An Automatic Indirect Immunofluorescence Cell Segmentation System

Yung-Kuan Chan, Der-Chen Huang, Kuo-Ching Liu, Rong-Tai Chen, and Xiao-yi Jiang

1 Department of Management Information Systems, National Chung Hsing University, Taichung 402, Taiwan
2 Department of Computer Science and Engineering, National Chung Hsing University, Taichung 402, Taiwan
3 Department of Science and Biotechnology, China Medical University, Taichung 402, Taiwan
4 Department of Computer Science, University of Münster, 48149 Münster, Germany

Correspondence should be addressed to Yung-Kuan Chan; ykchan@nchu.edu.tw

Received 26 February 2014; Accepted 24 April 2014; Published 22 May 2014

Copyright © 2014 Yung-Kuan Chan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Indirect immunofluorescence (IIF) with HEp-2 cells has been used for the detection of antinuclear autoantibodies (ANA) in systemic autoimmune diseases [1]. ANA testing can be used to scan a broad range of autoantibody entities and to describe them by distinct fluorescence patterns. The fluorescence patterns are usually identified by physicians manually inspecting the slides with the help of a microscope. However, due to lacking in satisfying automation of inspection and a low level of standardization, this procedure still needs highly specialized and experienced technicians or physicians to obtain diagnostic result. For this purpose, automatic inspection for fluorescence patterns in an IIF image may assist physicians, without relevant experience, in making correct diagnosis. As ANA testing becomes more popularly used, a functional automatic inspection system is essential and its clinical application becomes more urgent [2].

1. Introduction

IIF with HEp-2 cells has been frequently employed to detect ANA in systemic autoimmune diseases [1]. ANA testing can be used to scan a broad range of autoantibody entities and to describe them by distinct fluorescence patterns. The fluorescence patterns are usually identified by physicians manually inspecting the slides with the help of a microscope. However, due to lacking in satisfying automation of inspection and a low level of standardization, this procedure still needs highly specialized and experienced technicians or physicians to obtain diagnostic result. For this purpose, automatic inspection for fluorescence patterns in an IIF image may assist physicians, without relevant experience, in making correct diagnosis. As ANA testing becomes more popularly used, a functional automatic inspection system is essential and its clinical application becomes more urgent [2].

Roughly, IIF with HEp-2 cells can be classified into six different main patterns: diffuse, peripheral, coarse speckled, fine speckled, discrete speckled, and nucleolar patterns. Figure 1 shows the six distinct autoantibody fluorescence patterns in IIF images. From an image processing point of view, the fluorescence cell belonging to diffuse, peripheral, coarse speckled, or fine speckled pattern normally includes only one connected region. On the contrary, the discrete speckled and nucleolar patterns contain mass and some bright spots [3].

Ideally, a fluorescent image [4] shows only the structure of interest that is labelled with fluorescent dye, while the responses of unstained cells remain unobserved. Specificity of dyes with respect to cell types is sufficient for identifying supporting cells and receptor cells. The application of an array of fluorochromes has made it possible to identify cells and submicroscopic cellular components with a high degree of specificity amid nonfluorescing material. Many different
fluorescent dyes can be used to stain different structures or chemical compounds. The fluorescence microscope is capable of revealing the presence of a single molecule. Through multiple fluorescent labelling, different probes can simultaneously identify several target molecules.

Segmenting cells from an IIF image plays an important role in developing an automatic ANA inspection system. Many cell segmentation methods [1, 5–7] are available to separate cells from an image but cannot be employed to cut cells off well from a fluorescence image. Most of them adopt a global threshold to convert a gray-level image into a binary image, but the intensities of cells in a fluorescence image may lie in an extremely wide range. It is very difficult to detect all cells using a global threshold only. Hence, the methods mentioned above cannot often give a good segmentation result for fluorescence images.

An automatic ANA inspection system can be divided into HEP-2 cell segmentation, fluorescence pattern classification, and computer aided diagnosis phases. This study will focus on the first phase, developing an effective method for automatically cutting off the cells with fluorescence pattern from IIF images.

In this study, a color selector is presented to transform a color IIF image into a gray-level image, a run length enhancer is provided to suppress the hazy fluorescent halo, an adaptive filter is proposed to smooth the surfaces as well as filling in the holes on the cells, and a gradient computing method is given to effectively compute the gradients of an image. These proposed techniques can be also applied to segment objects from other kinds of images. In addition, watershed and distance transform techniques are used to split overlapping objects. Since the characteristics of the six cell patterns in IIF images are extremely different, a rough classifier is proposed to decide which set of the parameter values should be employed in the proposed segmentation method for segmenting cells from the IIF images with different patterns.

Cell culture [8] is the process by which prokaryotic, eukaryotic, or plant cells are cultivated under controlled conditions [9]. This process allows individual cells to act as an independent unit. In cell culture, the size and quantity of cells will increase during cell division, and the growth of cells will be constrained by culture variables such as nutrient depletion. Culture techniques have become an important and extensive part of biotechnology in applied research areas such as genetics, cytology, and pathology. Cell culture also brings huge economic benefits and prospects to agriculture, crop genetics, crop breeding, and food additives.

Since the growing cells must be separated into two or more dishes to avoid excessive growth in cell culture, biologists need to know the precise number or density of cells. Quantity measurement is an important task in cell culture. In addition, assessing medium suitability, determining population doubling times, and monitoring cell growth in culture all require ways to quantifying cell population. Cell quantification also allows standardization for manipulations such as transfection or cell fusion.

