Attributed Relational Graphs for Cell Nucleus Segmentation in Fluorescence Microscopy Images

Salim Arslan, *Student Member, IEEE*, Tulin Ersahin, Rengul Cetin-Atalay, and Cigdem Gunduz-Demir*, *Member, IEEE*

***Abstract*—More rapid and accurate high-throughput screening in molecular cellular biology research has become possible with the development of automated microscopy imaging, for which cell nucleus segmentation commonly constitutes the core step. Although several promising methods exist for segmenting the nuclei of monolayer isolated and less-confluent cells, it still remains an open problem to segment the nuclei of more-confluent cells, which tend to grow in overlayers. To address this problem, we propose a new model-based nucleus segmentation algorithm. This algorithm models how a human locates a nucleus by identifying the nucleus boundaries and piecing them together. In this algorithm, we define four types of primitives to represent nucleus boundaries at different orientations and construct an attributed relational graph on the primitives to represent their spatial relations. Then, we reduce the nucleus identification problem to finding predefined structural patterns in the constructed graph and also use the primitives in region growing to delineate the nucleus borders. Working with fluorescence microscopy images, our experiments demonstrate that the proposed algorithm identifies nuclei better than previous nucleus segmentation algorithms.**

***Index Terms*—Attributed relational graph, fluorescence microscopy imaging, graph, model-based segmentation, nucleus segmentation.**

I. Introduction

AUTOMATED fluorescence microscopy imaging systems are becoming important tools for molecular cellular biology research because they enable rapid high-throughput screening with better reproducibility. The first step of these systems typically includes cell/nucleus segmentation, which greatly affects the success of the other system steps. These systems can be used in different types of biological applications for cells showing different characteristics. Many types of cells grow as monolayer isolated cells, whereas some grow in overlayers on top of each other. These overlayered cells (also called cell clumps) take up more space, which makes them more confluent. A high occlusion level in cell clumps decreases contrast among the nuclei, which makes their boundaries more difficult to perceive. This, in turn, may result in identifying multiple nuclei as a single cluster or identifying a single nucleus as fragments. Moreover, nonideal experimental conditions, such as weak staining and poor background illumination, may lead to nucleus misidentification. Thus, it is of great importance to develop segmentation algorithms that are robust to nonideal conditions and can operate on the nuclei of isolated and overlayered cells.

Several studies on cell nucleus segmentation exist in the literature. When images comprise monolayer isolated or less-overlayered cells, relatively simple methods such as thresholding [1] and clustering [2] can be used. These methods, however, are typically inadequate for segmenting the nuclei of more-overlayered cells. In such cases, the most commonly used methods include marker-controlled watersheds and model-based segmentation algorithms. The former usually define a set of markers through pixel intensities/gradients [3] and/or distance transforms [4], [5], and let water rise only from these markers. These methods usually apply a merging process [6] on their results to overcome the over-segmentation problem, which is typically observed in watersheds. Model-based segmentation uses *a priori* information on nucleus properties to decompose overlayered nuclei. Examples include using roundness and convexity properties of a nucleus [7], [8] and a symmetry property of its boundaries [9]. Although all these methods lead to promising results, challenges still remain in segmenting the nuclei of overlayered cells because of the nature of the problem. Nucleus segmentation, like all other segmentation problems, heavily depends on the segmenter's abilities to differentiate between noise and nuclei, discern image variations, and decompose overlayered nuclei.

In this paper, we introduce a new model-based nucleus segmentation algorithm, which relies on a trivial fact that each nucleus has a left boundary, a right boundary, a top boundary, and a bottom boundary, and these boundaries must be in the correct positions in relation to each other. Fig. 1 illustrates these boundaries for four nuclei with the left, right, top, and bottom boundaries shown as green, yellow, pink, and red, respectively. In the figure, one can observe that the bottom (red) boundary of Nucleus 1 is not identified due to an uneven lighting condition. Similarly, the right (yellow) boundary of Nucleus 2 is not present in the image due to its partial overlapping with Nucleus 3. However, it is possible to identify these two nuclei by using only the present boundaries and their spatial relations. Moreover, here it is obvious that the green boundary of Nucleus 3

Manuscript received January 17, 2013; revised March 12, 2013; accepted March 20, 2013. Date of publication March 28, 2013; date of current version May 29, 2013. This work was funded by the Scientific and Technological Research Council of Turkey under Project TÜBİTAK109E206. *Asterisk indicates corresponding author.*

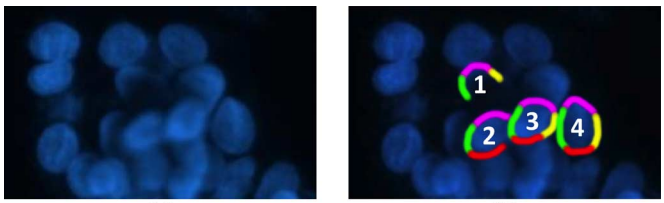
S. Arslan is with the Department of Computer Engineering, Bilkent University, TR-06800 Ankara, Turkey (e-mail: salima@cs.bilkent.edu.tr).

T. Ersahin and R. Cetin-Atalay are with the Department of Molecular Biology and Genetics, Bilkent University, TR-06800 Ankara, Turkey (e-mail: ersahin@bilkent.edu.tr; rengul@bilkent.edu.tr).

*C. Gunduz-Demir is with the Department of Computer Engineering, Bilkent University, TR-06800 Ankara, Turkey (e-mail: gunduz@cs.bilkent.edu.tr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMI.2013.2255309



(a) (b)

Fig. 1. (a) An image of HepG2 hepatocellular carcinoma cell nuclei. (b) For four individual nuclei, the left, right, top, and bottom boundaries are shown as green, yellow, pink, and red, respectively.

cannot belong to Nucleus 2, and we can see that Nucleus 3 overlaps with Nucleus 2. On the other hand, the boundaries of Nucleus 3 and Nucleus 4 form closed shapes, making them easy to separate from the background. In this work, such observations are our main motivations behind implementing the proposed algorithm.

Our contributions are threefold. First, we represent nucleus boundaries at different orientations (left, right, top, and bottom) by defining four types of primitives and construct an attributed relational graph on these primitives to represent their spatial relations (we call our algorithm *ARGraphs*). Second, we reduce the nucleus identification problem to locating predefined structural patterns on the constructed graph. Third, we employ the boundary primitives in region growing to delineate the nucleus borders. The proposed algorithm mainly differs from previous nucleus segmentation algorithms in the following aspect: Instead of directly working on image pixels, our algorithm works on high-level boundary primitives that better correlate with the image semantics. Using boundary primitives better decomposes overlayered nuclei and this method is generally less vulnerable to the noise and variations typically observed at the pixel level. The proposed algorithm differs from previous model-based segmentation methods by attributing boundary primitives to a type and locating nuclei via searching for structural patterns on a related graph constructed on the attributed primitives. Working on 2661 nuclei, our experiments demonstrate that the proposed algorithm improves the segmentation performance of fluorescence microscopy images compared to its counterparts.

II. Related Work

Many studies have been proposed for nucleus segmentation. Simple methods are usually sufficient to segment the nuclei of monolayer isolated cells; for example, one can obtain a binary map by thresholding [1], clustering [2], [10], or classification [11] and find connected components of this map to locate the nuclei. It is possible to refine boundaries using active contour models, which converge to final boundaries by minimizing the energy functions usually defined on intensities/gradients and contour curvatures [12]–[14].

To segment the nuclei of overlayered cells, however, it is necessary to decompose clusters into separate nuclei. There are two classes of methods commonly used for this purpose: marker-controlled watersheds and model-based segmentation algorithms. The former use predefined markers (which will correspond to nucleus locations) as regional minima and start a flooding process only from these markers. With this method, it is crucial to correctly determine the markers, and one way of doing that is to select local minima on pixel intensities/gradients [3], [6] and/or distance transforms [5], [15]. Another way is to iteratively apply morphological erosion on nuclear regions of the binary map [16], [17]. Because these methods usually yield more markers than the actual cells, it is common to use the h-minima transform to suppress undesired minima [4], [18]. Many studies postprocess the segmented nuclei obtained by these watersheds because they often yield over-segmentation. This postprocessing is based on features extracted from the segmented nuclei and boundaries of adjacent ones. To merge the over-segmented nuclei, the extracted features are subjected to rule-based techniques [19], [20], recursive algorithms [6], [21], and statistical models learned from training samples [22], [23].

