An Automated Gridding and Segmentation Method for cDNA Microarray Image Analysis

Wei-Bang Chen¹, Chengcui Zhang¹, and Wen-Lin Liu²
¹Dept. of Computer and Information Sciences, University of Alabama at Birmingham, USA
²Dept. of Management, Marketing, and Industrial Distribution, University of Alabama at Birmingham, USA

Abstract

Gridding and spot segmentation are two critical steps in microarray gene expression data analysis. However, the problems of noise contamination and donut-shaped spots often make signal extraction process a labor-intensive task. In this paper, we propose a three-step method for automatic gridding and spot segmentation. The method starts with a background removal and noise eliminating step, and then proceeds in two steps. The first step applies a fully unsupervised method to extract blocks and grids from the cleaned data. The second step applies a simple, progressive spot segmentation method to deal with inner holes and noise in spots. We tested its performance on real microarray images against a widely used software GenePix. Our results show that the proposed method deals effectively with poor-conditioned microarray images in both gridding and spot segmentation.

1. Introduction

DNA microarray is a powerful tool that allows biologists to monitor expression levels of thousands of genes simultaneously by using an array of DNA molecules that allow many hybridization experiments to be performed in parallel [1].

The DNA microarray experiment includes the process of hybridization, fluorescent image acquiring and data analysis. After fluorescent image is obtained, analysis software uses a grid to recognizing the position of spots on the chip and then determines the boundary for each spot [2]. Since the microarray images are often contaminated with noise, the automatic gridding and boundary detection process are usually not satisfied [3].

Several approaches have been proposed for automatic gridding. These methods include the use of Markov random field (MRF) [4] [5], template matching and seeded region growing method [6], the axis projections of image intensity [7] and so on [4]. These approaches usually require the numbers of rows and columns as mandatory input parameters. One of our main objectives is to generate grids in a fully automatic manner, without any user input.

Problems in nowadays tools for microarray image data analysis include inner holes, artifacts, blanks, etc. Inner holes problem corresponds to the situation that the spots may form a donut or comet shape, even overlap, due to the manufacturing quality of the slide. Another problem is uneven background or the scratch, noise on the background. Scratches are usually caused by touching the spots area accidentally; noises are often due to the inadequate washing; uneven background may occur as a result of improper counterstain. All the above problems also make automatic image processing require manual tuning. In such cases, human operators need to recognize the spots and fine-tune the boundary for each spot manually, making this process time-consuming and labor-intensive. In general, it usually takes an experienced operator approximately 20 to 40 minutes to deal with the gridding and fine-tuning issues for a single microarray slide with 43,200 spots on it.

The performance of spot segmentation will substantially affect the results of subsequent stages of the analysis [8]. There are several existing segmentation methods to analyzing microarray images: 1) The fixed circle segmentation and adaptive circle segmentation are two similar methods. The former uses a fixed radius circle on the spot while the later uses a circle with adjustable radius to fit the spot [9]. 2) Clustering-based segmentation uses k-means clustering and partitioning around medoids (PAM) to generate binary partition of the pixels based on the distribution of their intensities [10]. 3) Adaptive shape segmentation commonly uses watershed method or seeded region growing method which relies heavily on the selection of a seed point [9, 11]. 4) Adaptive thresholding method computes a threshold based on the Mann-Whitney(MW) test [12]. 5) Histogram methods use the histogram of a masked area to determine foreground and background [9]. 6) The Markov random field modeling [9]. 7) Gradient-based spot segmentation uses morphological analysis [13].

Since both gridding and spot segmentation are challenging tasks, they have been generally dealt with separately in the literature [9]. In this study, we propose a three-step approach to automatic gridding and spot segmentation combining three steps: background identification and noise removal, fully automatic gridding, and spot segmentation. In the first step, a pre-labeling process is carried out to identify pixels that are most likely background pixels by using a local-global thresholding technique, and then remove them from the foreground. It is followed by a noise pre-eliminating process in which a voting method based on spatial connectivity is applied to eliminate the majority of noise. The cleaned and pre-labeled data is then passed on to the second step, in which blocks and grids are generated in a fully automatic manner, without any user intervention. The third step takes the gridded data and performs a simple yet effective method to spot segmentation. The segmentation method
proposed here is based on a progressive two-class clustering scheme. Although more complex spot segmentation methods have been proposed (see the previous paragraph), our spot segmentation method is much simpler and we have found it to be effective. In experiments we compare our method with a widely used software GenePix [15] and found our results to be more robust.

Section 2 describes the details of the proposed approach. Section 3 presents the experimental results. Section 4 concludes this paper.

