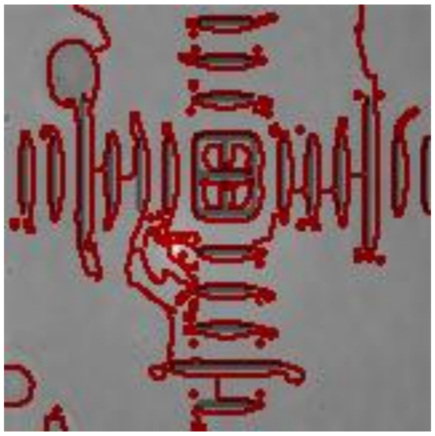
In order to avoid the bias for partitioning out small set of points, a different measure of disassociation was used. The cut cost as a fraction of the total edge connections to all the nodes in the graph (Ncut) was used instead. The Ncut of A, B is defined as:

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

The association (A, V) and (B, V) are the total weights of edges connecting nodes within A, V and B, V. The overall goal of the algorithm is to minimize the disassociation between groups and maximizing association within groups using the Ncut as the criteria.



Ncut image with number of segments = 8, sampleRadius = 10, sample_rate = 6.0, edgeVariance = 0.05



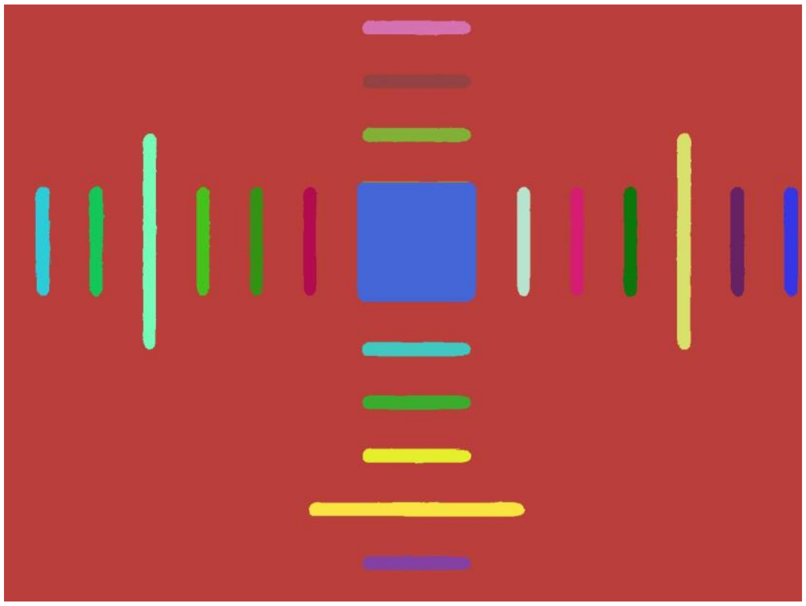
Ncut image with number of segments = 25, sampleRadius = 15, sample_rate = 9.0, edgeVariance = 0.017

Efficient graph-based segmentation

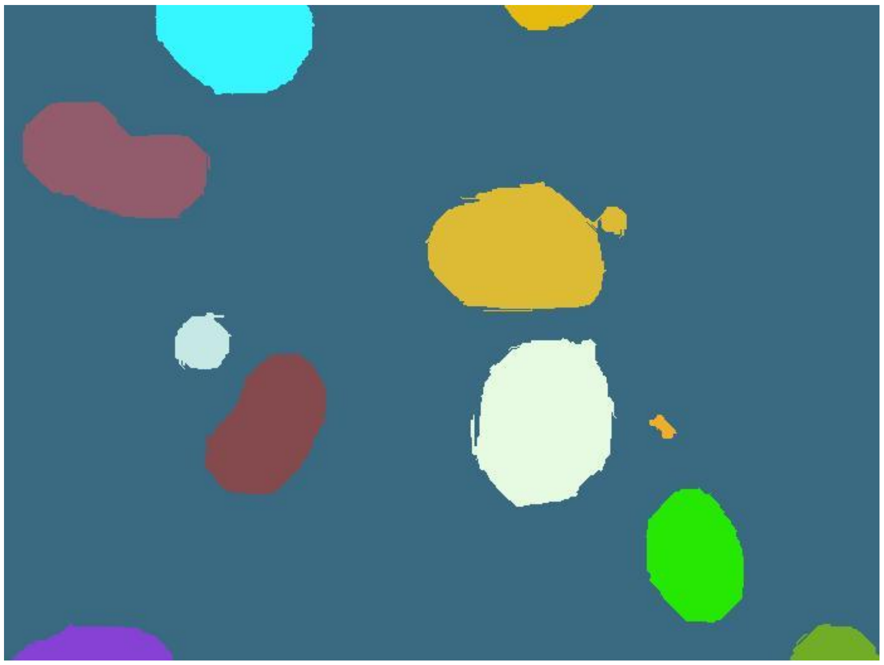
This algorithm is an efficient segmentation algorithm**,** based on a predicate for measuring evidence for a boundary between two regions. This algorithm runs linearly in the number of edges in the graph, makes greedy decisions, and produces segmentations that satisfy global properties. It preserves details in low-variability image regions while ignoring detail in high-variability regions.