Model-based segmentation algorithms decompose overlayered nucleus clusters into separate nuclei by constructing a model on *a priori* information about nucleus properties. A large set of these algorithms uses a nucleus' convexity property. Thus, they locate concave points, which correspond to places where two nuclei meet, on cluster boundaries and decompose the clusters from these points. They typically use curvature analysis [24], [25] to find the concave points but points can also be found by identifying pixels farthest from the convex hull of the clusters [8]. The studies use different methods, such as Delaunay triangulation [25], ellipse-fitting [26], and path-following [27], to decompose the cluster from the concave points. Another set of models uses a radial-symmetry property of nucleus boundaries. They locate nucleus centers by having pixels iteratively vote at a given radius and orientation specified by the predefined kernels [9], [28], [29]. One can also use radial-symmetry points in a rule-based method to find the splitting points [30]. A smaller set of models exists that uses the roundness property of a nucleus. These locate circles/ellipses on nuclear regions of the binary map to find initial boundaries and refine them afterwards [7], [31].

There are other classes of methods for nucleus segmentation. One uses filters that have been defined considering nucleus' roundness and convexity properties. Using the responses obtained from these filters, pixels with high responses are selected as initial nucleus locations. These locations are further processed to determine the nucleus borders [32], [33]. Another class uses density functions to locate nuclei on images. Studies use supervised [34] and unsupervised methods [35] to learn these functions.

III. Methodology

The proposed algorithm relies on modeling cell nucleus boundaries for segmentation. We approximately represent the boundaries by defining high-level primitives and use them in two main steps: 1) nucleus identification and 2) region-growing. In the first step, we construct a graph on the primitives according to their types and adjacency. Then, we use an iterative search algorithm that locates predefined structural patterns on the graph and identifies the located structures as nuclei if they satisfy the shape constraints. The region-growing step employs the primitives to find the nucleus borders and determines the

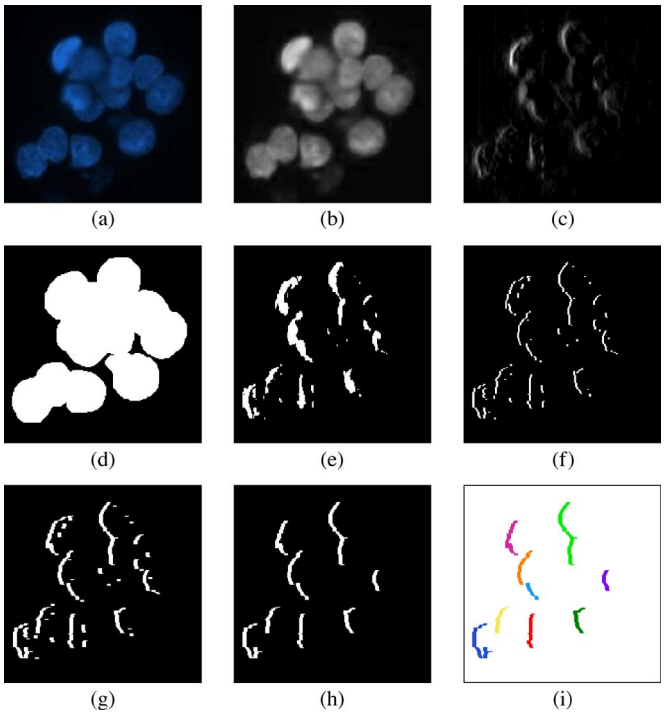


Fig. 2. Defining left boundary primitives: (a) original subimage, (b) blue band I_b of the image, (c) response map $R_{\texttt{left}}$ obtained by applying the Sobel operator $S_{\texttt{left}}$, (d) mask used to determine local Sobel threshold levels, (e) binary image $B_{\texttt{left}}$ after thresholding, (f) boundaries obtained after taking the leftmost pixels [here there are discontinuities between the boundaries because of their one-pixel thickness], (g) boundary map $P_{\texttt{left}}$ obtained after taking the d-leftmost pixels, (h) $P_{\texttt{left}}$ after eliminating small connected components, (i) left boundary primitives, each of which is shown in a different color.

growing direction and stopping point based on the primitive locations.

A. Primitive Definition

In the proposed method, we define four primitive types that correspond to the left, right, top, and bottom nucleus boundaries. These boundary primitives are derived from the gradient magnitudes of the blue band I_b of an image. To this end, we convolve the blue band I_b with each of the following Sobel operators, which are defined in four different orientations, and obtain four maps of the responses. Then, we process each of these responses, as explained below and illustrated in Fig. 2, to define the corresponding primitives

$$S_{\texttt{left}} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad S_{\texttt{right}} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$S_{\texttt{top}} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad S_{\texttt{bottom}} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

Let $R_{\texttt{left}}$ be the response map obtained by applying the Sobel operator $S_{\texttt{left}}$ to the blue band image I_b. We first threshold $R_{\texttt{left}}$ to obtain a binary left boundary map $B_{\texttt{left}}$. Here, we use local threshold levels instead of a global level because illuminations and gradients are commonly uneven throughout our images. To do this, we employ a mask that roughly segments nuclear regions from the background. For each connected component $C^{(k)}$ of this mask, we calculate a local threshold level $T^{(k)}_{\texttt{left}}$ on the gradients of its pixels by using the Otsu method. Then, the pixels of this component are identified as boundary if their responses are greater than the calculated local threshold.

Next, we fill the holes in $B_{\texttt{left}}$ and take its d-leftmost pixels. The map $P_{\texttt{left}}$ of the d-leftmost pixels is defined as

$$P_{\texttt{left}}(i,j) = \begin{cases} 1, & \text{if } B_{\texttt{left}}(i,j) = 1 \text{ and } \exists x \in \mathbb{Z}^+ \text{ s.t.} \\ & x \leq d \text{ and } B_{\texttt{left}}(i-x,j) = 0 \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

In this definition, the d-leftmost pixels are taken instead of just the leftmost pixels because, as illustrated in Fig. 2(f), the leftmost pixels do not always contain all the nucleus boundaries, and thus there may exist discontinuities between boundaries of the same nucleus. By taking the d-leftmost pixels, the discontinuities are more likely to be eliminated, as shown in Fig. 2(g). Finally, we eliminate the connected components of $P_{\texttt{left}}$ whose heights are less than a threshold t_{size} and identify the remaining ones as left boundary primitives [see Fig. 2(i)].

Likewise, we define the right boundary primitives $P_{\texttt{right}}$, top boundary primitives $P_{\texttt{top}}$, and bottom boundary primitives $P_{\texttt{bottom}}$. In each of these definitions, (1) is modified so that it gives the d-rightmost, d-topmost, and d-bottommost pixels. In eliminating small primitives, components whose heights are lower than the threshold t_{size} are eliminated for $P_{\texttt{right}}$, whereas those whose widths are less than t_{size} are eliminated for $P_{\texttt{top}}$ and $P_{\texttt{bottom}}$. Note that local threshold levels are separately calculated for each primitive type.

In this step, we use a mask to calculate local threshold levels. This mask roughly identifies nuclear regions but does not provide their exact locations. Our framework allows using different binarization methods such as adaptive thresholding [15] and active contours without edges [36]; however, because the mask is used just for calculating local thresholds, we prefer a simpler method. In our binarization method, we first suppress local maxima of the blue band image I_b by subtracting its morphologically opened image from itself; here we use a gray-scale opening operation. This process removes noise from the image without losing local intensity information. Then, we calculate a global threshold level on the suppressed image using the Otsu method.[1] Finally, we eliminate small holes and regions from the mask.

B. Nucleus Identification

Nuclei are identified by constructing a graph on the primitives and then applying an iterative algorithm that searches this

[1]We use the half of the Otsu threshold to ensure that almost all nuclear regions are covered by the mask. This is important because the primitives can only be defined on connected components of this mask. The original Otsu threshold would lead to smaller connected components that might not cover some primitives. Note that when the image is not clean, the mask covers a larger area (more pixels). However, this rarely introduces unwanted primitives because the pixels in this mask are not directly used to define the primitives; their gradients are used to calculate the local thresholds. Pixels form a primitive if their gradients are greater than their local thresholds and if they form a long enough structure. Our experiments confirm this observation: A larger number of gradients (pixels) only slightly changes the local thresholds and this change does not produce too many unwanted primitives.