![Figure 1. A typical microarray image](image)

**2. Methods**

A microarrayer produces a microarray slide by repetitively moving the spotting pins and printing the DNA sample onto the slide with a specific pattern. In Figure 1, a typical cDNA microarray slide is illustrated. We can observe that a typical microarray slide is composed of blocks and spots. A spot which is the basic unit of microarray image represents a gene. Spots are grouped into one or more blocks. In our case, we have 900 (30×30) spots within a block and 48 (4×12) blocks on a slide. This illustration also demonstrates a serious smear around the bottom that makes the problems of noise and uneven background even worse.

For each microarray slide, it will generate two images, corresponding to two different fluoresce dyes such as Cy3 (532nm) and Cy5 (635nm). In order to analyze and compare the signal intensities in these two channels, it is necessary to differentiate fluoresce signals from background, noise and other artifacts.

### 2.1. Background removal

In this step, pixels that are the most likely to be background pixels are identified and removed from the foreground. There are two approaches to identify the background pixels: the global approach and the local approach. The global approach uses a single threshold value to extract the background pixels while the local approach divides the entire slide into small areas and finds a proper threshold for each small area dynamically. In general, the local approach outperforms the global approach in identifying background pixels. However, the simple local approach may not be able to find a proper threshold for a small area if most of the pixels are background pixels. Our approach uses both global threshold and local threshold to recognize background pixels. To find the global threshold, we first divide the entire slide into a set of small areas $A$. 

$$
A \in \mathbb{R}^{m \times n} \leftrightarrow A = \{a_{ij} = \begin{bmatrix}
    a_{11} & \cdots & a_{1n} \\
    \vdots & \ddots & \vdots \\
    a_{m1} & \cdots & a_{mn}
\end{bmatrix}, a_{ij} \in \mathbb{R}
$$

where $m$ and $n$ are the dimensions of the small area, and $a_{ij}$ are the pixels in that area. It is worth mentioning that using smaller area gives better result, but the size of it must be larger than the spot size to ensure there are background pixels in each row and each column. For each small area, we calculate the average intensity of all pixels ($\bar{A}$) in that area and use that to represent this area. The average intensity should be higher in an area containing at least one spot than that of the area containing background pixels only. A matrix $E$ containing all the average intensity values is then used to represent the entire slide.

$$
E \in \mathbb{R}^{rc} \leftrightarrow E = \{\bar{A}_i j = \begin{bmatrix}
    \bar{A}_{11} & \cdots & \bar{A}_{1c} \\
    \vdots & \ddots & \vdots \\
    \bar{A}_{r1} & \cdots & \bar{A}_{rc}
\end{bmatrix}, \bar{A}_i j \in \mathbb{R}
$$

where $r \times c$ is the total number of areas. As shown in Figure 1, the margin areas outside the spot area should contain no spot. In each row and column of $A$, since the size of $A$ is usually much smaller than the width of the margins, we can assure that there is at least a small area, which is composed of background pixels only. Based on this assumption, we find the minimum value in each row and each column of the matrix $E$ and compute the average intensity $m_E$ and the standard deviation $\sigma_E$ as follows:

$$
H_i = E(i,.) = [\bar{A}_{i1}, \bar{A}_{i2}, \cdots, \bar{A}_{ic}]^T
$$
$$
V_j = E(.,j) = [\bar{A}_{1j}, \bar{A}_{2j}, \cdots, \bar{A}_{rj}]^T
$$
$$
m_E = \{\sum_{i=1}^{r} \min(H_i) + \sum_{j=1}^{c} \min(V_i)\}/(r + c); T_0 = m_E + 3 \times \sigma_E
$$

where $H_i$ and $V_j$ are vectors of the $i^{th}$ row and $j^{th}$ column in $E$. We assume that the intensity values of background pixels and signal pixels are two normally distributed populations. The two populations might overlap since the background intensity values might be uneven. To preserve all possible signal pixel values, we use $m_E + 3 \times \sigma_E$ as the global threshold ($T_G$) to represent the upper-bound for the low intensity background. In case of normal distribution, 3 standard deviations accounts for 99.7% of the low intensity background pixel values. Since we choose the minimum value for each row and column, the average intensity $m_E$ is relative low and thus is a good estimate of the average intensity of background.

