

A Novel Immersion Simulation Based Self-Organizing Transform with Application to Single-Cell Segmentation from Microscopy Images

Quanli Wang

Institute for Genome Sciences & Policy
Duke University, Durham, NC 27708, USA

Key words. Immersion simulation, image segmentation, watershed transform, cell counting.

Summary

Reliable cell segmentation plays an important role in biological imaging studies, though continues to be challenging due to the complex nature of many imaging scenes. The approach here uses a novel immersion simulation based self-organizing (ISSO) transform, an automated method for image segmentation. The method allows users to customize the immersion simulation process via user-defined or default self-organizing functions to incorporate prior information into segmentation. The transformed images usually have desirable features that can be further segmented using existing methods, with substantially improved segmentation results. The connections between the ISSO transform and existing watershed transforms is described. A Size Filter based on the ISSO transform is implemented and applied to various images and benchmark microscopy datasets from recent studies. With benchmark error rates well below those reported results in the literature, the comparison with other algorithms clearly demonstrates the benefits and flexibility of the ISSO method on various types of images.

Introduction

Segmentation, which means partitioning of an image into background and objects of interest such as cell nuclei, is central to many biological studies (Elowitz et al, 2002; Megason et al, 2007). For example, accurate segmentation of cell nuclei from microscopy images is usually a first and crucial step in single-cell measurement studies (Rosenfield et al, 2005; Gordan et al, 2007). The process usually involves information on aspects such as cell shape and size, gray level intensity distribution etc., and uses this either explicitly or implicitly in defining algorithms for cell/object identification (Carpenter et al., 2006; Li et al., 2008; Wang et al., 2008; Wang et al. 2009).

Multiple approaches to single-cell image segmentation exist. Global/adaptive thresholding is usually straightforward for situations where cell nuclei are well-separated and have relatively evenly distributed background intensities (Otsu 1979; Wu et al, 1995; Wu et al. 2000). A review of adaptive thresholding methods can be found in (Sezgin et al, 2004). Watershed methods (Vincent & Soille, 1991; Meyer, 1994), which are parameter-free and often used for more complicated problems, are implemented in commercial or open source software image processing packages such as Matlab and ImageJ (Abramoff et al, 2004); these are based on a recursive immersion simulation process. However,

watershed methods, which usually yield mosaic image representations of original images, often suffer from serious over-segmentation problem. Marker-controlled and hierarchical watershed methods are then often employed as remedies to the over-segmentation problem by either providing strong prior information (“marker”) to preprocess the image before applying watershed method (Beuchar, 1992), or by applying merging rules to post-process over-segmented regions from the watershed transform (Beuchar, 1994; Najman & Schmitt, 1996; Umesh Adiga & Chauduri, 2001; Mao et al, 2006). The success of these methods is heavily dependent on the choice of initial markers or the rules specific to the image under investigation, and it has proven difficult to define generally useful methods (Lin et al, 2005; Chen et al, 2006). Another popular approach uses active contours, such as snakes and various level set approaches, in which objects are represented as smooth contours or level sets (Kass et al., 1988; Osher & Sethian, 1988; Caselles et al. 1997; Chan & Vese 2001). These algorithms typically start with an initial contour approximation and evolve/iterate according to the minimization of some energy function that is defined on both contours and the enclosed regions. Other algorithms address specific problems by exploiting the unique features of given images. For example, Wu et al. (1998) used an elliptical model for cell detection and segmentation; Berlemont et al. (2007) used a feature-adapted beamlet model to detect DNA filaments in fluorescent microscopy; Wang et al. (2007) proposed a method that combines intensity and shape information for cell segmentation; Li et al. (2008) developed a gradient flow tracking method to address the segmentation problem of touching cell nuclei.

Despite active research in the field, cell segmentation remains a challenging problem due to the diversity and complexity of microscopy images across cell types and application contexts, with weak contrast, touching nuclei, diffused background and varying size and shape of cell nuclei all posing challenges to existing methods. Generality and portability of existing methods is also a challenge.

We report here on a novel Immersion Simulation based Self-Organizing (ISSO) transform that generalizes the watershed algorithm to tackle some of the above challenges; it does this by allowing users to define a self-organizing function to model how the objects of interest should behave or self-organize based on prior information such as size, shape or border smoothness during a recursive immersion simulation process. We further define and implement a special case of the ISSO transform – the Size Filter. We demonstrate the use and flexibility of the algorithm in three representative examples: a real-life image from the Matlab image processing package, an irregular-shaped binary particle image, and a microscopy image of budding yeast colonies. This is followed by more discussion on Size Filter customization, performance evaluation, parameter selection and parallel implementation to address various advantages of using ISSO Size Filter transforms for image segmentation. Performance is validated and compared with other leading algorithms using two benchmark microscopic image sets from the Broad Institute Bioimage Benchmark Collection (<http://www.broad.mit.edu/bbbc/>). With benchmark error rates well below the published results, we conclude that the proposed algorithm substantially improves segmentation results, while being general and portable.

Method

Overview of watershed algorithms and immersion simulation

As the proposed algorithm is closely related to various watershed algorithms, this section provides a brief introduction to some important aspects of watershed transform development and implementation (Roderick & Meijster, 2000).

Any grayscale image can be considered as a topographic surface or landscape. The original watershed transform is a segmentation method that simulates a landscape being flooded by rain. Rain falling on the surface of a landscape will flow to some regional minima, with all rain falling on a given catchment basin flowing toward the same regional minimum. Watershed lines that separate catchment basins are treated as boundaries between regions for the purpose of segmentation. An alternative model is based on immersion simulation. If the surface is flooded from its regional minima and the merging of water from different sources is prevented by building dams, the image is partitioned into two different sets: catchment basins and watershed lines (dams), which result in a natural segmentation of the original image.

Vincent and Soille (1991) proposed an efficient algorithmic implementation based on immersion simulations. Formally, let I be a grayscale image under study, with h_{\min} and h_{\max} being the smallest and largest values taking by I in its domain D_I respectively. Let $T_h(I)$ stand for the threshold set function of I at level h :

$$T_h(I) = \{ p \in D_I \mid I(p) \leq h \}, \quad h_{\min} \leq h \leq h_{\max}. \quad (1)$$

Then the following recursion between gray levels of image defines the watershed algorithm based on immersion simulations:

$$\begin{cases} X_{h_{\min}} = \{ p \in D_I \mid I(p) = h_{\min} \}, \\ X_{h+1} = MIN_{h+1} \cup IZ_{T_{h+1}(I)}(X_h), \quad h_{\min} \leq h < h_{\max}. \end{cases} \quad (2)$$

where X_h is the union of the set of catchment basins computed at level h , MIN_h denotes the union of all regional minima at level h , and $IZ_A(B)$ is the union of geodesic influence zones of the connected components of B with $B \subseteq A$. Full definition of $IZ_A(B)$ can be found in Vincent and Soille (1991).