Generally, biologists count the cells under microscope one by one or utilize the chambers. The most widely used chamber is the hemocytometer, a device designed for estimating the number of cells in a given volume under a microscope. This device separates the whole-cell samples into several large, equal squares, and each large square contains many small squares. Counting the cells in several small squares or larger squares provides an estimate of the total number of cells. However, manually counting the cells on an
image that contains an extremely large number of cells is a time-consuming, arduous, and inaccurate task. An automatic software system to count the cells on an image taken under a microscope would save time and frustration and improve accuracy.

The cells in an IIF image generally lie in a wide range of gray level intensities; even the intensities of some cells are too weak for human eyes to find them. The purpose of this paper is to develop a system for automatically segmenting the cells in an indirect immunofluorescence (IIF) image. We call it automatic indirect immunofluorescence cell segmentation (AICS) system which will enhance the cell contours and cut off the cells from an IIF image. The AICS system also can be used to count the cells on the IIF image.

In the AICS system, a color selector is proposed to transform a color IIF image into a gray-level image, a run length enhancer and an adaptive filter to get rid of noise and highlight the contours of objects on the IIF image, a new gradient computing method to compute the gradient of object contours, and an adaptive thresholding method to decide the most proper thresholds for detecting the cell contours on an IIF image. Moreover, a rough classifier is provided to classify IIF images and GAPD to decide the most suitable parameters used in the AICS system.

2. Related Works

In this section, we will briefly review some cell segmentation methods [1, 5–7], the performances of which will be compared with the performance of the AICS system in cell segmentation and counting. In addition, some measurements of object segmentation and counting errors [10] will be briefly reviewed as well.

2.1. Reviews. Eddins [5] presented an image segmentation strategy for isolating objects from an image. The objects could be anything: blood cells, stars, toner spots on a printed page, DNA microarray elements, or even quantum semiconductor dots. First, the strategy converts the image into a grayscale one and uses a morphological top-hat operator with a disk-shaped structuring element to smooth out the uneven illumination. Second, Otsu’s thresholding method [11] is employed to determine a good threshold for converting the image to a binary one. Finally, this strategy computes the distance transform of the complement of the binary image, and then uses the watershed transform to extract objects.

Althoff et al. [1] provided a cell contour segmentation method to cut off the neural stem cells from a time-lapse image sequence. The method is an iterative process for every image in the sequence and can be described as follows.

(i) Use multiscale Laplacian of Gaussian filters to separate image background from cell regions.

(ii) Select the centroids of the blobs that are most likely to be cells.

(iii) Use a dynamic programming to segment the cells.

<table>
<thead>
<tr>
<th>Test outcome</th>
<th>Ground truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster 1</td>
<td>True positive (TP)</td>
</tr>
<tr>
<td>Cluster 2</td>
<td>False negative (FN)</td>
</tr>
</tbody>
</table>

Tang and Ewert [6] also offered a method for severing neural stem cells from a sequence of images. This method first applies a Gaussian filter to remove noise from the images. It then performs fuzzy thresholding as follows: all pixels with intensity below a lower threshold \( t_l \) are set to 0, and all pixels with intensity above a higher threshold \( t_h \) are set to 1. The gray level intensities between \( t_l \) and \( t_h \) are linearly rescaled to the range [0, 1]. The chosen lower threshold \( t_l = \mu + 0.3\sigma \), where \( \mu \) is the mean value and \( \sigma \) is the standard deviation of the background intensity. Similarly, the chosen higher threshold \( t_h = \mu + 4\sigma \) is high enough to guarantee that the pixels brighter than \( t_h \) are really well inside the cells. Through the use of a fuzzy thresholding approach, the method becomes less sensitive to the exact values of these threshold levels than would have been the case if a standard crisp threshold had been used. Next, fuzzy gray weighted distance transform and watershed transform are used to separate cells from images.

In 2008, Yan et al. [7] proposed a method for segmenting the nuclei of cells from a genome-wide RNAi screening image. The method converts an image into a gray-scale one and applies Otsu’s thresholding method [11] to decide a suitable threshold for converting the grayscale image to a binary image. It then computes the distance transform of the complement of the binary image, employs an enhanced watershed algorithm to detect the nucleus contours of cells, and removes the objects with small areas.

2.2. Classification and Segmentation Measures. Positive predictive value and sensitivity are two most frequently used criteria to evaluate the effectiveness of a classifier. Positive predictive value can be seen as a measure of exactness or fidelity, whereas sensitivity is a measure of completeness. Table 1 lists all possible classification conditions in a binary classification. The positive predictive value PPV and sensitivity SEN are defined as follows [10]:

\[
PPV = \frac{TP}{TP + FP}, \quad SEN = \frac{TP}{TP + FN},
\]

There may be a trade-off problem between positive predictive value and sensitivity; greater positive predictive value decreases sensitivity and greater sensitivity leads to decreased positive predictive value. \( F \)-measure \( F \) is the harmonic mean of PPV and SEN and takes account of both measures [12]:

\[
F = \frac{2}{\frac{1}{PPV} + \frac{1}{SEN}} = \frac{2TP}{2TP + FN + FP}.
\]
3. AIICS System

The AIICS system contains two stages: preprocessing and cell segmenting. The preprocessing stage is to enhance cell contours and suppress noise contours for facilitating further cell segmenting. The cell segmenting stage is to cut off cells contours and suppress noise contours for facilitating further cell segmenting. The preprocessing stage is to enhance cell contours. The AIICS system then projects the hth pixel with three color components \((C_R, C_G, C_B)\) in \(f_{RGB}\) on a line passing through \((0, 0, 0)\) and \((C_R, C_G, C_B)\), and assigns the value, obtained by projecting \((C_R, C_G, C_B)\) on the line, to the gray-level intensity of the hth pixel of \(f_0\). Let max and min be the maximal and minimal gray-level intensities of all the pixels in \(f_0\). The AIICS system then changes the gray-level intensity \(x\) of each pixel in \(f_0\) into \((x - \min) / (\max - \min)\) × 255. This operation is to stretch the contrasts of \(f_0\) from 0 to 255.