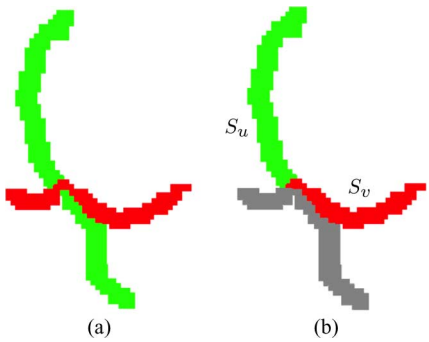


Fig. 3. Assigning an edge between a left and a bottom primitive: (a) primitives and (b) selected primitive segments.

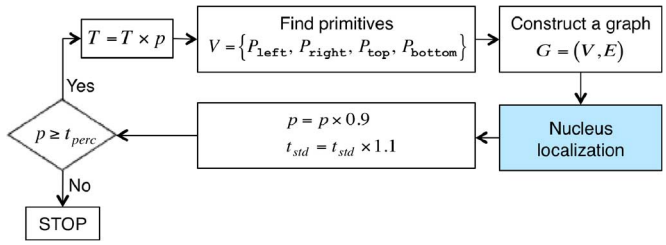


Fig. 4. Flowchart of the iterative search algorithm.

graph to locate structural patterns conforming to the predefined constraints. The details are given in the next subsections.

1) Graph Construction: Let $G = (V, E)$ be a graph constructed on the primitives $V = \{P_{\texttt{left}}, P_{\texttt{right}}, P_{\texttt{top}}, P_{\texttt{bottom}}\}$ that are attributed to their primitive types. An edge $e = (u, v) \in E$ is assigned between primitives u and v if the following three conditions are satisfied.

1) The primitives have overlapping or adjacent pixels.
2) One primitive is of the vertical (left or right) type and the other is of the horizontal (top or bottom) type.
3) Each primitive has a large enough segment on the correct side of the other primitive. For left and right primitives, the width of this segment must be greater than the threshold t_{size} (which was also used to eliminate small components in the previous step). Likewise, for top and bottom primitives, the height of the segment must be greater than t_{size}.

Fig. 3 illustrates the third condition: Suppose we want to decide whether or not to assign an edge between left primitive u and bottom primitive v, which are shown in green and red in Fig. 3(a), respectively. To do so, we first select the segment of each primitive that lies on the correct side of the other. It is obvious that the left boundaries of a given nucleus should be on the upper left-hand side of its bottom boundaries, and likewise, its bottom boundaries should be on the bottom right-hand side of its left boundaries. To reflect this fact, we select segment S_u of primitive u (which corresponds to the left boundaries), found on the upper left-hand side of v (which corresponds to the bottom boundaries). Similarly, we select the segment S_v of v, which is found on the bottom right-hand side of u. Fig. 3(b) shows the selected segments in green and red; nonselected parts are shown in gray. Finally, we assign an edge between u and v if the height of S_u and the width of S_v are greater than threshold t_{size}.

2) Iterative Search Algorithm: Each iteration starts with finding boundary primitives, as explained in the primitive definition step in Section III-A. In that step, the local thresholds $\mathcal{T}^{(k)} = \langle T^{(k)}_{\texttt{left}}, T^{(k)}_{\texttt{right}}, T^{(k)}_{\texttt{top}}, T^{(k)}_{\texttt{bottom}} \rangle$ are separately calculated for each connected component $C^{(k)}$ of the binary mask, and pixels with Sobel responses greater than the corresponding thresholds are identified as boundary pixels. These pixels are then processed to obtain the primitives.

Our experiments reveal that primitives identified using the vectors $\mathcal{T}^{(k)}$ do not always cover all nucleus boundaries. This situation is attributed to the fact that illumination and gradients are not even throughout an image. For instance, boundary gradients of nuclei closer to an image's background are typically higher than those located towards a component center. When the thresholds are decreased to cover lower boundary gradients, false primitives may be found especially in nuclei with higher boundary gradients. Thus, to consider lower boundary gradients while avoiding false primitives, we apply an iterative algorithm that uses different thresholds in its different iterations. For that, we start with a vector $\mathcal{T}^{(k)}$ and decrease it by 10% in every iteration, which results in additional primitives with lower boundary gradients in the following iterations. This algorithm continues until the decrease percentage p reaches the threshold t_{perc}. Note that p is initially set to 1, which implies that the algorithm always uses the initial threshold vector $\mathcal{T}^{(k)}$ in its first iteration.

In each iteration, a graph is constructed on the identified primitives (Section III-B1). Afterwards, predefined structural patterns are searched for on this graph to locate nuclei in an image. The nucleus localization step (explained in the next subsection) locates only nuclei that satisfy a constraint, leaving the others to later iterations, which results in greater location precision. Fig. 4 depicts the flowchart of the search algorithm, where t_{std} corresponds to the threshold that defines the constraint in the nucleus localization step. To find more nuclei in later iterations, this threshold is also relaxed by 10% in every iteration. The details of the constraint (and the threshold) are given in the next subsection.

3) Nucleus Localization: Nuclei are located by searching for two structural patterns on the constructed graph: `4PRIM` and `3PRIM`. The `4PRIM` pattern consists of four adjacent primitives (each of which has a different type) and the edges between these primitives. That is, this pattern should consist of one left, one right, one top, and one bottom primitive forming a connected subgraph. Instances of this pattern are shown in Fig. 5, with their edges in black. As observed in the figure, there can be two different subtypes of this pattern. The first subtype (shown with dashed black edges) corresponds to the ideal case, where all nucleus boundaries have high gradients. Thus, the corresponding primitives form a closed shape (a loop on the graph). The second subtype (shown with solid black edges) corresponds to a less-ideal case, where some parts of the boundaries do not have high enough gradients, probably because of overlayered nuclei. In this case, the boundary primitives of all types exist but they cannot form a closed shape (they form a chain on the graph). The `3PRIM` pattern corresponds to the least ideal case, in which one boundary type cannot be identified at all; it

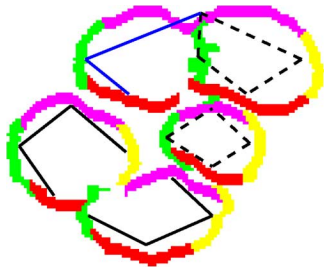


Fig. 5. Two structural patterns used for nucleus localization: `4PRIM` and `3PRIM`. Corresponding edges are shown in black and blue, respectively. The `4PRIM` pattern has two subtypes that correspond to subgraphs, forming a loop (dashed black edges) and a chain (solid black edges).

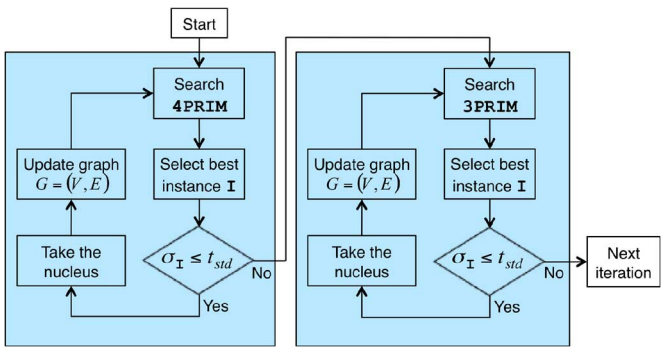


Fig. 6. Flowchart of the nucleus localization method.

may be missing due to overlayered nuclei or some nonideal experimental conditions. Hence, the `3PRIM` pattern should contain three adjacent primitives (each has a different type) and the edges between these primitives. In Fig. 5, these edges are shown in blue.

The nucleus localization step starts with searching instances of the `4PRIM` pattern on the constructed graph $G = (V, E)$, which may contain more than one of such instances. Hence, the proposed localization method selects the best instance that satisfies the shape constraint, takes the segment of each primitive that lies on the correct sides of the others (Fig. 3), updates the primitive vertices V and the edges E of the graph, and continues with the next-best instance. The localization process continues until no instance that satisfies the shape constraint remains. This step continues by searching instances of the `3PRIM` pattern in a similar way. The flowchart of this localization method is given in Fig. 6.