Introducing prior information into the watershed transform using markers

As a parameter-free algorithm, it is well known that the watershed transform described above tends to produce over segmentation when applied to real images (Jalba et al, 2004; Jung, 2007; Cristoforetti et al, 2008). One reason for over segmentation is due to the existing high frequency noise in the input image, which is usually dealt with by preprocessing steps such as filtering or smoothing. Yet another reason is intrinsic to the watershed transform itself as the final segmentation result is completely driven by the regional minima, regardless any other prior information about the objects of interest. This problem is usually dealt with by introducing a set of regions called markers in a

marker image M_I (Beucher, 1992), based on features specific to the problem. By assigning h_{\min} to the points in the markers, and imposing the markers as regional minima, the input image I is modified as I_M to only have regional minima at markers at level h_{\min} . As a result, the watershed algorithm now produces watershed lines only around the markers. Using this approach, the recursive watershed algorithm defined in (2) can be rewritten as

$$\begin{cases} X_{h_{\min}} = \{p \in M_I \mid I_M(p) = h_{\min}\} = T_{h_{\min}}, \\ X_{h+1} = IZ_{T_{h+1}(I_M)}(X_h), \quad h_{\min} \leq h < h_{\max}. \end{cases} \quad (3)$$

There are other approaches to dealing with the over segmentation issue, such as creating a hierarchical model on catchment basins using various criteria to facilitate region merging. For example, Beucher (1994) proposed a hierarchical watershed algorithm using “dynamics” to define a tree model to merge the catchment basins based on given dynamics scales.

Motivation to generalize the watershed transforms

A typical application of watershed algorithms involves most, if not all, of four steps:

- Step 1: Smooth the original image using morphological operations such as opening/closing, or applying filters to remove high frequency components.
- Step 2: Generate a set of initial markers for image background and objects of interest and modify the image to make markers the only regional minima. This step is usually done manually or automatically based on some problem-specific criteria.
- Step 3: Apply marker-controlled watershed algorithm to the modified image using markers from Step 2 as seeds, to allow the markers to expand, thus segmenting the images into regions.
- Step 4: Establish hierarchical models on segmentation result from Step 3 based on some scale that is appropriate to the given problem, and perform further region merging/splitting to obtain the final segmentation.

The success of the parameter-free watershed transform in Step 3 depends on either Step 2 or Step 4 to provide a context that takes advantage of some specific features of the problem. The watershed transform cannot be tailored to directly incorporate prior information such as size, shape etc. about the object of interest. Motivated by this, we propose a new Immersion Simulation based Self-Organizing transform that generalizes the watershed transform by allowing the user to provide a self-organizing function to integrate some object modeling directly into the immersion simulation process. The aim is to improve control over the segmentation process itself, instead of modeling the input markers or over segmented result regions.

Immersion Simulation based Self-Organizing transforms

Let $T_h(I)$ be the threshold set of an image I at level h as defined in (1). We define the Immersion Simulation based Self-Organizing transform (ISSO) by the following recursion between gray levels of the image:

$$\begin{cases} X_{h_{\min}} = SO(0, T_{h_{\min}}(I), \theta), \\ A_{h_{\min}} = X_{h_{\min}}, \\ X_{h+1} = SO(X_h, T_{h+1}(I), \theta), \quad h_{\min} \leq h < h_{\max}, \\ A_{h+1} = AGR(A_h, X_{h+1}, \phi), \quad h_{\min} \leq h < h_{\max}, \end{cases} \quad (4)$$

where h_{\min} , h_{\max} , X_h are the same as in (2), $SO(B_1, B_2, \theta)$ is any function that maps binary images B_1 and B_2 into another binary image of the same size using a given parameter vector θ , A_{h+1} is a gray scale image, and $C = AGR(A, B, \phi)$ is any function that maps a gray scale image A and a binary image B into another gray scale image C using a given parameter vector ϕ , such that

$$\forall p \in D_I, A(p) > 0 \Rightarrow C(p) > 0 \quad (5)$$

By definition (4), ISSO takes a two-step approach at each immersion level h using two status variables X_h and A_h respectively, with X_h keeping the updated level-to-level segmentation result, and A_h recording cumulated transform results up to the current level. Condition (5) is designed to guarantee that, during the immersion simulation process, the immersion at a higher level does not invalidate the previous result. Two functions SO , short for self-organizing, and AGR , short for aggregation, are designed to update X_h and A_h respectively, and each plays a distinct role in the process. First, the function SO acts as an object modeling tool to decide how the result from the previous level h should update/expand itself, or self-organize, based on the newly available information T_{h+1} , the threshold set at current level $h+1$. A parameter vector θ is provided to incorporate the prior information regarding the specific segmentation problem into this updating process. Secondly, the function AGR acts as a moderator to decide how to combine together the cumulative transform result A_h from previous levels and the newly updated segmentation result X_{h+1} at the current level, possibly using the prior information provided by the parameter vector ϕ .

Typically, the final transform result from (4) is in the form of a grayscale image, which has to be further segmented using existing methods. We will, however, demonstrate that the transformed images usually have very desirable features that can be easily segmented using standard existing methods.

ISSO transform examples

Based on (4) and (5) we give two examples that are special cases of ISSO transform. We use the first example to demonstrate that the marker-controlled watershed algorithm is a

special case of ISSO transform and then propose a novel ISSO Size Filter as another example, using it in the remaining sections to evaluate our approach.

- ISSO example 1: Marker-controlled watershed algorithm

The recursive transform defined by (4) collapses to (3) if we set $AGR(A, B, \phi) = B$ and $SO(B_1, B_2, \theta) = IZ_{B_2}(B_1)$, which makes the watershed transform a special case of ISSO.

However, the proposed formulation (4) does provide opportunities for embedding object modeling and prior information into the immersion simulation process and makes it more easily adapted to specific image segmentation problems.

- ISSO example 2: Size Filter

The size of an object within a given image is usually the most recognizable feature other than intensity values for many image segmentation problems. The following realization of ISSO gives a very simple example that exploits this feature.

Suppose B_2 is a binary image made of k connected components Z_1, Z_2, \dots, Z_k with corresponding areas of P_1, P_2, \dots, P_k in terms of the number of pixels, and that $\theta = (size_{\min}, size_{\max})$ is a vector defined by two given values $size_{\min}$ and $size_{\max}$, while ϕ is a vector of zero length. Then defining the SO and AGR functions as

$$\begin{cases} SO(B_1, B_2, \theta) = \bigcup_{\{t | size_{\min} \leq P_t < size_{\max}\}} Z_t, \\ AGR(A, B, \phi) = A + B, \end{cases} \quad (6)$$

gives a realization of the ISSO transform. In this simple case, the SO function simply filters out the connected components in B_2 (the threshold set) that are too large or too small, and AGR function simply aggregates the result by adding up the filtered binary masks for the threshold set at all levels. We call this realization (6) an ISSO Size Filter as it only uses the minimum and maximum sizes for object modeling.