3.1. Preprocessing

In the preprocessing stage, the AIICS system first transforms a color IIF image \(f_{RGB}\) into a gray-level image \(f\). The AIICS system computes the color histogram of \(f_{RGB}\) with 6 bins. Let black (background color), red, green, blue, cyan, and yellow be the representative colors of the six bins, respectively, and let \((C_{R,1}, C_{G,1}, C_{B,1})\) be the RGB color components of the representative color of the hth bin. In \(f_{RGB}\) for each pixel \(p\) with three color components \((C_R, C_G, C_B)\), \(p\) will be thrown into bin \(j\) where

\[
J = \arg\left(\min_{i=0}^{5} \sqrt{(C_{R,i} - C_R)^2 + (C_{G,i} - C_G)^2 + (C_{B,i} - C_B)^2}\right).
\]

(4)

Let \(n_i\) be the number of pixels which are thrown into bin \(i\), and \(k\) is defined as \(k = \arg(\max_{i=0}^{5} n_i)\). The AIICS system then projects the hth pixel with three color components \((C_R, C_G, C_B)\) in \(f_{RGB}\) on a line passing through \((0, 0, 0)\) and \((C_{R,k}, C_{G,k}, C_{B,k})\), and assigns the value, obtained by projecting \((C_R, C_G, C_B)\) on the line, to the gray-level intensity of the hth pixel of \(f_0\). Let max and min be the maximal and minimal gray-level intensities of all the pixels in \(f_0\). The AIICS system then changes the gray-level intensity \(x\) of each pixel in \(f_0\) into \((x - \min) / (\max - \min)\) × 255. This operation is to stretch the contrasts of \(f_0\) from 0 to 255.

3.1.2. Run Length Enhancer

In IIF images, cells are bright objects protruding out from a uniform dark background. However, some cells are often surrounded by a hazy fluorescent halo, such as the image in Figure 3(a). The hazy fluorescent halo will make cell segmentation more difficult. Therefore, this paper proposed a run length enhancer to suppress the hazy fluorescent halo. For each pixel \(f_0(i, j)\) located at the coordinates \((i, j)\) on \(f_0\), the run length enhancer draws eight line segments \(l_0, l_1, \ldots, l_7\), all of which pass through \(f_0(i, j)\), and the included angle between \(l_h\) and X-axis is 22.5 × \(h\). Let \(W_r\), consisting of \(m_r\) × \(m_r\) pixels, be a window on \(f_0\), where \(f_0(i, j)\) is the central pixel of \(W_r\). We call \(W_r\) the related window of \(f_0(i, j)\). Assume that \(l_h\) passes through pixels \(P_{h1}\) and \(P_{h2}\) which are located on the boundary of \(W_r\). Figure 4 illustrates the related window \(W_r\) of \(f_0(i, j)\) marked by red color. If the gray-level intensity of \(f_0(i, j)\) is greater than the average gray-level intensity of all the pixels in \(W_r\), then the run length enhancer assigns \(\max_{h=0}^{7}(1/m_r) \sum q_{rh}\) to the gray-level intensity of \(f_0(i, j)\); otherwise it gives \(\min_{h=0}^{7}(1/m_r) \sum q_{rh}\) to the gray-level intensity of \(f_0(i, j)\), where \(c_{rh}\) is the gray-level intensity of each pixel on line segment \(l_h\) from \(P_{h1}\) to \(P_{h2}\).
Figure 3: The result of preprocessing stage.

(a) $f_0$
(b) $f_r$
(c) $f_a$

3.1.3. Adaptive Filter. Figure 3(b) shows that some cells are uneven and there may exist some holes on the cells in $f_r$, indicated by a red arrow; therefore, the AIICS system adopts an adaptive filter to smooth the surfaces and to fill in the holes on the cells. Let $\text{Ave}_g$ be the average gray-level intensity of all the pixels in $f_r$, let $W_a$, consisting of $m_x \times m_y$ pixels, be a related window of $f_r(i, j)$, and let $\text{Ave}_a$ as well as $\text{Min}_a$ be the average and minimal gray-level intensities of all the pixels in $W_a$. Since cell pixels in $f_r$ are generally brighter than background pixels, the adaptive filter gives

$$
\begin{align*}
\text{Ave}_a, & \quad \text{if } \text{Ave}_a \geq \text{Ave}_g \\
\text{Min}_a, & \quad \text{otherwise},
\end{align*}
$$

(5)

to $f_a(i, j)$. After that, the adaptive filter stretches the pixel contrast of $f_a$ from 0 to 255. Figure 3(c) is the $f_a$ obtained by executing the adaptive filter on the image in Figure 3(b). Figure 3(c) shows that the adaptive filter can smooth the surfaces and fill in the holes on the cells in $f_r$.

3.2. Cell Segmenting. This stage is to cut off cells from an IIF image. It consists of gradient computing, contour extracting, and overlapping cells splitting steps.

3.2.1. Gradient Computing. The gradients of pixels in an image can provide an abundance of object contour information. Let $\nabla f_a$ denote the gradient vector field of pixel $f_a(i, j)$. The run length enhancer then stretches the gray-level contrast of $f_r$ from 0 to 255. Figure 3(b) is the $f_r$ obtained by executing the run length enhancer on image $f_0$ in Figure 3(a). Figure 3(b) explains that the run length enhancer can suppress the hazy fluorescent halo.