The shape constraint is defined to process round and more-regular-shaped nuclei in the first iterations of the search algorithm (Section III-B2) and deal with more-irregular shapes later. Irregular shapes in previous iterations can turn into round shapes in later iterations, in which additional primitives can be found. To quantify the shape of an instance (nucleus candidate), we use the standard deviation metric. We first identify the outermost pixels o_i of the selected primitive segments (the union of the leftmost, rightmost, topmost, and bottommost pixels of the left, right, top, and bottom primitive segments), calculate the radial distance r_i from every pixel o_i to the centroid C of the outermost pixels, and then calculate the standard deviation σ of the radial distances r_i (Fig. 7). This standard deviation is close to zero for

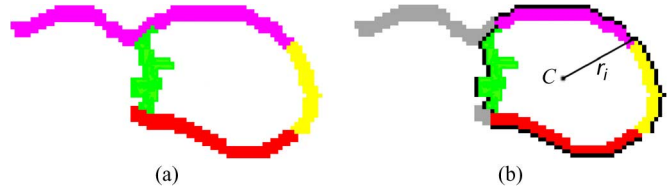


Fig. 7. Determining the outermost pixels and calculating a radial distance r_i. The primitive segments on the correct sides of other primitives are identified and the outermost pixels are selected. Nonselected primitive segments are indicated in gray.

round shapes and becomes larger for more-irregular ones. Thus, the best nucleus candidate is the one with the smallest standard deviation. Moreover, to impose the shape constraint, we define a threshold t_{std}. If the standard deviation of the best candidate is greater than this threshold, we stop searching the current structural pattern. As mentioned, this threshold is relaxed by 10% in every iteration of the search algorithm so that more-irregular-shaped nuclei can be identified.

After selecting the best nucleus instance, we remove its selected primitive segments from the primitive maps and update the graph edges. For example, in Fig. 7, the selected segments (shown in green, yellow, pink, and red) and the edges between them would be removed, whereas the nonselected segments (shown in gray) would be left in the primitive maps. Thus, the same nucleus corresponding to the same set of primitive segments cannot be identified more than once in different iterations. Note that the graph edges for the nonselected segments would be redefined, if necessary.

Fig. 8 illustrates graphs constructed in different iterations with indicating the edges of the `4PRIM` and `3PRIM` patterns that satisfy the standard deviation constraint of the corresponding iteration. Here, the edges of the patterns satisfying the constraint are shown as blue dashed lines and the others as black solid lines. This figure also shows selected primitive segments of these patterns.

The nucleus identification essentially groups primitives to form a nucleus. In that sense, it can be regarded as a contour grouping method, which aims to group contour (edge) fragments to find object boundaries. However, as opposed to previous contour grouping studies [37], [38], our algorithm defines high-level boundary primitives that are attributed to a type and groups them via searching for structural patterns on a relational graph constructed on these attributed primitives.

C. Region Growing

The previous step identifies the primitives of located nuclei. It is simple to delineate an individual nucleus if its primitives form a closed shape; however, this usually does not occur due to noise and overlayering. One might consider connecting the primitives' end points by a line but this might result in unnatural boundaries, especially for `3PRIM` instances. Moreover, some primitives may have been incorrectly assigned to a nucleus in the previous step and must be corrected. For example, in Fig. 9(a), primitives of nuclei for a subimage from Fig. 2(a) are shown. We observe that most nuclei do not have a closed form. Further, the top primitive of the red nucleus (indicated

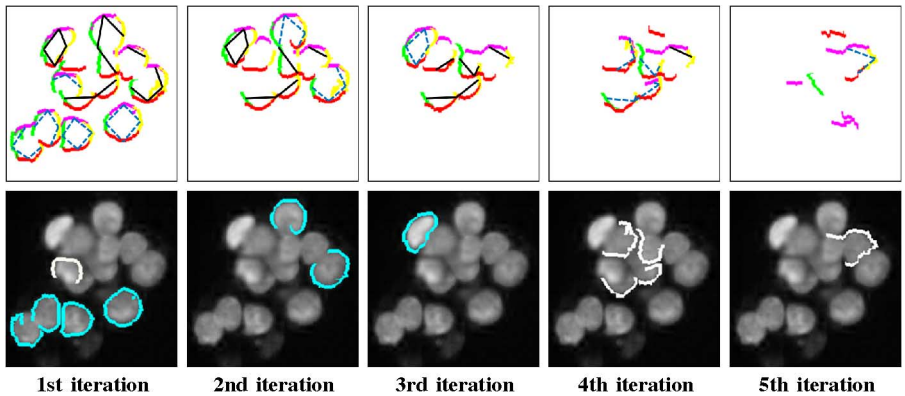


Fig. 8. Illustration of graphs constructed in different iterations and selected primitive segments in these iterations. In the first row, the graph edges of the 4PRIM and 3PRIM patterns that satisfy the standard deviation constraint of the corresponding iteration are indicated as dashed blue lines and the others as black solid lines. In the second row, selected segments of these 4PRIM and 3PRIM patterns are shown in cyan and white, respectively.

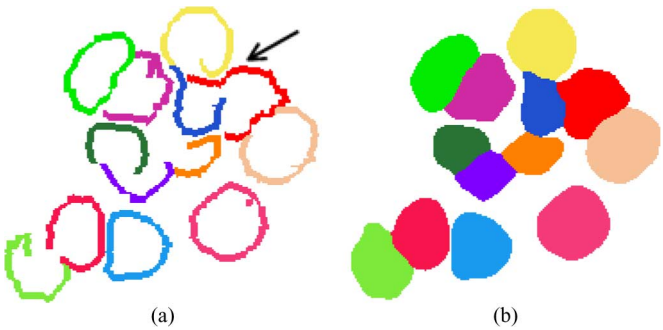


Fig. 9. (a) Primitives identified for different nuclei and (b) nucleus boundaries delineated after the region-growing algorithm.

with an arrow) is incorrectly identified; it contains the top primitive of the blue nucleus next to it.

Thus, to better delineate nuclei, we use a region-growing algorithm that takes the centroids of each nucleus' primitives as starting points and grows them considering the primitive locations. We use the primitive locations for two ends: First, a pixel cannot grow in the direction of the outer boundaries of a primitive as determined by the primitive type; e.g., a pixel cannot grow to the top outer boundaries of the top primitive. Second, for each nucleus, pixels grown after touching a primitive pixel for this nucleus are excluded from the results. Doing this stops growing without the need for an additional mask, and it also results in better boundaries. In this algorithm, we use the geodesic distance from a pixel to a starting point as the growing criterion. Last, to obtain smoother boundaries, we apply majority filtering, using a filter radius of W, on the results and fill the holes inside the located nuclei. Example results obtained by this growing step are shown in Fig. 9(b).

IV. Experiments

A. Dataset

We conduct our experiments on two datasets of fluorescence microscopy images of human hepatocellular carcinoma (Huh7 and HepG2) cell lines. Because the amount of cells growing in overlayers is much higher in the HepG2 cell line, the cell nuclei in that dataset are more overlayered than the Huh7 nuclei. This situation makes segmentation more difficult for the HepG2 dataset. To understand the effectiveness of our proposed algorithm on the nuclei of less overlayered and more overlayered cells, we test it on both sets separately and observe the segmentation performance. These sets contain 2661 nuclei in 37 images, 16 of which belong to the Huh7 and 21 of which belong to the HepG2 cell lines.

Both cell lines were cultured and their images taken in the Molecular Biology and Genetics Department of Bilkent University. The cell lines were cultured routinely at 37 °C under 5% CO_2 in a standard Dulbecco's Modified Eagle Medium (DMEM) supplemented with 10% Fetal Calf Serum (FCS). For visualization, nuclear Hoechst 33258 (Sigma) staining was used. The images were taken at 768 × 1024 resolution, under a Zeiss Axioscope fluorescent microscope with an AxioCam MRm monochrome camera using a 20× objective lens and a pixel size of 3.4 μm × 3.4 μm. We compare the automated segmentation results with manually delineated gold standards, where nuclei were annotated by our biologist collaborators.

B. Evaluation

We evaluate the segmentation results visually and quantitatively. The common metrics for quantitative evaluation are accuracy and precision. Accuracy measures how close the segmentation is to the gold standard, whereas precision measures the reproducibility of multiple segmentations of the same image [39], [40]. In our study, we follow an approach similar to [40] to measure accuracy; however, we do not assess precision because the proposed algorithm is deterministic and always gives the same segmentation for the same image.