We present three examples that involve directly applying the Size Filter (6) for segmentation problems in the following section and give more discussions and examples on customizing the Size Filter in a later section.

Representative segmentation examples using the ISSO Size Filter

Segmentation Example 1: Identifying pears in a gray scale real-life image

We first give an example of applying the ISSO Size Filter to segment an image from Matlab Image Processing Toolbox. Fig 1a gives the gray scale input image of many touching pears. The goal is to identify/separate pears from the background and also from

each other. Fig 1b is the typical resulting mosaic image after applying the watershed transform to the gradient magnitude of (a). It is evident that the image is highly over segmented due to the blemishes and high frequency noise. Fig 1c is obtained using the marker-controlled watershed transform as suggested by Matlab image analysis experts (<http://www.mathworks.com/products/demos/image/watershed/ipexwatershed.html>). The foreground markers are generated by a multi-step procedure, which involves gradient magnitude calculation using Sobel edge filter, image smoothing by "opening by reconstruction" and "closing by reconstruction" morphological operations with disk structure elements of 20 and 5 respectively, and also thresholding to remove regional maxima that is less than 20 pixels. The background markers are calculated by applying watershed transform to the distance map of the Otsu thresholding of the original image (a). Fig 1d, is the result $A_{h_{\max}} > 0$ after applying the proposed ISSO Size Filter (6) with $\theta = (500, 17500)$, i.e, we assume that all the pears to be identified will have a size of at least 500 pixels and at most 17500 pixels, which are roughly visual estimates of the pear sizes in the image. Fig 1c and Fig 1d are both using random colors to indicate the identified pears and the segmentation results are superimposed on top of the original image. It is evident that the ISSO Size Filter identifies more pears correctly in a more coherent matter.

Segmentation Example 2: Segmentation of a binary particle image

We use an example here from Sun et al (2009), who proposed a 6-step adaptive watershed algorithm to segment irregular-shaped binary images based on the concept of overlap parameter. This demonstrates the application of the ISSO Size Filter to such binary images. The proposed procedure is implemented in three steps:

- Step 1: Calculate the distance map of the input binary image.
- Step 2: Apply the ISSO Size Filter to the distance map and extract local maxima as markers from the result. Note that, we set parameter $size_{\max}$ to a very large value here to just filter out objects that are too large.
- Step 3: Segment the distance map from Step 1 according to the markers from Step 2, using the marker-controlled watershed algorithm.

While the first and the third steps are standard for most binary segmentation problems, the second step is the most important; it uses the size filter to reduce the spurious local maxima to achieve satisfactory result as shown in Fig 2.

Fig 2a is the input irregular-shaped binary particle image, courtesy of Sun et al (2009). Fig 2b shows the result from the conventional watershed algorithm using the Matlab watershed transform, which happens to be the special case for Size Filter with $size_{\min} = 0$. The over segmentation is evident here. Fig 2c and Fig 2d give results using the Size Filter with parameters $size_{\min} = 25$ and $size_{\min} = 100$ respectively. Fig 2e gives in Y-axis the total number of segmented objects against in X-axis the only parameter $size_{\min}$. Further investigation shows that the proposed algorithm : 1) produces over-segmented results

when $size_{min}$ is smaller than 7; 2) produces under-segmentation results when $size_{min}$ is larger than 157; 3) gives the same result as shown in Fig 2c for any parameter between 8 and 52; 4) gives the same result as shown in Fig 2d for any parameter between 53 and 157. It is evident that, the algorithm is relatively insensitive to the choice of parameters and produces very stable and satisfying results in a wide range of parameters. Testing on many different binary images suggests that a default setting of $size_{min} = 10$ usually suppresses most spurious local minima and yields good segmentation results.

Segmentation Example 3: cell segmentation of microscopy image of budding yeast

Cell segmentation of clustered budding yeast from a microscopy image has been challenging but critical for many biological studies (Elowitz et al, 2002). Here we use the Size Filter to both identify the foreground and segment the foreground into individual cells for such images. A three step procedure is used:

- Step 1: Apply the Size Filter to enhance the contrast between the foreground (regions of cell clusters) and the background;
- Step 2: Identify the foreground by thresholding the result from Step 1;
- Step 3: Use the procedure described in Example 2 to segment the binary foreground for individual budding yeast cells.

Fig 3a shows the input testing microscopic image of budding yeast colonies, generally shown as the dark and round regions in the image and tightly clustered. Fig 3b gives the result $A_{h_{max}}$ after Step 1, using the trivial parameter $\theta = (1, a/4)$, where a is the total number of pixels in the image. Fig 3c shows the extracted foreground after Step 2 by suppressing all maxima in Fig 3b whose height is less than 20 using the Matlab function `ihmax` (any number between 10 and 30 will give very similar result). Fig 3d gives the final segmentation result after Step 3 using the default parameter setting.

In this example, we use a very simple global thresholding method to convert the transformed result in Step 1 into a binary image in Step 2. In case of dealing with a series of similar images, other methods such as Local Histogram Equalization or Rank Transform on Hybrid images (Wang et al, 2009) can be employed to avoid manually choosing a global threshold. Such examples are included in the software package accompanying this paper and freely available as web-based supporting material.

Discussions and Algorithm evaluations

Customizing ISSO Transforms

The Size Filter (6) defined above provides a simple example demonstrating the ability to incorporate prior information such as the size about the objects of interest into the immersion simulation process to shape the final results. By customizing the SO and/or

AGR functions according to the unique features of a given problem, we can apply this approach to a broad range of imaging problems.

- *Example using customized SO function:* in case we want to remove the "salt & pepper" noise from an image, we can customize the *SO* function in (6) to filter out any object in the threshold sets that is larger than a given minimum size such as 1. The transform yields a mask for small objects, or the potential candidate pixels where "salt & pepper" noise occurs. By applying a median filter to the original image, but only to pixels on the mask, we can effectively remove most "salt & pepper" noise with blurring the input image. We illustrate this in Fig 4, where Fig 4a shows a typical image contaminated by "salt & pepper" noise. Fig 4b is the result of applying a median filter, a commonly used method for such situation, Fig 4c is the binary mask for "salt & pepper" locations using the above mentioned method and Fig 4d is the final result of applying the median filter to the original image, but only to the masked pixels in Fig 4c. Compared to Fig 4b, Fig 4d is much less blurred as expected.
- *Example using customized AGR function:* if we instead (a) customize the *AGR* function in (6) such that

$$\begin{cases} \phi = h, \\ AGR(A, B, \phi) = A.*(1 - B) + \phi * B, \end{cases} \quad (7)$$

where h is the immersion level, " $.*$ " denotes the element-wise product, and (b) filter out the objects that are too small, the resulting transform will be a smoothed image with smaller objects - such as the blemishes in Segmentation Example 1 - being suppressed, but with larger objects such as pears unaffected; see demonstration in Fig 4e.