Figure 4: The eight line segments in run length enhancer.
where $\nabla$ is the vector differential operator. The components of $\nabla f_a$ are the partial derivatives of $f_a(i, j)$. That is,

$$
\nabla f_a = [g_x, g_y] = \left[ \frac{\partial f_a}{\partial x}, \frac{\partial f_a}{\partial y} \right].
$$

(6)

The magnitude of the vector is given to

$$
\text{Mag}(\nabla f_a) = (g_x^2 + g_y^2)^{1/2} = \left( \left( \frac{\partial f_a}{\partial x} \right)^2 + \left( \frac{\partial f_a}{\partial y} \right)^2 \right)^{1/2}.
$$

(7)

Let $g_{xx}, g_{xy}, g_{yx},$ and $g_{yy}$ be

$$
g_{xx} = \frac{\partial^2 f_a}{\partial x^2}, \quad g_{xy} = \frac{\partial^2 f_a}{\partial x \partial y},
$$

$$
g_{yx} = \frac{\partial^2 f_a}{\partial y \partial x}, \quad g_{yy} = \frac{\partial^2 f_a}{\partial y^2},
$$

and let $\nu = (v_x, v_y)$ be a unit vector in XY plane. We define the local contrast of $f_a(i, j)$ at the XY plane in the direction of $\nu$ as

$$
g_{xx}v_x^2 + 2g_{xy}v_xv_y + g_{yy}v_y^2,
$$

$$
A = \begin{pmatrix} g_{xx} & g_{xy} \\ g_{yx} & g_{yy} \end{pmatrix}.
$$

(9)

It is well known that a quadratic form equation (9) has a maximal and a minimal value for varying $\nu$. These extreme values coincide with the eigenvalues of matrix $A$, and they are attained when $\nu$ is the corresponding eigenvector [13]. The extreme values can be obtained by (10)

$$
\lambda_+ = \frac{g_{xx} + g_{yy}}{2} \pm \sqrt{\left(\frac{g_{xx} - g_{yy}}{4}\right)^2 + g_{xy}g_{yx}}.
$$

(10)

For each pixel $f_a(i, j)$, a pair $(\lambda_+, \lambda_-)$ is given, so that $\lambda_+$ is the greatest gradient magnitude of $f_a(i, j)$ in a certain gradient direction $\theta_+; \lambda_-$ is the gradient of $f_a(i, j)$ in other gradient direction $\theta_-$ and the included angle between $\theta_+$ and $\theta_-$ is $90^\circ$.

After computing the $\lambda_+$ and $\lambda_-$ of all the pixels in $f_a$, the $\lambda_+$ and $\lambda_-$ of $f_a(i, j)$ are, respectively, transformed into

$$
f_+(i, j) = 255 \times \frac{\lambda_+ - \min_+}{\max_+ - \min_+},
$$

$$
f_-(i, j) = 255 \times \frac{\lambda_- - \min_-}{\max_- - \min_-}.
$$

(11)

where $\max_+$ and $\min_+$ are the maximum and the minimum $\lambda_+$ of all the pixels in $f_a$ and $\max_-$ and $\min_-$ are the maximum and the minimum $\lambda_-$ of all the pixels in $f_a$. Figures 5(b) and 5(c) demonstrate the $f_+$ and $f_-$ obtained by the gradient computing from the image $f_a$ shown in Figure 5(a).

From Figure 5, one can obviously observe that $f_+$ and $f_-$ can effectively describe the gradient of $f_a$ but the object contours in $f_+$ (resp. $f_-$) are expanded (resp. shrunk) compared to the real object contours in $f_a$. Hence, $f_+$ and $f_-$ are integrated into $f_b$ by the following formula:

$$
f_b(i, j) = \frac{f_+(i, j)}{(255 - f_-(i, j)) + 1},
$$

where the 1 in denominator is to ensure the value in denominator not to be zero.

Next, the contrasts of all the pixels in $f_b$ are stretched to range from 0 to 255. Figure 5(d) is the $f_b$ obtained by executing the gradient computing approach on the image in Figure 5(a). Figure 5(d) shows that the gradient computing operation can provide the impressive gradient of an image.

3.2.2. Contour Extracting. Since the gradients of some object contours are indistinct, such as the gradient indicated by a red arrow in Figure 5(d), the AIICS system takes an adaptive thresholding approach to transform $f_b$ into a binary image $f_h$ in which a 1-bit (resp. 0-bit) represents one white pixel (resp. black pixel). The adaptive thresholding approach first uses Otsu's thresholding method [11] to partition all the pixels in $f_b$ into two groups according to their gray-level intensities, then respectively computes the average gray-level intensity of all the pixels in each group, and considers the smaller one $\mu_b$ of the two average gray-level intensities to be the gray-level intensity of the background pixels in $f_b$.

Let $W_h$, consisting of $m_b \times m_h$ pixels, be the related window of $f_b(x, y)$, then $\mu_b$ be the average gray-level intensity of all the pixels in $W_h$, and let $\sigma_b$ be the standard deviation of the gray-level intensities of all the pixels in $f_b$. Two thresholds $T_1$ and $T_2$ are set to

$$
T_1 = \mu_b + \frac{\sigma_b}{2}, \quad T_2 = \mu_b + \frac{\sigma_b}{2}.
$$

(13)

The $f_h(x, y)$ is given to be

$$
f_h(x, y) = \begin{cases} 1, & (f_b(x, y) > T_1) \text{ or } \quad (T_2 < f_g(x, y) \leq T_1 \text{ and } \mu_b > T_1), \\ 0, & \text{otherwise}. \end{cases}
$$

(14)

Figure 6(a) illustrates the $f_h$ generated by the adaptive thresholding approach from the image in Figure 5(d).