We assess accuracy using two methods: nucleus-based and pixel-based. In nucleus-based evaluation, the aim is to assess the accuracy of an algorithm in terms of the number of correctly segmented nuclei. A nucleus is said to be correctly segmented if it *one-to-one matches* with an annotated nucleus in the gold standard. For that, we first match each computed nucleus to an annotated one and vice versa; a computed nucleus N is matched to an annotated nucleus A if at least half of N's area overlaps with A. Next, segmented nuclei that form unique pairs with annotated ones are counted as one-to-one matches, after which nucleus-based *precision*, *recall*, and *F-score* are calculated. Addi-

tionally, we consider *oversegmentations*, *undersegmentations*, *misses*, and *false detections*. An annotated nucleus is oversegmented if two or more computed nuclei match the annotated nucleus, and annotated nuclei are undersegmented if two or more match the same computed nucleus. A computed nucleus is a false detection if it does not match any annotated nucleus and an annotated nucleus is a miss if it does not match any computed nucleus.

In pixel-based evaluation, the aim is to assess accuracy in terms of the areas of correctly segmented nuclei. We use the one-to-one matches found in nucleus-based evaluation and consider overlapping pixels of the computed-annotated nucleus pairs of these matches as true positives. Then, we calulate pixel-based *precision*, *recall*, and *F-score* measures.

C. Parameter Selection

The proposed algorithm has four external model parameters: 1) primitive length threshold t_{size} from the primitive definition and graph construction, 2) percentage threshold t_{perc} from the iterative search algorithm, 3) standard deviation threshold t_{std} from the cell localization, and 4) radius W of the structuring element of the majority filter from the region growing. We select the values of these parameters for the training nuclei. We randomly select five images from each of the Huh7 and HepG2 sets, using 785 nuclei from these 10 images as the training set. We use the nuclei in the rest of the images as the test sets, which include 891 nuclei for the Huh7 and 985 nuclei for HepG2 cell lines.

For parameter selection, we first consider all possible combinations of the following values: $t_{\mathrm{size}} = \{10, 15, 20, 25\}$, $t_{\mathrm{perc}} = \{0.2, 0.3, 0.4, 0.5\}$, $t_{\mathrm{std}} = \{3, 4, 5, 6\}$, and $W = \{4, 5, 6\}$. Then, we select the parameter combination that gives the best F-score for the training nuclei. The selected parameters are $t_{\mathrm{size}} = 15$ pixels, $t_{\mathrm{perc}} = 0.3$, $t_{\mathrm{std}} = 4.0$, and $W = 5$. We discuss the effects of these parameter values on segmentation performance in the next section. Additionally, an internal parameter d is used to define boundary primitives by taking the d-outermost pixels of the binary maps of the Sobel responses. Smaller values of d are not enough to put the boundaries of the same cell under the same primitive, whereas larger values connect the boundaries of different cells. Thus, we internally set $d = 3$, which gives good primitives in our experiments.

D. Comparisons

We use three comparison algorithms: adaptive h-minima [4], conditional erosion [16], and iterative voting [9]. The algorithms from the first two studies implement marker-controlled watersheds; they identify markers based on shape information. Adaptive h-minima [4] obtains a binary segmentation via active contours without edges and calculates an inner distance map that represents the distance from every foreground pixel to the background. Then, it identifies regional minima as the markers, found after applying the h-minima transform to the inverse of the map, and calculates an outer distance map representing the distance from the foreground pixels to their nearest marker. Finally, it grows the markers using the combination of the outer distance map and the gray-scale image as a marking function.

Conditional erosion [16] obtains a binary image by histogram thresholding and iteratively erodes its connected components by a series of two cell-like structuring elements of different sizes. It first uses the larger element until the sizes of the eroded components fall below a threshold. The component shapes are coarsely preserved due to the size of the structuring element and its round shape. It next uses the smaller element on the remaining components and stops the iterations just before the component sizes become smaller than a second threshold. Considering the eroded components as the markers, it then applies a watershed algorithm on the binary image.

Iterative voting [9] defines and uses a series of oriented kernels for localizing saliency, which corresponds to nucleus centers in a microscopic image. This study localizes the centers from incomplete boundary information by iteratively voting kernels along the radial direction. It continues iterations, in which the shape of the kernel and its orientation are refined, until convergence. It then identifies the centers by thresholding the vote image computed throughout the iterations and outputs a set of centers that can be used as the markers in a watershed algorithm. In our experiments, we use the software provided by the authors of [9], available at http://vision.lbl.gov/Publications/ieee_trans_ip07, to find the nucleus centers and apply a marker-controlled watershed algorithm on the binary image obtained by histogram thresholding. These three comparison algorithms have their own parameters. We also select their values for the training nuclei by following the methodology given in Section IV-C. For details of the algorithms' parameters, the reader is referred to the technical report given in [41].

V. Results

In Fig. 10, we present visual results for example subimages obtained by the algorithms. Note that these subimages are of different sizes, but they have been scaled for better visualization. Also note that the sizes of the images from which they are cropped are the same and that we run the algorithms on the original-sized images. Fig. 10 demonstrates that all algorithms can accurately segment the nuclei of monolayer isolated cells (the first column on the left of the figure); however, compared to the others, the proposed algorithm (*ARGraphs*) better segments the nuclei of touching and more overlayered cells. For these nuclei, the other algorithms commonly lead to more undersegmentations, as also observed in the quantitative results (see Table I).

In Fig. 11, we present the number of one-to-one matches in bar graph form. We provide precision, recall, and F-score measures in Table II. These results show that our *ARGraphs* algorithm improves segmentation performance for both datasets compared to the others. The improvement is more notable for the HepG2 dataset, in which cells are more overlayered. These findings are consistent with the visual results, and we attribute them to the following properties of our algorithm. First, conditional erosion and adaptive h-minima heavily rely on an initial map to find their markers. If this map is not accurately determined (which is usually the case) or if there are no background pixels inside a nucleus cluster, these methods are inadequate at separating nuclei in relatively bigger clusters. This failing is observed in the results of these algorithms, given in the second and

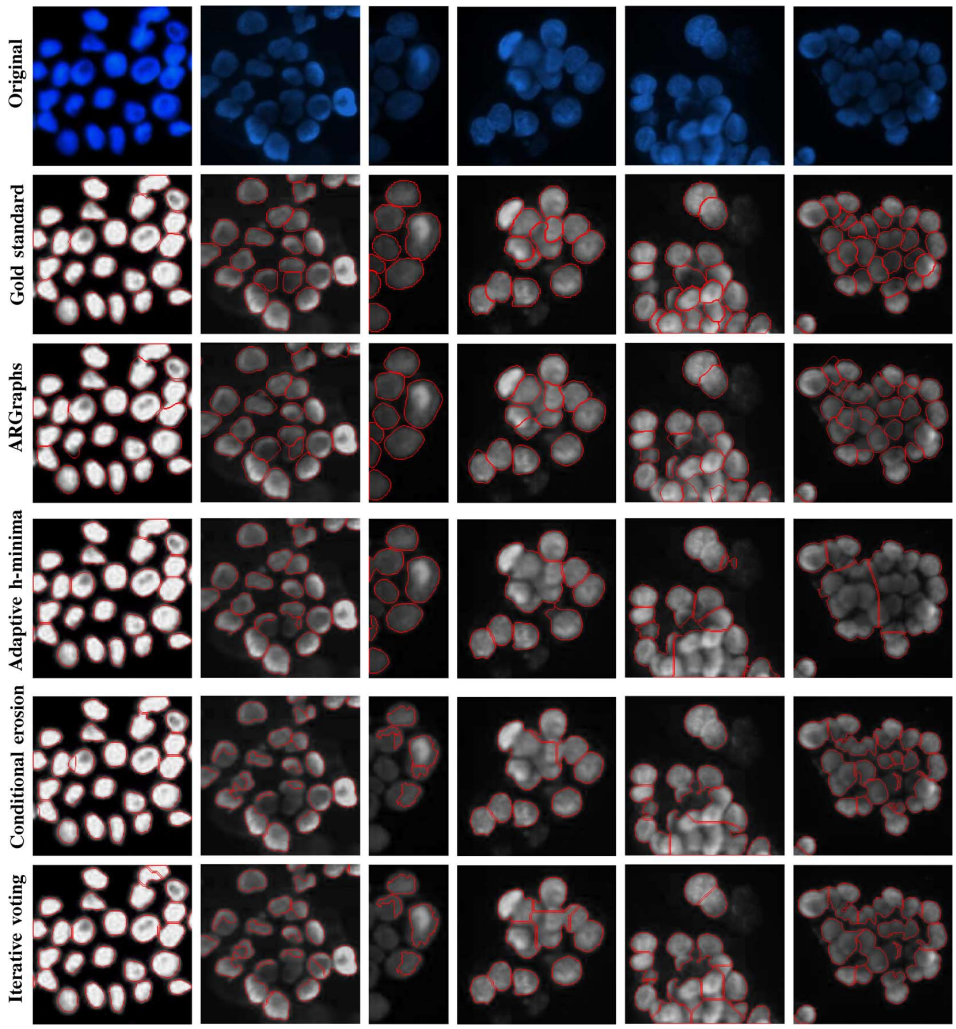


Fig. 10. Visual results obtained by the algorithms for various subimages. The image sizes have been scaled for better visualization.