We can also incorporate shape-related prior information into the *SO* function. For example, for most cell nuclei segmentation problems we can assume that the nuclei have somewhat smooth borders or have no holes. This can be reflected in the *SO* function by applying a morphological opening/closing to the threshold set for border smoothing and by filling holes before or after filtering out unwanted objects based on size. We give examples in the following section using two benchmark image sets from the Broad Bioimage Benchmark Collection; in these examples we calculate benchmarks and make comparisons with two leading software packages: CellProfiler (Carpenter et al, 2006) and Gradient Flow Tracking method (Li et al, 2008).

Validations Using Benchmark Image Sets

We apply the customized Size Filter above for cell nuclei segmentation to two image sets from the Broad Bioimage Benchmark Collection (BBBC). To test and evaluate existing or new algorithms for bioimage analysis, the Broad Institute at MIT set up a collection of freely downloadable microscopy images, as well as some types of "ground truth" (expected result) obtained by human counters. The site also defines suitable benchmarks on how closely each algorithm matches the ground truth for cell counting, segmentation and etc. We use two datasets on cell nuclei counting from microscopy images of Human

HT29 Colon Cancer 1 cells (human) and Drosophila KC 167 1 cells (fly) These two datasets all have published benchmark results from the leading software package CellProfiler; and the relatively large variations in ground truth assessments from human observers suggest that these are non-trivial and challenging test images; see <http://www.broad.mit.edu/bbbc/>.

We summarize our comparison results through Fig 5 and Fig 6. Fig 5a lists details about the datasets, the methods, parameters and benchmark calculations. The same ISSO Size Filter procedure is applied to both datasets with different parameters. Fig 5b gives the benchmark results using the proposed method compared to the published results from CellProfiler as well as to recent results from Gradient Flow Tracking. In both examples, the proposed method outperforms the published results using the CellProfiler, improving the benchmarks from 17% and 6.7% to 3.99% and 0.1% respectively. While Gradient Flow Tracking method does perform almost as well as the ISSO transform for the human dataset, it performs poorly for the fly dataset. Fig 6 gives segmentation results for randomly selected images from both datasets using different methods. More specifically, Fig 6a, 6b and 6c are the input images from the human sample and two fly samples respectively; Fig 6d through 6f are the corresponding results using CellProfiler, Fig 6g—6i are the results from the Gradient Flow Tracking method and Fig 6j—6l are the results using the proposed method. It is evident that ISSO Size Filter consistently gives results that are significantly less over or under segmented.

Parallel Implementation of the Size Filter

The ISSO Size Filters defined in (6) and (7) are immediately suitable for parallel implementation. In contrast to the marker-controlled watershed algorithm, the self-organizing function in (6) does not depend on the first input B_1 , or X_h in (4). Hence, we can first calculate X_h independently and in parallel at each level h . Then all results at all levels are summarized based on AGR functions defined in (6) or (7). As a result, the recursive algorithm (4) can be reorganized as illustrated by the flowchart shown in Fig 7, which fits very well to the popular Map/Reduce parallel algorithm scenario (Dean & Ghemawat, 2004) by treating the SO function as a Map step and the AGR function as a Reduce step.

Algorithm Sensitivity to Parameters: Comments on example of cell nuclei

The procedure introduced above for the algorithm evaluation depends mainly on three pieces of prior information for a successful segmentation, highlighted here for examples of segmenting cell nuclei:

1. the maximum size of the cell nuclei;
2. the minimum disk structure element radius chosen to smooth the cell nuclei borders;
3. the minimum highest intensity value that an object has to reach to be considered as a cell nuclei.

The first parameter (the maximum size of the cell nuclei) is usually determined by visually identifying and estimating the largest cell nuclei in the image, and is generally not very sensitive to the final segmentation/counting result. The choice of the second parameter (the minimum disk structure element) can be critical to the final segmentation result, thus generally serves as a tuning parameter. The third parameter is introduced mainly to filter out the objects that are too dim to be considered as cell nuclei and usually can be chosen with ease. Fig 8 gives plots of the benchmark error rates for the human and fly datasets as a function of minimum structure element radius when the other two parameters are fixed. The horizontal lines give the published benchmarks. We can conclude that: a). Inappropriate choice of this parameter can cause over segmentation when it is too small or under segmentation when it is too large; and b). there is a wide range of parameter values in each case that the proposed algorithm outperforms the published results, suggesting relative parameter insensitivity of our algorithm.

Conclusions

The novel Immersion Simulation based Self-Organizing (ISSO) transform involves three key components: 1) define or use a default self-organizing function to perform binary transformation; 2) define or use a default aggregation function to compose the binary output images from 1) into a transformed grayscale image; and 3) perform immersion simulation on the threshold set of original image using self-organizing function from 1) and aggregation function from 2) to transform the image into a new grayscale image. The latter step defines an image that usually has desirable features such as better contrast or more homogenous intensity distribution that can be further segmented using existing method such as global/local thresholding, marker-controlled watershed or Level Set approaches. By contrast, the marker-controlled watershed transform uses only predefined initial markers as starting point, and hardwires the influence zone function as a self-organizing function during the immersion simulation; this is thus a special case of ISSO.

Defining an ISSO Size Filter as a special case of ISSO transforms, we demonstrate this new approach through a variety of segmentation examples and show that the new algorithm can be used to effectively segment both irregular-shaped binary particle images and grayscale images. Using a modified Size Filter that incorporates shape-related information into the self-organizing function, we successfully segmented cell nuclei in the microscopic images from two benchmark image sets of human and fly. We achieve much better benchmark error rates than currently published results, and give examples for customizing the ISSO transform for different image processing problems such as noise removal, image smoothing etc.

Compared to the watershed algorithm, a set of parameters allow for fine tuning of the ISSO algorithm and provide opportunities for easily customizing the transform through user defined self-organizing functions and aggregation functions. We also note that the proposed Size Filter transform is intrinsic parallel and can be readily implemented in a parallel computing environment. Furthermore, our numerical experiments suggest that the proposed algorithm is relatively insensitive to the choice of parameters compared to

other leading algorithms, and experiments also suggest that a good choice of parameters can indeed make the segmentation more accurate.

Our future work includes developing automated method for parameter selection, developing more self-organizing and aggregation functions to be suitable and applicable to other segmentation problems, and integrating the method into the software packages CellProfiler and CellTracer. The current implementation of the method and examples can be downloaded from the CellTracer website <http://www.stat.duke.edu/research/software/west/celltracer/isso>.

Acknowledgements

I am grateful to Duke colleagues Jarad Niemi, Chee-Meng Tan, Lingchong You, Joe Nevins and Mike West for useful and formative discussions, and to H.Q. Sun for providing the binary particle image. The research is supported, in part, by the National Science Foundation (grants DMS-0342172 and BES-0625213) and the National Institutes of Health (grants CA-112925 and P50-GM081883-01).