A graph-based segmentation was used in the algorithm and a predicate for evaluating whether or not there is evidence for a boundary between two regions of an image is used. The predicate is based on the dissimilarity between elements along the boundary of two regions relative to a measure of dissimilarity among neighboring elements within each of the two regions. It compares the inter-component differences to the within component difference. The internal difference and difference between two regions are defined.

The algorithm first sorts the edges of a graph into non-decreasing weights. Then different segmentations (for all edges) are constructed given the previous segmentation. In the construction step, if vertices connected by the qth edge in the ordering are disjoint and the segmentation and weight are small compared to the internal difference of both components, merge the components; otherwise do nothing When all the segmentations are iterated through, return the final segmentation. This algorithm obeys the global properties of being neither too fine nor too coarse when using the predicate.



Efficient segmentation algorithm Parameters: sigma = 0.9, K = 450, min = 100



Efficient Segmentation algorithm parameters: sigma = 1.2, K = 1300, min = 100

DRLSE

In general level set formulations, the level set function develops irregularities during the evolution. This can cause many numerical errors and possible destroy the stability of evolution. Re-initialization is applied periodically to replace the degraded level set function, but the entire process is not ideal. Distance regularized level set evolution (DRLSE), takes care of these errors through the use of a new level set formulation in which the regularity of the level set function is maintained during the set evolution. The level set evolution is “derived as the gradient flow that minimizes energy functional with a distance regularization term and an external energy that drives the motion of the zero level set toward desired locations.” Through the use of DRLSE, the need for re-initialization is not needed and thus avoiding the numerical errors.

DRLSE image with parameters: timestep = 5, iter_outer = 50, lambda = 5, alfa = 2.2, epsilon = 1.0, sigma = 1.2

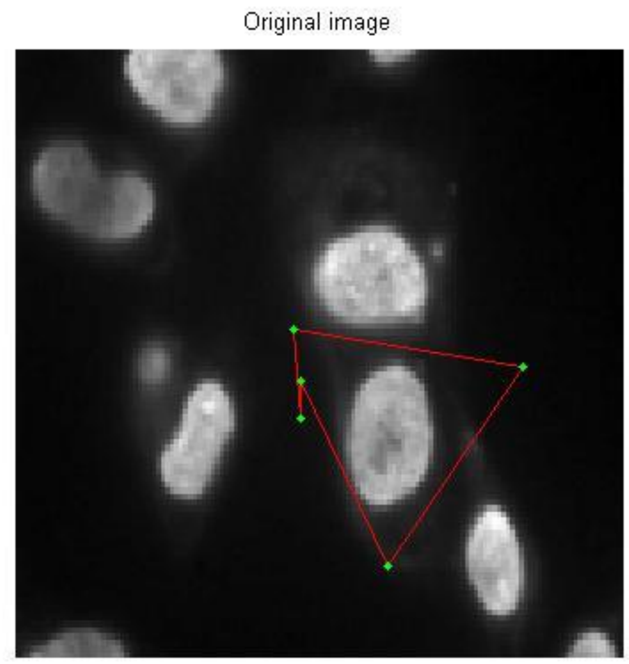
DRLSE image with parameters: timestep = 5, iter_outer = 60, lambda = 5, alfa = 2.2, epsilon = 1.0, sigma = 1.2

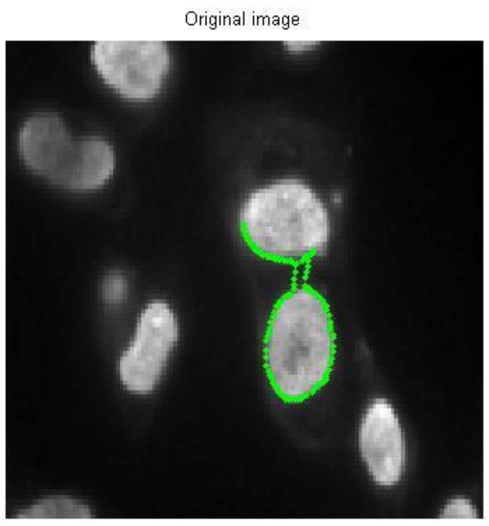
Active Contour based on Gradient Vector Flow:

Active contours are defined as curves within an image domain that can move under the influence of forces coming either internally (coming from within the curve itself) or external (computed from the image data). Under the influence of these forces, the contour conforms to an object boundary or other desired features within an image. They are widely used in many applications, including edge detection and segmentation.