Since the contrast of the pixel at the vicinity of the cell contour or noise is generally high, it may cause a false contour with the thickness of over one pixel. The contour of an object should be one pixel in thickness. The AIICS system adopts a hit-and-miss transform-based skeletonization (HMTS) algorithm [14] to thin the contour of an object so that the contour of an object is the thickness of one pixel. We refer to the eliminated candidate contour pixels as the redundant-contour pixels and the remaining contour pixels as true-contour pixels. We call this algorithm the thinning operation.

Let $W_f(x, y)$, consisting of $3 \times 3$ pixels, be the related window of $f_h(x, y)$. The thinning operation compares $W_f(x, y)$
with each of the eight structuring elements shown in Figure 7, where the gray pixels stand for the do-not-care pixels (A do-not-care pixel may be a 1-bit pixel or a 0-bit pixel). The thinning operation considers that \( W_t(x, y) \) is matched if the positions and values of 1- and 0-bits on one structuring element are completely the same as those on \( W_t(x, y) \), regardless of the do-not-care pixels. If \( W_t(x, y) \) is matched, \( f_t(x, y) = 0 \); otherwise, \( f_t(x, y) = f_b(x, y) \). The algorithm repeats this procedure until no more thinning can be performed. The thinning algorithm is to cut off the redundant-contour pixels, so that the contours have a thickness of only one pixel. Figure 6(b) shows the result after running the thinning operation on the image in Figure 6(a).

However, the uneven object contours tend to cause small spurs on the obtained skeleton by the thinning operation. These spurs are not expected. Therefore, a spur pruning operation is required to remove them. The procedure of the spur pruning operation [14] is entirely the same as the thinning operation except for the eight structuring elements in Figure 7 which are replaced by the eight structuring elements in Figure 8. Let \( f_s \) be the result obtained by the spur pruning operation from \( f_t \). Figure 6(c) displays the result image \( f_s \) after running the spur pruning operation on the image \( f_t \) shown in Figure 6(b).
Mathematical Problems in Engineering

In \( f_s \), some false contours should be removed. Let \( P_i \) and \( P_j \) be two pixels on the contour of the \( k \)-th object \( O_k \) in \( f_s \), and let \( d_{ij} \) be the distance between \( P_i \) and \( P_j \). Assume \( d_{k,k} \) is the maximal one among all the \( d_{ij} \)'s on the contour of \( O_k \). We call \( d_{k,k} \) the major length of \( O_k \). If the major length of \( O_k \) is less than the one-third of the average major length of all the object contours in \( f_s \), then \( O_k \) is considered to be noise and removed from \( f_s \). After that, \( f_s \) in Figure 6(c) is changed into \( f_c \) in Figure 6(d).

3.2.3. Overlapping Cells Splitting. Some juxtaposed objects may share overlapping regions, which will make cell distinguishing and counting imprecise. We call the area containing some touching or overlapping cells an overlapping area. The watershed algorithm is commonly used to process the overlapping object [15–18]. Watershed based segmentation can give good results for gray level images with different minima and catchment basins. For a binary image, however, there are only two gray levels 0 and 1, respectively, standing for background and cell. If two cells are connected together in the binary image \( f_c \), only one minimum and catchment basin will be formed in the topographic surface. In employing the watershed algorithm to segment connected cells, distance transform [19] is often adopted to preprocess the image to make it more suitable for segmentation by watershed algorithm, for assigning the catchment basins in the overlapping area.

Distance transform [19, 20] labels each pixel in an overlapping area with the distance of the pixel to the nearest pixel outside the overlapping area. Let \( R \) be an overlapping area in \( f_c \). For each pixel \( P(x, y) \) in \( R \) located at the coordinates \((x, y)\) on \( f_c \), the distance transform can be determined by the following three steps.

1. Attach a label \( d(x, y) = \infty \) to each pixel \( P(x, y) \) in the overlapping area on \( f_c \) and give a label \( d(x, y) = 0 \) to each pixel outside the overlapping area.

2. Then travel through \( f_c \) pixel by pixel. Replace the label of each pixel \( P(x, y) \) in \( f_c \) with \( \text{Min}(d(x, y), d(x-1, y-1)+1, d(x-1, y)+1, d(x-1, y+1)+1, d(x, y-1)+1, d(x, y)+1, d(x, y+1)+1, d(x+1, y-1)+1, d(x+1, y)+1, d(x+1, y+1)+1) \).

3. Repeat step (2), until all labels have been converted to finite values.

The most intuitive way to explain the watershed based segmentation is to imagine that a hole is drilled in each minimum of the surface and we flood water into different catchment basins from the holes. If the water of different catchment basins is likely to merge due to further immersion, a dam is built to prevent the merging. This flooding process will eventually reach a stage when only the top of dam (the watershed lines) is visible above the water line.