References

- Abramoff, M.D., Magelhaes, P.J. & Ram, S.J. (2004) Image Processing with ImageJ. *Biophotonics International* 11(7), 36-42.
- Berlemont, S., Bensimon, A. & Olivo-Marim, J.C. (2007) Detection of DNA Filaments in Fluorescent Microscopy Using Feature-adapted Veamlet Transforms. *MICCAI International Workshop on Microscopic Image Analysis with Applications in Biology*.
- Beuchar, S. (1992) The watershed transformation applied to image segmentation, presented at 10th Pfefferkorn Conf. on Signal and Image Processing in Microscopy and Microanalysis.
- Beuchar, S. (1994) Watershed, hierarchical segmentation and waterfall algorithm. In J. Sera and P. Soille, editors, *Mathematical Morphology and its applications on image and signal processing*. 69-76. Kluwer Academic Press.
- Carpenter, A.E., Jones, T.R., Lamprecht, M.R., Clarke, C. Kang, I.H., Friman, et al; (2006) CellProfiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biology* 7:R100.
- Caselles, V., Kimmel, R. & Sapiro, G. (1997) Geodesic active contours. *Int. J. Comp. Vis.* 22, 61-79.
- Chan, T. & Vese, L. (2001) Active contours without edges. *IEEE Trans. On Image Processing* 2, 266-277.

- Chen, X., Zhou, X. & Wong, S.T. (2006) Automated segmentation, classification, and tracking of cancer nuclei in time-lapse microscopy. *IEEE Trans. Biomed. Eng.* 53, 762-766.
- Cristoforetti, A., Faes, L. & Ravelli F., et al. (2008) Isolation of the left atrial surface from cardiac multi-detector CT images based on marker-controlled watershed segmentation. *Med. Eng. Phis* 30, 48-58.
- Dean, J. & Ghemawat, S. Simplified Data Processing on Large Clusters. *OSDI'04: Sixth Symposium on Operating System Design and Implementation.*
- Elowitz, M.B., Levine, A.J., Siggia, E.D. & Swain, P.S. (2002) Stochastic gene expression in a single cell. *Science* 297, 1183-1186.
- Gordon, A. Colman-Lerner, A., Chin, T.E., Benjamin, K.R., Yu, R.C. & Brent Roger (2007) Single-cell quantification of molecules and rates using open-source microscope-based cytometry. *Natures Methods* 4, 175-181.
- Jalba, A.C., Wilkinson, M.H.F. & Roerdink, J.B.T.M (2004) Automatic segmentation of diatom images for classification. *Microsc. Res. Tech.* 65, 72-85.
- Jung, C.R. (2007) Unsupervised multiscale segmentation of color images. *Pattern Recognit. Lett.* 28, 523-533.
- Kass, M., Witkin, A. & Terzopoulos (1988), Snakes, active contour model. *International Journal of Computer Vision*, 321-331.
- Li, G., Liu, T., Nie, J., Guo, L., Chen, J., Zhu, J., Xia, W., Mara, A., Holley, S. & Wong, S.T.C. (2008) Segmentation of touching cell nuclei using gradient flow tracking. *Journal of Microscopy* 231, 47-58.
- Lin, G., Chawla, M.K., Olson, K., Guzowski, J.E., Barnes, C.A. & Roysam, B. (2005) Hierarchical, model-based merging of multiple fragments for improved three-dimensional segmentation of nuclei. *Cytometry* 64A, 20-33.
- Mao, K.Z., Zhao, P. & Tan, P.H. (2006) Supervised learning-based cell image segmentation for P53 immunohistochemistry. *IEEE Trans. Viomed. Eng.* 53, 1153-1163.
- Megason, S.G. & Fraser, S. E. (2007) Imaging in systems biology. *Cell* 130, 784-795.
- Otsu, N. (1979) A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man. Cybern.*, 9, 62066.
- Meyer, F. (2004) Topographical distance and watershed lines. *Signal Processing* 38(1), 113-125.

Najman, L. & Malik, J. (1996) Geodesic Saliency of Watershed Contours and Hierarchical Segmentation. *IEEE Trans. On Pattern Analysis and Machine Intelligence* 18(12), 1163-1173.

Sezgin, M. & Sankur, B. (2004) Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging* 13(1), 146-165.

Osher, S. Sethian, J. (1988) Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-jacobi formulations. *Journal of Computational Physics*, 12-49.

Otsu, N. (1979) A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man. Cybern.* 9, 62-66.

Roerdink, J.B.M.T. & Meijster, A. (2000) The Watershed Transform: Definitions, Algorithms, and Parallelization Strategies. *Fundamenta Informaticae* 41, 187-228.

Rosenfield, N., Young, J.W., Alon, U., Swain, P.S. & Elowitz, M.B. (2005) Gene regulation at the single-cell level. *Science* 307, 1962-1965.

Sezgin, M. & Sankur, B. (2004) Survey over image thresholding techniques and quantitative performance evaluation. *J. Electronic Imaging*. 13(1), 146-16.

Sun, H.Q & Luo, Y.J. (2009) Adaptive watershed segmentation of binary particle image. *Journal of Microscopy*.

Umesh Adigam, P.S. & Chaudhuri, B.B. (2001) An efficient method based on watershed and rule-based merging for segmentation of 3-D histopathological images. *Pattern Recogn.* 34, 1449-1458.

Vincent, L. & Soille, P. (1991) Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Trans. Pattern Anal. Mach. Intell.* 13, 583-598.

Wang, M., Zhou, X., Li, F., Huckins, J., King, R.W. & Wong, S.T.C. (2008) Novel cell segmentation and online SVM for cell cycle phase identification in automated microscopy. *Bioinformatics* 24:1, 94-101.

Wang, Q., Neimi, J., Tan, C., You, L. & West, M. (2009) Image Segmentation and dynamic lineage analysis in single-cell fluorescence microscopy. *Journal of Synth. Biol.* (in the press).

Wu, H.S., Barba, J. & Gill, J. (2000) Iterative thresholding for segmentation of cell images. *J. Microsc.* 197, 296-304.

Wu, K., Gauthier, D. & Levine, M.D. (1995) Live cell image segmentation. *IEEE Trans. Biomed. Eng.* 42, 1-12.