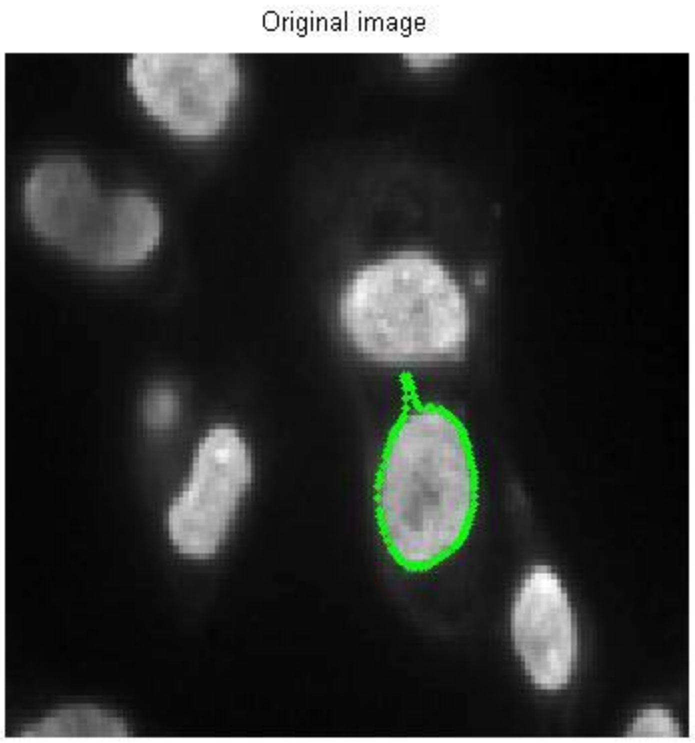
There are two basic active contour models: parametric active contours and geometric active contours. In this project we implement the parametric active contour. Parametric active contours use parametric curves within an image and move them toward desired features in the image. The forces that drive these curves towards the features are called potential forces, which are represented as negative gradient of a potential function. These belong to the category of external forces. Some internal forces are also used to keep the curves from bending excessively from the potential forces.

$$E_{\text{ext}}^{(2)}(x, y) = -|\nabla[G_\sigma(x, y) * I(x, y)]|^2$$
$$\mathbf{F}_{\text{ext}}^{(\mathrm{p})} = -\nabla E_{\text{ext}}.$$

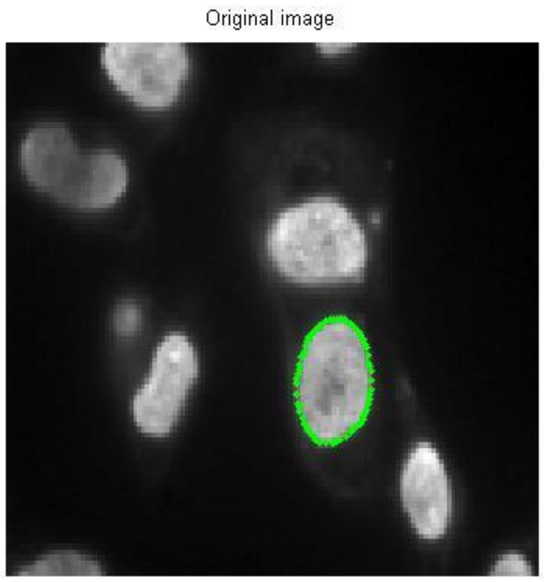




Deformation using default parameters:



Parameters: Alpha = 0.3 beta = 0.5, gamma = 2, kappa = 0.5



Parameters:

Alpha = 0.5 beta = 0, gamma = 1, kappa = 1

Comparisons

Having obtained the results as above, it is very evident that the ITK based algorithms perform better as compared to contour based algorithms for controlling the number of iterations is very much hazy. While a particular number of iterations may identify an obscure object very efficiently, it may turn the process really slow with loss in more obvious features in the image. The ITK based algorithms have been developed with this in mind and documentation is available in a decent amount too to follow.

The ITK based results are much clearer when the images are segmented and noise reduction is possible to a very great extent. Having said that, Graph cut and vector flow based algorithms provide much better intermediate visualization to assess how the parameters are affecting the process of segmentation. While all the algorithms are relatively fast, the slowest one was found to be DRLSE.

References

1. Shi, Jianbo, and Jitendra Malik. "Normalized cuts and image segmentation." Pattern Analysis and Machine Intelligence, IEEE Transactions on 22.8 (2000): 888-905.
2. Felzenszwalb, Pedro F., and Daniel P. Huttenlocher. "Efficient graph-based image segmentation." International Journal of Computer Vision 59.2 (2004): 167-181.
3. C. Li, C. Xu, C. Gui, and M. D. Fox, "Distance Regularized Level Set Evolution and its Application to Image Segmentation", IEEE Trans. Image Processing, vol. 19 (12), pp. 3243-3254, 2010
4. C. Xu and J. L. Prince, ``Snakes, Shapes, and Gradient Vector Flow," *IEEE Transactions on Image Processing*, 7(3), pp. 359-369, March 1998.

5. http://www.itk.org/
6. http://designest.de/2009/11/matitk-additional-documentation/