Let \( P_iP_j \) be the line segment passing through \( P_i \) and \( P_j \) which are two pixels on the contour of the \( k \)-th object \( O_k \) in \( f_c \), let \( d_{ij} \) be the distance between \( P_i \) and \( P_j \), let \( D_M = d_{min} \) be the major length of \( O_k \), where \( d_{min} \) is the longest distance between any two pixels on the contour of \( O_k \), and let \((x_1, y_1)\) as well as \((x_2, y_2)\) be the coordinates of \( P_i \) and \( P_j \) on \( f_c \). Assume \( D_m = d_{f_{min}} \) is the distance of pixels \( P_i \) and \( P_{m'} \) on the contour of \( O_k \), \( P_iP_{m'} \) passes through \((x_1 + x_2) / 2, (y_1 + y_2) / 2 \), and \( P_iP_{m'} \) is perpendicular to \( P_iP_m \). We call \( D_⊥ \) the minor length of \( O_k \).

Let \( A_k \) be the area of \( O_k \) and let \( A_{ave} \) be the average area of all the objects on \( f_c \). The AIICS system considers the object \( O_k \) to be an overlapping area and takes the watershed algorithm [17] to split \( O_k \) into more objects, only if

\[
\frac{D_M}{D_⊥} > 1.5, \quad A_k > 2 \times A_{ave}.
\] (15)

After that, \( f_c \) is converted into other binary image \( f_{w_i} \). Figure 9(b) demonstrates the \( f_{w} \) obtained by running this watershed algorithm on the \( f_c \) of the image in Figure 9(a), where only the overlapping areas surrounded by white closed curves in Figure 9(b) are further split by the watershed algorithm.

There may be still some cells split into more than one region. The AIICS system tries to combine them. Let \( \mu_A \) and \( \sigma_A \) be the average and standard deviation of the areas of all the object regions on \( I_{w} \). If the area of object region \( O_k \) is less than \( \mu_A - 0.5 \times \sigma_A \), the AIICS system merges \( O_k \) into one object region which is adjacent to \( O_k \) and has the longest common edge with \( O_k \). After region combination, \( f_{w_i} \) is converted into \( f_m \). Figure 9(c) displays the \( f_m \) obtained from the \( f_{w_i} \) in Figure 9(b) by the region combination.

4. Rough Classifier

There are six distinct autoantibody fluorescence pattern cells which have quite different fluorescence characteristics. To effectively sever the cells from the IIF images, the authors roughly classify IIF images into two categories, uniform pattern category and fleck pattern category. Most of the cells on an image in uniform pattern category are with coarse speckled pattern, diffused pattern, fine speckled pattern, or peripheral pattern, while most cells on an image in fleck pattern category are with discrete speckled pattern or nucleolar pattern. The nuclei of the cells on the images in fleck pattern category have bright spots. Since both image categories have significantly different properties, the AIICS system will take a rough classifier to classify IIF images and use distinct parameter values to segment the cells from the images.

Given an IIF image, the rough classifier first applies Otsu’s thresholding method [11] to determine a threshold \( T_b \), for converting the IIF image into a binary image. A pixel is considered to an object pixel if its gray-level intensity is
Mathematical Problems in Engineering

5. Genetic Algorithm Based Parameter Detector (GAPD)

According to the experimental results, the AIICS system can give better results when given different $m_r$ and $m_a$ for segmenting cells from fleck pattern category images and from uniform pattern category images. Assume that $m_r$ and $m_a$ are given to be $m_r = m_r^*$ as well as $m_a = m_a^*$ for segmenting cells greater than $T_b$; else it is considered to be one background pixel. If the average area of all the cell regions is less than a given threshold $T_{\text{area}}$, then the image is classified to fleck pattern category otherwise to uniform pattern category. Then, the AIICS system will take different $m_r$ and $m_a$ to segment cells on the images from different categories. In this paper, we will use a genetic algorithm to compute the most suitable $T_{\text{area}}$, $m_r$, and $m_a$. 

Figure 9: The results of overlapping cells splitting.

Figure 10: Example for GAPD.
from fleck pattern category images and $m_r = m''_r$ as well as
$m_a = m'_a$ for segmenting cells from uniform pattern category
images.

The performance of the AIICS system is significantly
affected by the values of $T_{area}'$, $m_r'$, $m''_r$, $m_a'$, $m''_a$, and $m_b$ in segmenting the cells on an IIF image. In this paper,
a genetic algorithm based parameter detector (GAPD) is
provided to determine the most suitable values of $T_{area}'$, $m_r'$,
$m''_r$, $m_a'$, $m''_a$, and $m_b$. GAPD makes use of a binary string,
consisting of six substrings $s_{area}'$, $s_{r}'$, $s_{a}'$, $s_{r}''$, $s_{a}''$, and $s_b$ which,
respectively, comprise $n_{area}'$, $n_{r}'$, $n_{a}'$, $n_{r}''$, $n_{a}''$, and $n_b$ binary
bits, to represent a chromosome $Ch$. $s_{area}'$, $s_{r}'$, $s_{a}'$, $s_{r}''$, $s_{a}''$, and $s_b$ can be used to describe $T_{area}'$, $m_r'$, $m''_r$, $m_a'$, $m''_a$, and $m_b$ in segmenting cells from IIF images. For each chromosome $Ch$,
$T_{area}'$, $m_r'$, $m''_r$, $m_a'$, $m''_a$, and $m_b$ can be encoded as $100 + 5 \times n_{area}'$, $2n_r' + 3$, $2n_a' + 3$, $2n_r'' + 3$, $2n_a'' + 3$, and $2n_b' + 3$, where $n_{area}'$, $n_{r}'$, $n_{a}'$, $n_{r}''$, $n_{a}''$, and $n_b$ are the numbers of $1$-bits in $s_{area}'$, $s_{r}'$, $s_{a}'$, $s_{r}''$, $s_{a}''$, and $s_b$, individually.