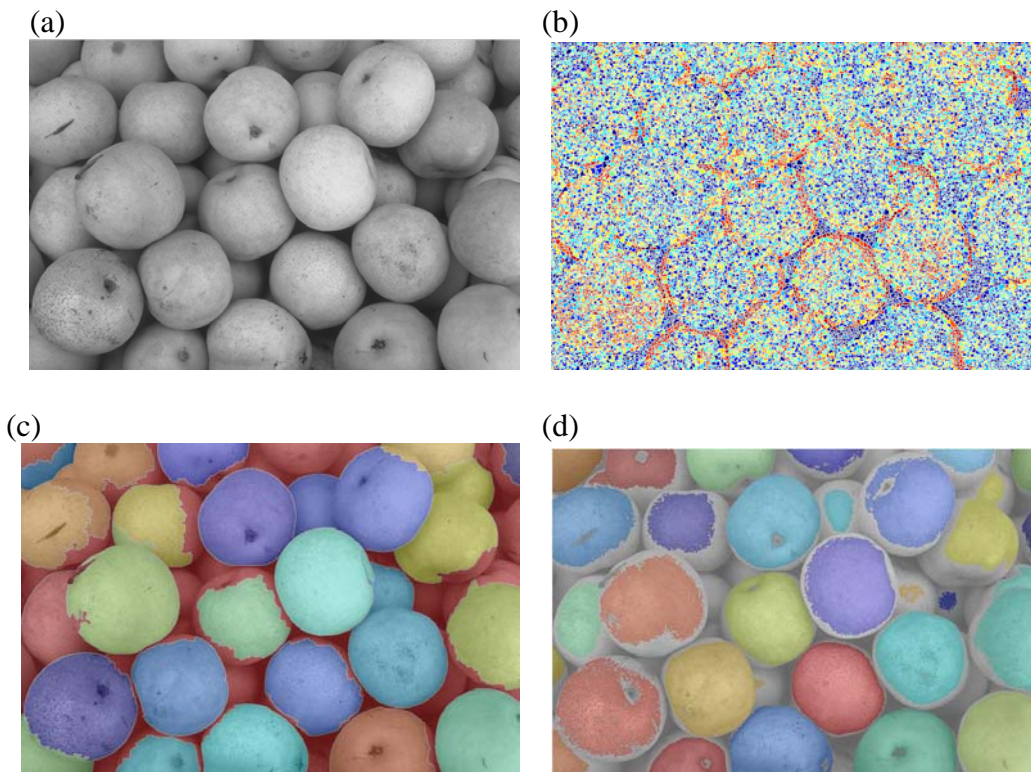


Figure 1: (a) the original image of pears; (b) result of applying watershed transform to the gradient magnitude of (a); (c) result of applying marker-controlled watershed transform to the gradient magnitude of (a), with the foreground markers generated by sequentially applying morphological opening by reconstruction, closing by reconstruction and removing small regional maxima, and with the background generated by applying watershed transform to the distance transform of the Otsu thresholding of the original image. (d). result of applying ISSO Size Filter to the original image.

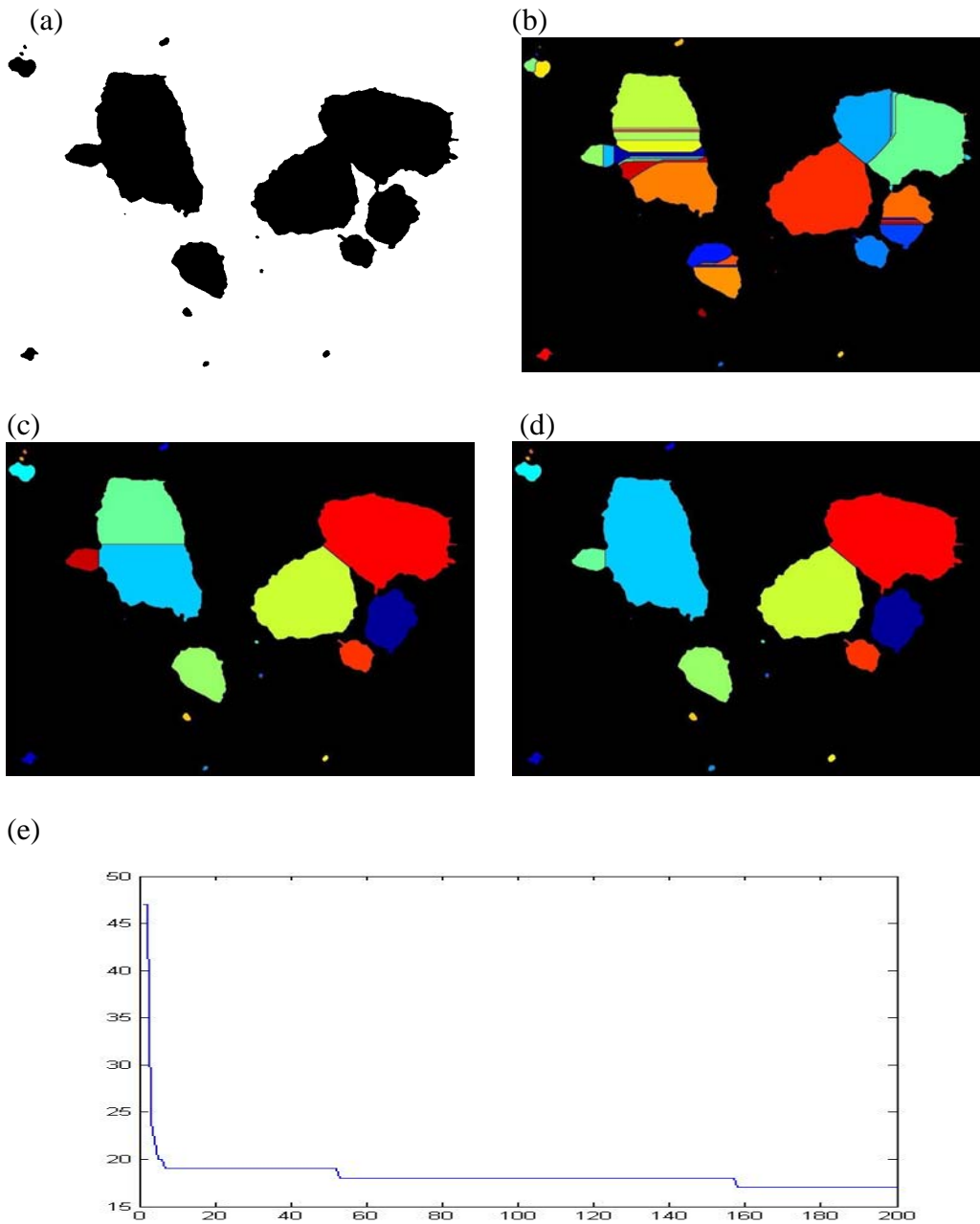


Figure 2: (a) the input irregular-shaped binary particle image; (b) result using watershed segmentation; (c) result using ISSO-Size Filter with parameter $Size_{min} = 10$; (d). result using ISSO-Size Filter with parameter $Size_{min} = 100$; (e) number of particles (Y-axis) vs. parameter $Size_{min}$ (X-axis).