TABLE I
COMPARISON OF THE ALGORITHMS IN TERMS OF COMPUTED-ANNOTATED NUCLEUS MATCHES ON THE (a) Huh7 AND (b) HepG2 TEST SETS

	One-to-one	Overseg	Underseg	False	Miss
ARGraphs	788	35	51	17	20
Adaptive h-minima	745	11	120	28	15
Conditional erosion	738	27	77	34	49
Iterative voting	721	51	60	59	34

(a)

	One-to-one	Overseg	Underseg	False	Miss
ARGraphs	780	37	116	44	52
Adaptive h-minima	684	4	280	43	17
Conditional erosion	617	15	297	57	56
Iterative voting	721	58	131	75	45

(b)

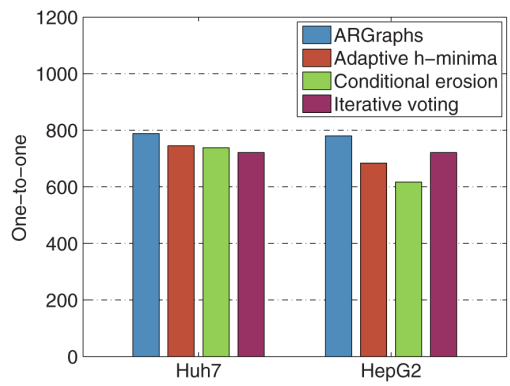


Fig. 11. Number of one-to-one matches for the Huh7 and HepG2 test sets.

third columns of Fig. 10. The problem is more obvious in the results of conditional erosion, which uses global thresholding to obtain its map. Our algorithm uses an initial map to find local thresholds but nowhere else, which makes it more robust to inaccuracies of this map.

Second and more importantly, our algorithm uses the fact that a nucleus should have four boundaries, each of which locates one of its four sides, in its segmentation. This helps our algorithm identify nuclei even when their boundaries are only partially present (using the `4PRIM` pattern) or even when one of the boundaries missing (using the `3PRIM` pattern). In this way, we can (to a degree) compensate for the negative effects of overlayered nuclei. This property is evident in the last three columns of Fig. 10. It is worth noting that iterative voting also does not require an initial map to find nucleus centers; however, in using

TABLE II
COMPARISON OF THE ALGORITHMS IN TERMS OF NUCLEUS-BASED AND PIXEL-BASED PRECISION, RECALL, AND F-SCORE MEASURES ON THE (a) Huh7 AND (b) HepG2 TEST SETS

	Nucleus-based			Pixel-based		
	Precis	Recall	F-score	Precis	Recall	F-score
ARGraphs	88.14	88.44	88.29	78.28	85.51	81.74
Adaptive h-minima	88.27	83.61	85.87	82.57	79.47	80.99
Conditional erosion	85.21	82.82	84.01	86.11	72.61	78.78
Iterative voting	81.28	80.92	81.10	81.26	68.48	74.33

(a)

	Nucleus-based			Pixel-based		
	Precis	Recall	F-score	Precis	Recall	F-score
ARGraphs	81.41	79.19	80.28	65.75	75.37	70.24
Adaptive h-minima	80.37	69.44	74.50	67.16	66.33	66.74
Conditional erosion	73.89	62.63	67.80	64.24	55.86	59.76
Iterative voting	75.89	73.19	74.52	70.67	61.12	65.55

(b)

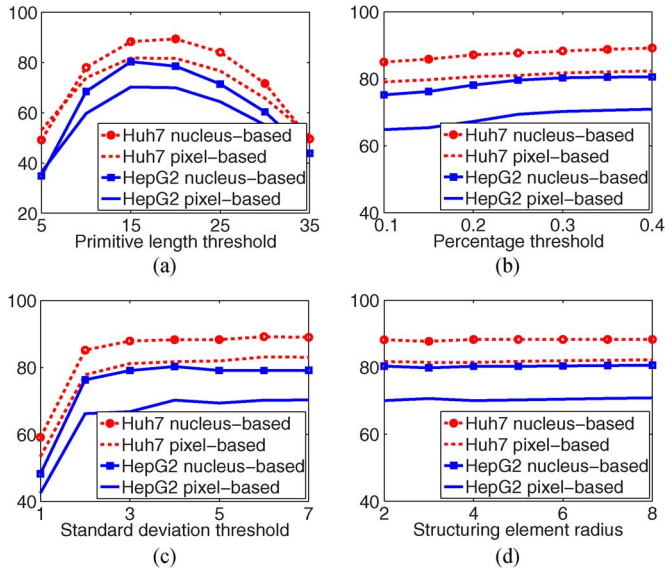


Fig. 12. For the Huh7 and HepG2 test sets, cell-based and pixel-based F-score measures as a function of the (a) primitive length threshold t_{size}, (b) percentage threshold t_{perc}, (c) standard deviation threshold t_{std}, and (d) radius W of the structuring element.

shape information to define its kernels, it may give incorrect results when nuclei appear in irregular shapes due to overlayering. When we analyze the nucleus centers found by iterative voting, we observe that this algorithm usually fails to find nuclei that are partially obscured by others.

Another class of methods uses filters (such as sliding band filters [32] and Laplacian of Gaussians [33]) to detect nuclei by considering the nucleus' blob-like convex shape. They identify places with high filter responses as initial nucleus locations, and further process them to find nucleus borders. These methods can give good initial locations especially for the nuclei of monolayer isolated cells; however, when cell nuclei appear in irregular and nonconvex shapes due to overlayering, they can lead to incorrect initial nucleus locations.

A. Parameter Analysis

We investigate the effects of each parameter on the performance of the proposed algorithm. To this end, we fix three of the four parameters and observe the changes in nucleus-based and pixel-based F-score measures with respect to different values of the other parameter. In Fig. 12, we present the analysis results separately for the Huh7 and HepG2 test sets.

The first parameter is the primitive length threshold t_{size}, which is used in defining the primitives and constructing the graphs. In the former, primitive components whose width/height is smaller than t_{size} are eliminated because they are more likely to be noise than nucleus boundaries. Likewise, in the latter, primitives smaller than t_{size} are not considered part of a structural pattern. Larger values of this parameter eliminate more primitives, especially the ones belonging to small nuclei, and also increase misses and undersegmentations. On the other hand, smaller thresholds increase the number of false primitives, which results in more false detections and oversegmentations. As observed in Fig. 12(a), these both lower the F-score measures.

The second parameter is the percentage threshold t_{perc} used by the iterative search algorithm. This algorithm searches for structural patterns on the graph G, which is constructed on primitives identified by thresholding the Sobel responses. Initial thresholds are computed using the Otsu method and relaxed in later iterations to find more primitives. However, there must be a limit to this relaxation because too-small thresholds (as a result of several iterations) falsely identify noise as primitives. Thus, we use a threshold to stop iterations so that segmentation is not affected by these falsely identified primitives. Smaller values yield more false detections, which lower F-scores, as observed in Fig. 12(b).

The next parameter is the standard deviation threshold t_{std} used in nucleus localization. In the iterative search algorithm, structural patterns are identified as nuclei if they satisfy the shape constraint, which is based on their standard deviation. When smaller thresholds are used, the nucleus candidates need to be rounder to be detected. For this reason, the algorithm can find only a few structural patterns, which lowers the number of one-to-one matches thus decreasing F-scores [see Fig. 12(c)]. The last parameter is radius W of the structuring element of the majority filter. This parameter is used in region growing, which applies majority filtering on segmented nuclei to smooth their boundaries. The results given in Fig. 12(d) show that this parameter affects performance the least.