GAPD applies the accumulated historical IIF images to
train the most appropriate values of $T_{area}'$, $m_r'$, $m''_r$, $m_a'$, $m''_a$, and $m_b$ by using a genetic algorithm. The manually drawn cell regions are considered as a collection of ground truths. The relative distance error (RDE) [21] is often used to measure the segmentation errors of a segmentation method. GAPD will use RDE as the measure of fitness of $Ch$ based on the values of $T_{area}'$, $m_r'$, $m''_r$, $m_a'$, $m''_a$, and $m_b$ encoded by $Ch$.

Initially, GAPD randomly generates $N$ chromosomes,
each with $n_{area}' + n_r' + n_a' + n_{r}'' + n_{a}'' + n_b$ binary bits.
To evolve the best solution, the genetic algorithm repeatedly
executes mutation, crossover, and selection operations until
the relative fitness values of the reserved chromosomes are
very similar to one another or the number of iterations is
equal to MAXRUN.

In mutation operation, for each of the $N$ reserved chro-
mosomes, GAPD uses a random number generator to specify
one bit $b$ from each of the substrings $s_{area}'$, $s_{r}'$, $s_{a}'$, $s_{r}''$, $s_{a}''$, and $s_b$. Then, all the $b$'s are replaced by $\neg b$'s to generate a new
chromosome, where $\neg$ signifies the operator "NOT."
\[
\text{Ch}_1 \left[ (n_{\text{area}} + n_{r'} + n_{a'} + n_{r''} + \left\lfloor \frac{n_{a''}}{2} \right\rfloor + 1) \cdots n_{\text{area}} + n_{r'} + n_{a'} + n_{r''} + n_{r'''} + (n_{b'} + n_{b''} + n_{b'''} + n_{b}) \right],
\]
\[
\text{Ch}_2 \left[ n_{\text{area}} + n_{r'} + n_{a'} + n_{r''} + n_{r'''} + (n_{b'} + n_{b''} + n_{b'''} + n_{b}) \right],
\]
\[
\text{Ch}_3 \left[ (n_{\text{area}} + n_{r'} + n_{a'} + n_{r''} + n_{r'''} + \left\lfloor \frac{n_{a''}}{2} \right\rfloor + 1) \cdots n_{\text{area}} + n_{r'} + n_{a'} + n_{r''} + n_{r'''} + n_{b'} + n_{b''} + n_{b} \right],
\]

(17)

into another new chromosome.

In selection operation, \( N \) optimal chromosomes are selected from the \( N \) chromosomes reserved in the previous iteration and \( N \) as well as \( 2 \times N \) chromosomes created in the mutation and crossover operations, respectively, according to their fitness. GAPD continuously performs the three operations—mutation, crossover, and selection, until the related fitness of the reserved \( N \) chromosomes are very close. Finally, GAPD employs the chromosome \( \text{Ch} \) with the best fitness in the \( N \) reserved chromosomes to describe \( T'_{\text{area}}, m'_r, m''_r, m'_a, m''_a, m'_b, \) and \( m''_b \) in segmenting cells from IIF images.

Figure 10(a) is a chromosome \( \text{Ch} \) indicating \( T'_{\text{area}} = 110, m'_r = 9, m''_r = m''_a = m''_b = 7, \) and \( m'_b = 5 \), where \( n_{\text{area}} = n_{r'} = n_{a'} = n_{r''} = n_{r'''} = n_{b'} + n_{b''} = n_{b} = 4 \). Figure 10(b) displays a new chromosome \( \text{Ch}' \) derived from \( \text{Ch} \) by the mutation operation, where the underlined bits are the randomly selected bits \( b \). Figure 10(c) displays two new chromosomes, \( \text{Ch'}_1 \) and \( \text{Ch'}_2 \), generated from chromosomes \( \text{Ch}_1 \) and \( \text{Ch}_3 \) by the crossover operation.

6. Experiments

This section is to explore the performance of the AIICS system in segmenting and counting the cells on an IIF image by experiments. In these experiments, 195 IIF images, 160 of which are in uniform pattern category and 35 of which are in fleck pattern category, are used as test images. Figure 1 shows a part of the test images. One can obviously observe that there are quite different characteristics with these images. The AIICS system hence employs rough classifier to classify an IIF image into fleck category or uniform pattern category and gives different values of \( m_{r} \) and \( m_{a} \) in segmenting the cells on the IIF image.

The first experiment is to compute the most suitable parameters \( T'_{\text{area}}, m'_r, m''_r, m'_a, m''_a, m'_b, \) and \( m''_b \). In this experiment, 17 of the 160 uniform pattern images and 3 of the 35 fleck pattern images are randomly selected to train \( T'_{\text{area}}, m'_r, m''_r, m'_a, m''_a, \) and \( m'_b \) via GAPD, where \( N = 10, N' = 10, n_{r'} = n_{a'} = n_{r''} = n_{a''} = n_{b'} + n_{b''} = n_{b} = 10, \) and MAXRUN = 200. The experimental results show that the AIICS system will give better segmentation results when \( T'_{\text{area}} = 210, m'_r = 3, m''_r = 11, m'_a = 17, m''_a = 3, \) and \( m'_b = 3 \) in segmenting cells from IIF image. In the following experiments, \( T'_{\text{area}}, m'_r, m''_r, m'_a, m''_a, m'_b, \) and \( m''_b \) are set to the values obtained in this experiment.