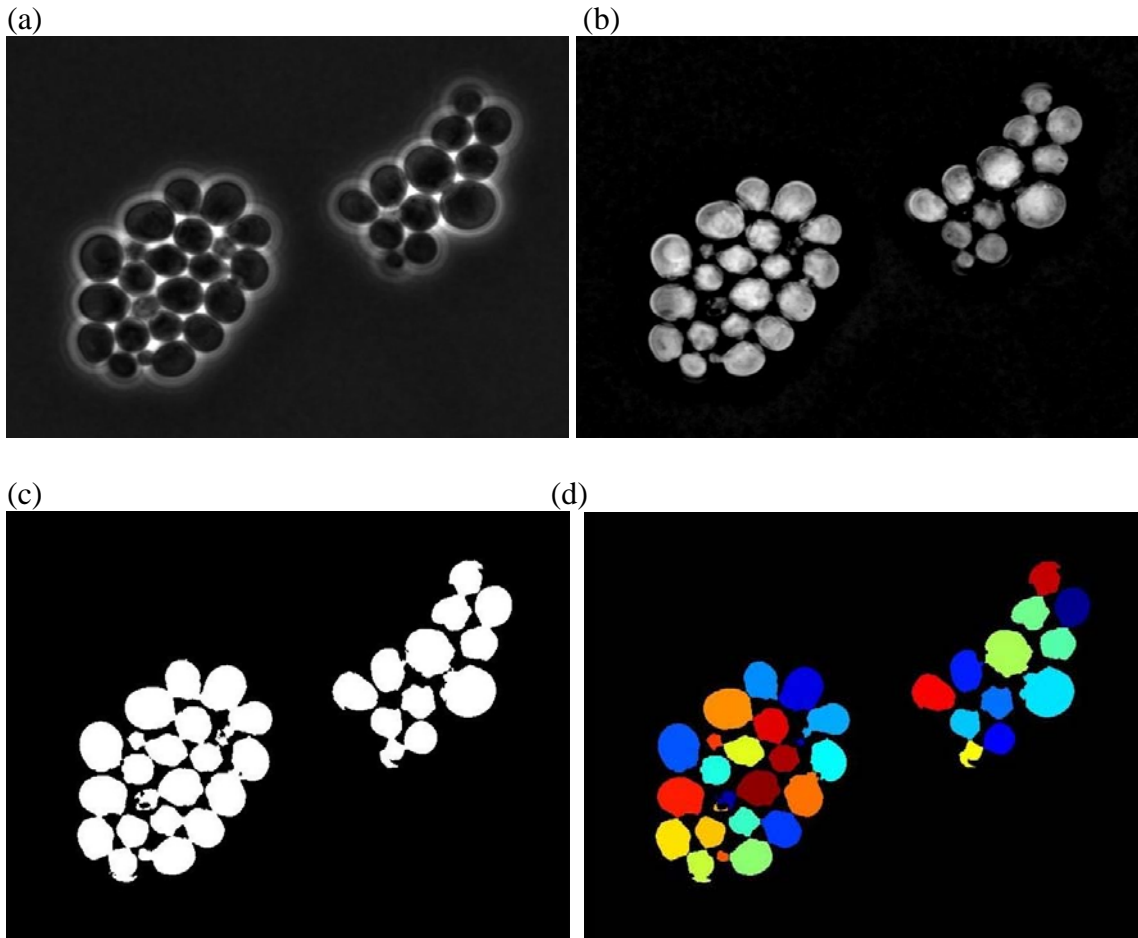
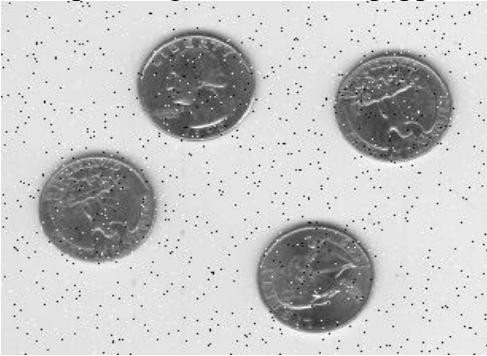
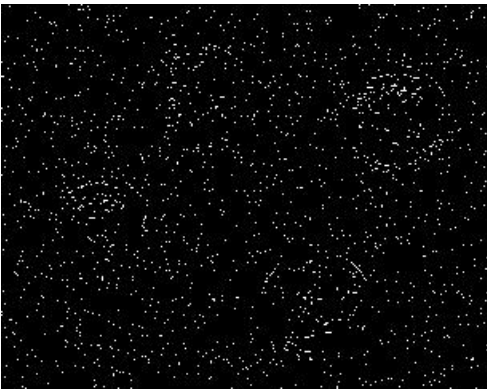


Fig 3: Segmentation of budding yeast cells using ISSO Size Filer. a) Input budding yeast cell image; b) result from ISSO Size Filter; c) Extracted foreground after global thresholding; d) final segmentation result.

(a) Input image with "salt & pepper" noise (b) Result with median filter



(c) Identified noise using Size Filter (d) Filtered image with less blur



(e) Smoothed pear image using modified Size Filter



Fig 4: (a) an image contaminated with "salt & pepper" noise; (b) result using a median filter to remove "salt & pepper" noise; (c) identified noise-contaminated pixels using Size Filter; (d) final result of applying median filter only to pixels identified as noise; (5) result using modified Size Filter to smooth Fig 1a, with most blemishes being removed.

(a)

		Human HT29 Colon Cancer 1	Drosophila KC 167 1(fly)
		6	5
Images per sample		1	10
Total number of samples		6	50
Method		Step1: Apply the modified Size Filter to transform input images; Step 2: Use Rank Transform on hybrid images with default parameters threshold images; Step 3. Use the procedure described in Example 2 for final binary Segmentation.	
Parameters:	Structure element radius	2	5
	Maximum cell size	5000	
	Minimum Highest Intensity value	0 (not used)	20
Benchmark calculations:		First compute for each image the absolute difference between the algorithm's count and the average of the humans' counts, then divide by the latter to obtain the deviation from ground truth (in percent). The mean of these values over all 6 images is the final result.	First compute for each sample the algorithm's mean cell count over the 10 images of the sample. Next, calculate the absolute difference between this mean and the average of the humans' counts for the sample, then divide by the latter to obtain the deviation from ground truth (in percent). The mean of these values over all 5 samples is the final result.

(b)

Method	Human	Fly
CellProfiler	6.7%	17%
Gradient Flow Tracking	1.35%	82%
ISSO Transform	1.13%	3.9%

Figure 5: (a) Basic information used for algorithm evaluation; (b) Benchmarks comparison of different algorithms.