B. Discussion

The experiments show that our *ARGraphs* algorithm leads to promising results for the nuclei of isolated and overlayered cells. They also show that the algorithm is robust to a certain amount of pixel-level noise, which typically arises from nonideal experimental conditions such as weak staining and poor background illumination. These findings are attributed to the following: The *ARGraphs* algorithm models high-level relational dependencies between the attributed boundary primitives instead of directly using an initial map or gradients defined at the pixel-level. Moreover, it follows an iterative approach to define the primitives, doing so on relatively high Sobel responses in its early iterations and on lower ones in later iterations. Thus, boundaries that appear broken in initial iterations may later become complete.

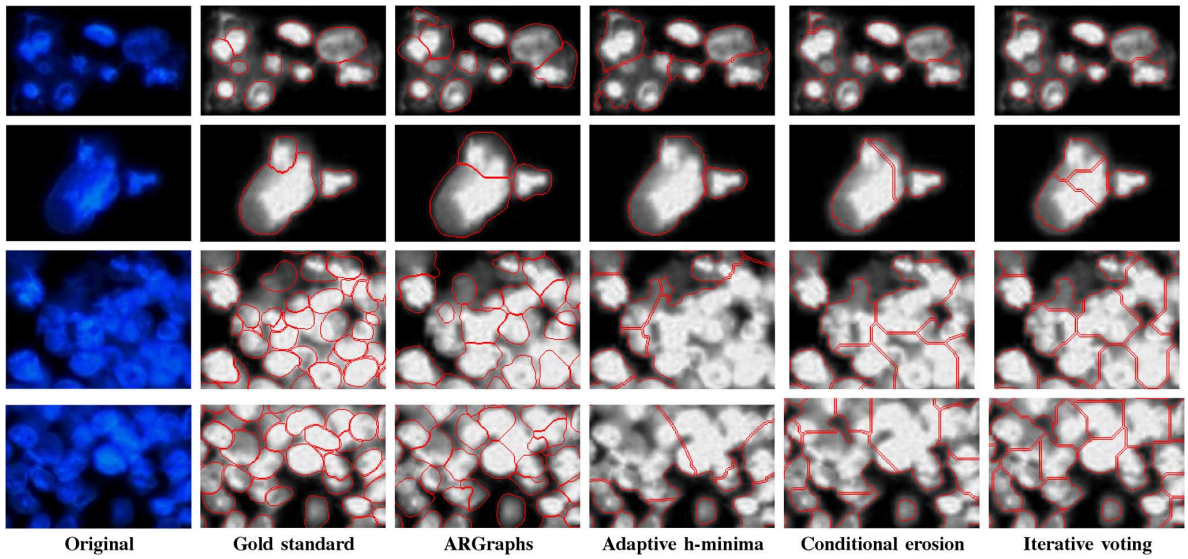


Fig. 13. Top two rows: results for subimages containing inner intensity variation and texture. Bottom two rows: results for subimages containing very dense nucleus clusters. The sizes of the subimages have been scaled for better visualization.

This iterative nature of the algorithm also helps handle inner-nuclear texture to some degree. Because of this texture, it is possible to find pixels with high responses inside a nucleus. However, the magnitudes of these responses are generally lower than those of the pixels found on nucleus contours. For this reason, it is more likely to define primitives on contour pixels and thus to form a nucleus on these primitives in earlier iterations. Because we exclude primitive segments of the selected nucleus from later iterations, primitives defined on textured regions do not typically introduce new nucleus fragments. A primitive can be defined on pixels only if these pixels form a connected component with a length greater than t_{size}. Textured regions can thus lead to primitives when the texture granularity is large enough, but defining a primitive does not automatically define a nucleus; to do that, the primitive must be part of a `3PRIM` or `4PRIM` pattern. This criterion further decreases the likelihood of textured regions leading to unwanted nucleus fragments. We analyze the results of the algorithms on subimages containing some inner-nuclear intensity variation and texture, and as seen in the first two rows of Fig. 13, our algorithm performs better than the others.

When images contain too-dense nucleus clusters with some texture and noise, the accuracy of the proposed algorithm decreases. Examples of such subimages are given in the last two rows of Fig. 13. However, as also evident in this figure, the problem becomes more difficult for all algorithms, and even for humans. As a future work, one could explore incorporating texture features into primitive definition to locate better boundary primitives, which may increase accuracy.

VI. Conclusion

This paper presents a new model-based algorithm for segmenting nuclei in fluorescence microscopy images. This algorithm models how a human locates a nucleus by identifying the boundaries of its four sides. To this end, the model defines high-level primitives to represent the boundaries and transforms nucleus identification into a problem of identifying predefined structural patterns in a graph constructed on these primitives. In region growing, it then uses the primitives to delineate the nucleus borders. The proposed algorithm is tested on 2661 nuclei taken from two different cell lines and the experiments reveal that it leads to better results for isolated monolayer and more overlayered cells than its counterparts.

This work uses an iterative algorithm to search for structural patterns in the attributed relational graph defined on the boundary primitives. This search algorithm uses a greedy approach, which selects the locally optimal instance at each stage. This instance is the one, for which the standard deviation of radial distances from the outer boundaries of its primitives is the smallest. One may consider this algorithm as an approximation of a much harder combinatorial optimization problem. Given a set of primitives defined in different iterations, locate structural patterns to minimize the sum of the standard deviation of all selected instances. This problem requires an exhaustive search to find the globally optimal solution. The greedy approach, however, gives a feasible solution although it does not guarantee finding the global optimum. Different search algorithms could also be implemented to find a better solution to this problem. For example, one could define a probability function for the standard deviation metric of instances. At each stage, an instance could probabilistically be selected using this function instead of selecting the one with the smallest metric. Repeating this algorithm many times, multiple segmentation results would be obtained and combined in an ensemble to find the final segmentation. Exploring different search algorithms is another future research direction for this work.

The proposed model locates the primitives using Sobel operators and attributes them to a type based on the orientation of this operator. We use the Sobel operator because it is relatively simple and effective for our experiments. Nevertheless, another gradient operator could be used to locate the primitives. One could extract features from the primitives and attribute them to

a type, using the extracted features in a classifier. However, in that case, the classifier would need to be trained, which would require determining a set of labels for training primitives, which may not be straightforward. Using different methods to locate and attribute the primitives could be another future research direction.

REFERENCES

[1] X. Chen, X. Zhou, and S. T. Wong, "Automated segmentation, classification, and tracking of cancer cell nuclei in time-lapse microscopy," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 4, pp. 762–766, Apr. 2006.

[2] A. A. Dima, J. T. Elliott, J. J. Filliben, M. Halter, A. Peskin, J. Bernal, M. Kociolek, M. C. Brady, H. C. Tang, and A. L. Plant, "Comparison of segmentation algorithms for fluorescence microscopy images of cells," *Cytom. Part A*, vol. 79A, pp. 545–559, 2011.

[3] C. Wahlby, I.-M. Sintor, F. Erlandsson, G. Borgefors, and E. Bengtsson, "Combining intensity, edge and shape information for 2D and 3D segmentation of cell nuclei in tissue sections," *J. Microsc.*, vol. 315, pp. 67–76, 2004.

[4] J. Cheng and J. C. Rajapakse, "Segmentation of clustered nuclei with shape markers and marking function," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 3, pp. 741–748, Mar. 2009.

[5] M. Wang, X. Zhou, F. Li, J. Huckins, R. W. King, and S. T. C. Wong, "Novel cell segmentation and online SVM for cell cycle phase identification in automated microscopy," *Bioinformatics*, vol. 24, no. 1, pp. 94–101, 2008.

[6] K. Nandy, P. R. Gudla, R. Amundsen, K. J. Meaburn, T. Misteli, and S. J. Lockett, "Automatic segmentation and supervised learning-based selection of nuclei in cancer tissue images," *Cytom. Part A*, vol. 81A, pp. 743–754, 2012.

[7] N. Kharma, H. Moghnieh, J. Yao, Y. P. Guo, A. Abu-Baker, J. Laganiere, G. Rouleau, and M. Cheriet, "Automatic segmentation of cells from microscopic images using ellipse detection," *IET Image Process.*, vol. 1, no. 1, pp. 39–47, 2007.