The purpose of the second experiment is to probe the performance of the gradient computing method proposed in this paper. In this experiment, the AIICS system is applied to segment cells out from 20 IIF images, except that the gradient computing method is replaced by Sobel [22], Prewitt [22], Roberts [22], and Laplacian [22]. The 20 IIF images, including 1242 cells, are randomly selected from the 195 test images. Misclassification error (ME) [23], relative area error (RAE) [23], modified Hausdorff distance (MHD) [23], and relative distance error (RDE) [21] are often adopted to measure the segmentation errors. In this experiment, they are used to evaluate the performances of Sobel, Prewitt, Roberts, and Laplacian and the gradient computing method proposed in this paper. Table 2 shows the results of this experiment. The experimental results indicate that our method is much better than Sobel, Prewitt, Roberts, and Laplacian methods in computing the gradient of an image.

The third experiment is to explore the performances of the AIICS system, Eddins’, Althoff et al.’s, Tang et al.’s, and Yan et al.’s methods in segmenting cells on an IIF image. In this experiment, the 20 test images employed in the second experiment are also used as test images; ME, RAE, MHD, and RDE are employed to evaluate the segmentation errors. Table 3 demonstrates the results obtained in this experiment. The fourth experiment is to scrutinize the performance of the AIICS system and to compare this performance with those of Eddins’ [5], Althoff et al.’s [1], Tang and Ewert [6], and Yan et al.’s [7] methods in counting cells on an IIF image after segmenting the cells out. Tables 4, 5, and 6 show the experimental results, respectively, obtained by using all 195 IIF images, only 160 uniform pattern images, and only 35
Table 5: The results of experiment 4 by using 160 uniform pattern images as test data.

<table>
<thead>
<tr>
<th></th>
<th>Eddins</th>
<th>Althoff</th>
<th>Tang</th>
<th>Yan</th>
<th>AIICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPV</td>
<td>0.45</td>
<td>0.73</td>
<td>0.89</td>
<td>0.89</td>
<td>0.99</td>
</tr>
<tr>
<td>SEN</td>
<td>0.79</td>
<td>0.43</td>
<td>0.73</td>
<td>0.80</td>
<td>0.92</td>
</tr>
<tr>
<td>DACC</td>
<td>0.41</td>
<td>0.37</td>
<td>0.67</td>
<td>0.73</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Table 6: The results of experiment 4 by using 35 fleck pattern images as test data.

<table>
<thead>
<tr>
<th></th>
<th>Eddins</th>
<th>Althoff</th>
<th>Tang</th>
<th>Yan</th>
<th>AIICS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPV</td>
<td>0.81</td>
<td>0.39</td>
<td>0.74</td>
<td>0.67</td>
<td>0.94</td>
</tr>
<tr>
<td>SEN</td>
<td>0.32</td>
<td>0.11</td>
<td>0.66</td>
<td>0.58</td>
<td>0.82</td>
</tr>
<tr>
<td>DACC</td>
<td>0.30</td>
<td>0.09</td>
<td>0.54</td>
<td>0.45</td>
<td>0.78</td>
</tr>
</tbody>
</table>

Table 7: The classification results obtained by the rough classifier.

<table>
<thead>
<tr>
<th>Test outcome</th>
<th>Uniform</th>
<th>Uniform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground Truth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uniform</td>
<td>160</td>
<td>4</td>
</tr>
<tr>
<td>Fleck</td>
<td>0</td>
<td>31</td>
</tr>
</tbody>
</table>

Table 8: The segmentation results of the erroneously classified images obtained by the AIICS system with $(m_r = 17, m_a = 3)$ and $(m_r = 3, m_a = 11)$.

<table>
<thead>
<tr>
<th></th>
<th>$(m_r = 17, m_a = 3)$</th>
<th>$(m_r = 3, m_a = 11)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPV</td>
<td>0.98</td>
<td>1.00</td>
</tr>
<tr>
<td>SEN</td>
<td>0.94</td>
<td>0.87</td>
</tr>
<tr>
<td>DACC</td>
<td>0.92</td>
<td>0.87</td>
</tr>
</tbody>
</table>

fleck pattern images as the test data. The experimental results indicate that the AIICS system is much superior to Eddins’ [5], Althoff et al’s [1], Tang and Ewert [6], and Yan et al’s [7] methods in counting cells on an IIF image.

The fifth experiment is to investigate the effect of rough classifier. In this experiment, 160 uniform pattern images and 35 fleck pattern images are used as test data. Table 7 shows the experimental results in classifying IIF images to the uniform pattern category or to the fleck pattern category by rough classifier. The classification measures obtained in this experiment are PPV = 0.988, SEN = 0.943, and $F = 0.965$.

Table 7 displays that four images in fleck pattern category are erroneously classified to uniform pattern category. After that, in this experiment, the AIICS system is used to separate the cells from the four images with $(m_r = 17, m_a = 3)$ and $(m_r = 3, m_a = 11)$, respectively. Table 8 demonstrates the experimental results which tell that it is more appropriate to categorize the four images to the uniform pattern category in cell segmentation by the AIICS system.

**7. Conclusions**

In this paper, the AIICS system is proposed to automatically segment and count cells on an IIF image. The experimental results show that the AIICS system can effectively segment and count the cells on an IIF image, even though the characteristics of the IIF images are extremely different. In this paper, color selector, run length enhancer, and adaptive filter are provided to remove noise and to enhance the contours of cells on an image. To cut off the cells from an image effectively, a new gradient computing method is proposed to compute the gradient of edge and an adaptive threshold method is given to decide the most proper thresholds for detecting the cells on an image. In addition, rough classifier is presented to classify IIF images. GAPD is presented to decide the most suitable parameters used in the AIICS system. These techniques can be also applied in segmenting the objects from other kind of images.

**Conflict of Interests**

The authors declare that they have no conflict of interests regarding the publication of this paper.

**References**