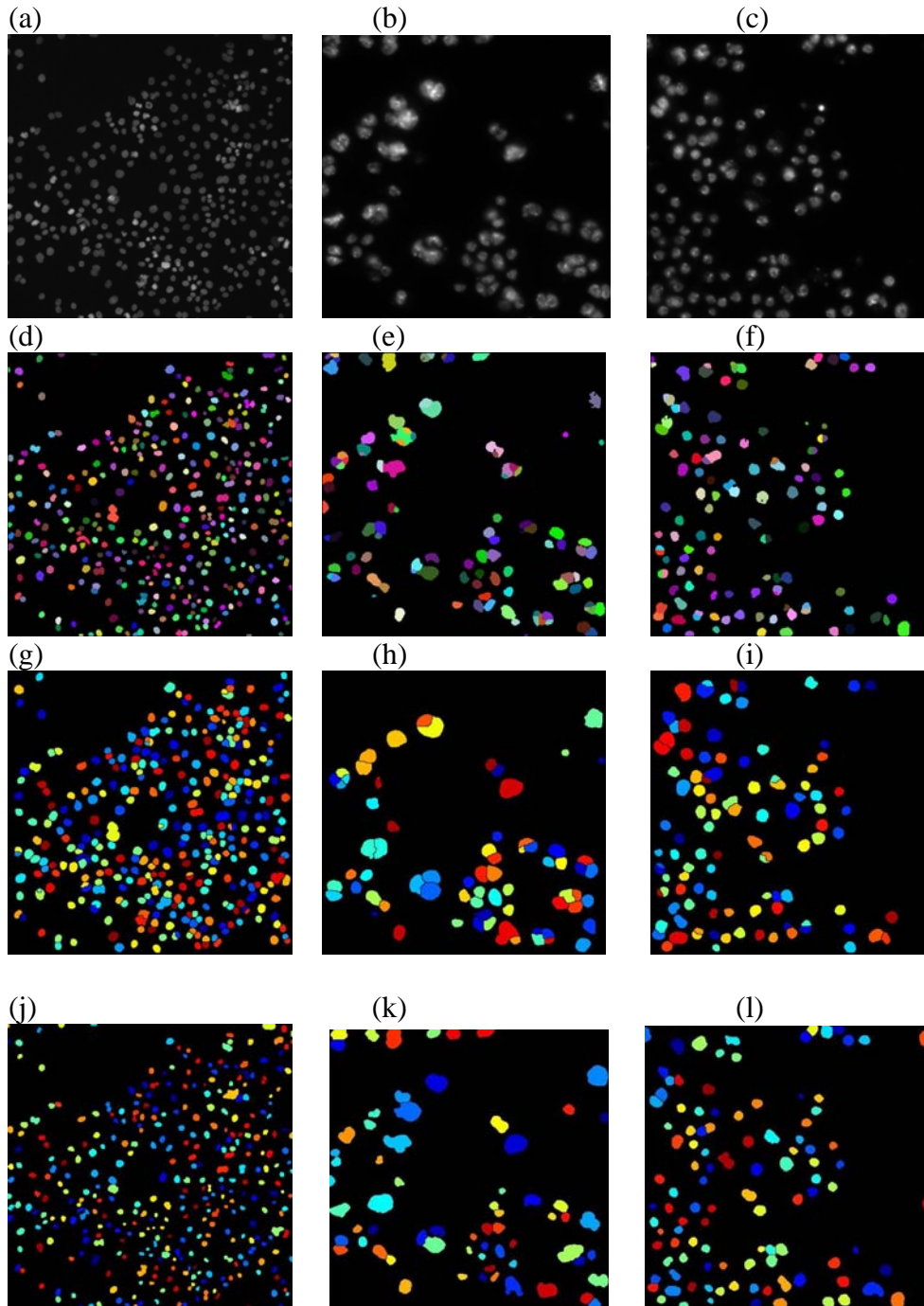


Figure 6: Segmentation results for selected human and fly data. (a)--(c) input images; (d)--(f) results from Gradient Flow Tracking method; (g)--(i) results from CellProfiler; (j)--(l) results using proposed method.

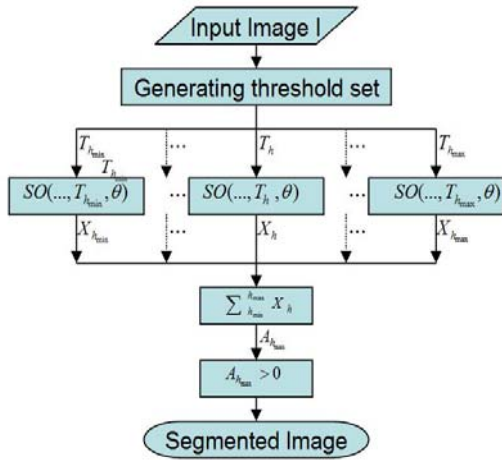
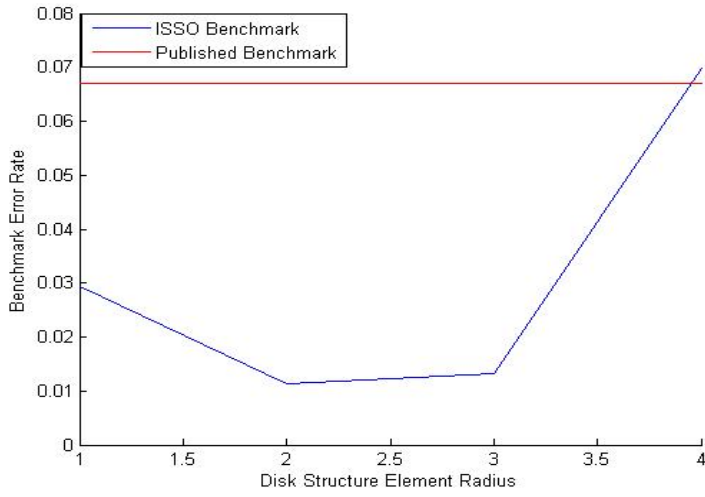


Fig 7: Flowchart for ISSO Size Filter

(a)



(b)

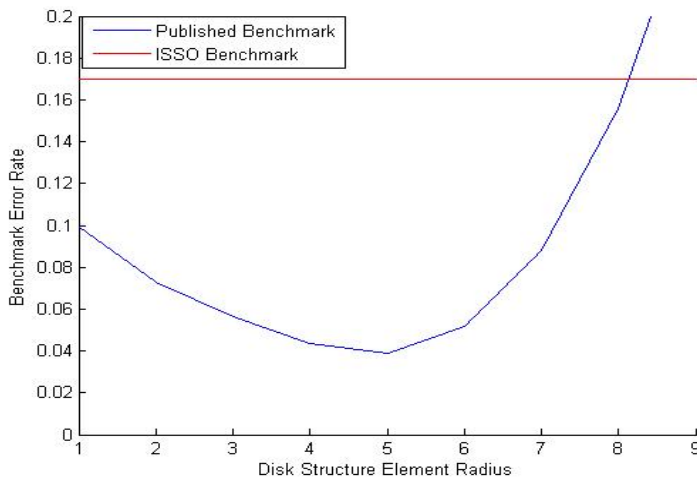


Fig 8: Benchmarks vs. tuning parameter "disk structure element radius" for human and fly datasets. a) human dataset; b) fly dataset. Blue lines are benchmarks from proposed algorithm and red lines are the published results from CellProfiler.