[8] S. Kumar, S. H. Ong, S. Ranganath, T. C. Ong, and F. T. Chew, "A rule-based approach for robust clump splitting," *Pattern Recognit.*, vol. 39, pp. 1088–1098, 2006.

[9] B. Parvin, Q. Yang, J. Han, H. Chang, B. Rydberg, and M. H. Barcellos-Hoff, "Iterative voting for inference of structural saliency and characterization of subcelluar events," *IEEE Trans. Med. Imag.*, vol. 16, no. 3, pp. 615–623, Mar. 2007.

[10] D. Fenistein, B. Lenseigne, T. Christophe, P. Brodin, and A. Genovesio, "A fast, fully automated cell segmentation algorithm for high-throughput and high-content screening," *Cytometry A*, vol. 73, pp. 958–964, 2008.

[11] Z. Yin, R. Bise, M. Chen, and T. Kanade, "Cell segmentation in microscopy imagery using a bag of local Bayesian classifiers," in *Proc. IEEE Int. Symp. Biomed. Imag.: From Nano to Macro*, 2010, pp. 125–128.

[12] F. Bunyak, K. Palaniappan, S. K. Nath, T. I. Baskin, and G. Dong, "Quantitative cell motility for in vitro wound healing using level set-based active contour tracking," in *Proc. IEEE Int. Symp. Biomed. Imag.: From Nano to Macro*, 2006, pp. 1040–1043.

[13] G. Xiong, X. Zhou, and L. Ji, "Automated segmentation of drosophila RNAi fluorescence cellular images using deformable models," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 11, pp. 2415–2424, 2006.

[14] S. K. Nath, K. Palaniappan, and F. Bunyak, "Cell segmentation using coupled level sets and graph-vertex coloring," in *Proc. Med. Image Comput. Assist. Interv.*, 2006, vol. 9, pp. 101–108.

[15] J. Lindblad, C. Wahlby, L. Ji, E. Bengtsson, and A. Zaltsman, "Image analysis for automatic segmentation of cytoplasms and classification of Rac1 activation," *Cytometry A*, vol. 57A, no. 22–33, 2004.

[16] X. Yang, H. Li, and X. Zhou, "Nuclei segmentation using marker controlled watershed, tracking using mean-shift, and Kalman filter in time-lapse microscopy," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 11, pp. 2405–2414, 2006.

[17] H. Zhou and K. Z. Mao, "Adaptive successive erosion-based cell image segmentation for p53 immunohistochemistry in bladder inverted papilloma," in *Proc IEEE Eng. Med. Biol. Soc.*, 2005, pp. 6484–6487.

[18] C. Jung, C. Kim, S. W. Chae, and S. Oh, "Unsupervised segmentation of overlapped nuclei using Bayesian classification," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 12, pp. 2825–2832, Dec. 2010.

[19] P. S. U. Adiga and B. B. Chaudhuri, "An efficient method based on watershed and rule-based merging for segmentation of 3-D histopathological images," *Pattern Recognit.*, vol. 34, pp. 1449–1458, 2001.

[20] P. R. Gudla, K. Nandy, J. Collins, K. J. Meaburn, T. Mitseli, and S. J. Lockett, "A high-throughput system for segmenting nuclei using multiscale techniques," *Cytometry A*, vol. 73A, pp. 451–466, 2008.

[21] G. Lin, M. K. Chawla, K. Olson, J. F. Guzowski, C. A. Barnes, and B. Roysam, "Hierarchical, model-based merging of multiple fragments for improved three-dimensional segmentation of nuclei," *Cytometry A*, vol. 63A, pp. 20–33, 2005.

[22] X. Zhou, F. Li, J. Yan, and S. T. C. Wong, "A novel cell segmentation method and cell phase identification using Markov model," *IEEE Trans. Inf. Technol. Biomed.*, vol. 13, no. 2, pp. 152–157, Mar. 2009.

[23] G. Lin, U. Adiga, K. Olson, J. F. Guzowski, C. A. Barnes, and B. Roysam, "A hybrid 3D watershed algorithm incorporating gradient cues and object models for automatic segmentation of nuclei in confocal image stacks," *Cytometry A*, vol. 56A, pp. 23–36, 2003.

[24] O. Schmitt and S. Reetz, "On the decomposition of cell clusters," *J. Math. Imag. Vis.*, vol. 22, no. 1, pp. 85–103, 2009.

[25] Q. Wen, H. Chang, and B. Parvin, "A Delaunay triangulation approach for segmenting clumps of nuclei," in *Proc. IEEE Int. Symp. Biomed. Imag.: From Nano to Macro*, 2009, pp. 9–12.

[26] S. Kothari, Q. Chaudry, and M. D. Wang, "Automated cell counting and cluster segmentation using concavity detection and ellipse fitting technique," in *Proc. IEEE Int. Symp. Biomed. Imag.: From Nano to Macro*, 2009, pp. 795–798.

[27] M. Farhan, O. Yli-Harja, and A. Niemisto, "A novel method for splitting clumps of convex objects incorporating image intensity and using rectangular window-based concavity point-pair search," *Pattern Recognit.*, vol. 46, no. 3, pp. 741–751, 2013.

[28] O. Schmitt and M. Hasse, "Radial symmetries based decomposition of cell clusters in binary and gray level images," *Pattern Recognit.*, vol. 41, no. 6, pp. 1905–1923, 2008.

[29] X. Qi, F. Xing, D. J. Foran, and L. Yang, "Robust segmentation of overlapping cells in histopathology specimens using parallel seed detection and repulsive level set," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 3, pp. 754–765, Mar. 2012.

[30] H. Kong, M. Gurcan, and K. Belkacem-Boussaid, "Partitioning histopathological images: An integrated framework for supervised color-texture segmentation and cell splitting," *IEEE Trans. Med. Imag.*, vol. 30, no. 9, pp. 1661–1677, Sep. 2011.

[31] T. Jiang, F. Yang, and Y. Fan, "A parallel genetic algorithm for cell image segmentation," *Electron. Notes Theo. Comp. Sci.*, vol. 46, pp. 1–11, 2001.

[32] P. Quelhas, M. Marcuzzo, A. M. Mendonca, and A. Campilho, "Cell nuclei and cytoplasm joint segmentation using sliding band filter," *IEEE Trans. Med. Imag.*, vol. 29, no. 8, pp. 1463–1473, Aug. 2010.

[33] Y. Al-Kofahi, W. Lassoued, W. Lee, and B. Roysam, "Improved automatic detection and segmentation of cell nuclei in histopathology images," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 4, pp. 841–852, Apr. 2010.

[34] V. S. Lempitsky and A. Zisserman, "Learning to count objects in images," *Proc. Adv. Neural Inf. Process. Syst.*, pp. 1324–1332, 2010.

[35] O. Sertel, G. Lozanski, A. Shana'ah, and M. N. Gurcan, "Computer-aided detection of centroblasts for follicular lymphoma grading using adaptive likelihood-based cell segmentation," *IEEE Trans. Biomed. Eng.*, vol. 57, no. 10, pp. 2613–2616, Oct. 2010.

[36] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, Feb. 2001.

[37] Q. Zhu, G. Song, and J. Shi, "Untangling cycles for contour grouping," in *Proc. IEEE Int. Conf. Comp. Vis.*, 2007, pp. 1–8.

[38] J. H. Elder, A. Krupnik, and L. A. Johnston, "Contour grouping with prior models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 6, pp. 661–674, Jun. 2003.

[39] S. K. Warfield, K. H. Zou, and W. M. Wells, "Simultaneous truth and performance level estimation (STAPLE): An algorithm for the validation of image segmentation," *IEEE Trans. Med. Imag.*, vol. 23, no. 7, pp. 903–921, Jul. 2004.

[40] J. K. Udupa, V. R. LeBlanc, Y. Zhuge, C. Imielinska, H. Schmidt, L. M. Currie, B. E. Hirsch, and J. Woodburn, "A framework for evaluating image segmentation algorithms," *Comput. Med. Imag. Grap.*, vol. 30, pp. 75–87, 2006.

[41] S. Arslan, T. Ersahin, R. Cetin-Atalay, and C. Gunduz-Demir, Attributed relational graphs for cell nucleus segmentation in fluorescence microscopy images: Supplementary Material Comput. Eng., Bilkent Univ., Tech. Rep. BU-CE-1301, 2013 [Online]. Available: http://www.cs.bilkent.edu.tr/tech-reports/2013/BU-CE-1301.pdf