The local approach is similar to the global one. Within each area $A$, we find the minimum pixel value in each row and column, and use $m_L + 4 \times \sigma_L$ as the local threshold ($T_L$). Here $m_L$ and $\sigma_L$ are computed similarly to $m_E$ and $\sigma_E$. The basic assumption here is that in a small area, the intensity of a signal (spot) pixel is higher than that of a background
pixel. In addition, since we collect the minimums in each row and column, the \( m_i \) here is usually below the average of the background. This is especially the case for the sample slide shown in Figure 1 in which background intensities increases towards the bottom of the slide. Therefore, we use a relatively higher threshold \( m_i + 4 \times \sigma \) to eliminate not only the background pixels with low intensity, but also some of the background pixels with low-moderate intensities locally.

With the global threshold and the local threshold, the pixels with intensities lower than the global threshold are first removed from the foreground. Next, in each small area, the pixels with intensities higher than the global threshold but lower than the local threshold are eliminated as well. By doing so, most of the background pixels can be removed.

2.2. Pre-eliminating noise

After the majority of the background being removed, the next issue is the noises, which usually have similar intensity level with the signals. To eliminate noise, we observe that signal pixels usually group together to form a round-shaped spot. Therefore, if a pixel has its intensity greater than the threshold of the background but does not connect well to its neighbors, then it could be a noise pixel. Based on such observations, our proposed method examines a pixel by using a 5 by 5 lookup window as shown in Figure 2. The center of the matrix is the target pixel surrounded by its 24 neighbors. As mentioned above, since signals usually group together to form a round-shaped spot, if a pixel is a real signal, it should help to shape and form the spot in a positive way. Therefore, we may check the intensities of its neighbors in pairs such as \( P_1 \) & \( P_1' \), \( P_2 \) & \( P_2' \), and \( P_3 \) & \( P_3' \) etc. If the intensities of both neighbors in pair are zero (background pixels), then we vote the target pixel for noise. Otherwise, we vote the target pixel for signal. When the number of noise votes reaches 5, we say the target pixel is a noise. Figure 2 shows two examples. The dark cells indicate the background pixels and the white cells stand for signals. In Figure 2(a), we have only 4 votes (\( P_4 \) & \( P_4' \), \( P_5 \) & \( P_5' \), \( P_9 \) & \( P_9' \) and \( P_{10} \) & \( P_{10}' \)) for noise. Since the number of votes for noise is 4, the target pixel is considered to be a signal. Similarly, in Figure 2(b), since the number of votes for noise is 5, the target pixel is considered to be noise.

2.3. Finding margins

Margin removal is an inseparable step into automatic gridding as margins do not contain any spot. To detect margin boundaries, we implement an algorithm to continuously scan from the four edges of the slide toward the center of the slide. As a result, two vertical scan processes (one upward, one downward) and two horizontal scan processes (one leftward and one rightward) are initiated. The number of signal pixels in each scan-line and the maximum length of the continuous background pixels are recorded. When the maximum length of continuous background pixels drops sharply, and the number of signals continues to increase for five scan-lines, the scan process stops. We then track back seven scan-lines and set the current row/column as a margin boundary. The purpose of tracing back seven pixels is to ensure that no signal pixel will be left outside the margin boundary accidentally.

![Figure 2. Example of signal and noise pixels](image)

2.4. Detecting blocks

As shown in Figure 1, a microarray image typically consists of one or more blocks. It is also known that the spots are better aligned within a block than across blocks. Therefore, it would be better to generate grids in block level rather than in slide level.

To detect the boundary for each block without knowing the number of blocks in each row/column, we make use of the following three observations as shown in Figure 3. First, the gap between two blocks is wider than the gap between two continues rows of spots. Second, the continuity of background pixels is usually better in block gap areas. Third, as shown in Figure 3, if we form a box around a dividing line between two rows of spots, it will cover lots of signals. On the other hand, if we form a box around a block boundary, the box will contain fewer signals. We use the above observations to implement an effective algorithm to detect block boundaries.

![Figure 3. Lines between spot rows and lines between blocks](image)
block gaps and which lines belong to the gaps between rows of spots, we examine all the 3 criteria mentioned at the beginning of Section 2.4 for each candidate line. A line is selected as a block boundary candidate if both of its neighboring line(s) and itself satisfy all the criteria. This process will preserve almost all the lines between blocks and remove the majority of lines between spot rows.

Since there could be more than one block boundary lines within each block gap, we need to find the representative line that sits midway between two neighboring blocks. To that end, we first group adjacent lines into a band, and generate a central line to represent that band. The distance values between any two continuous central lines are then examined. Such distance values should be almost identical as all blocks are similar in size and are placed on the slide at a relatively fixed gap.

Therefore, if the variance of the distance values is very small, the process stops. Otherwise, it will further check the width of each band. Since the width of a block gap is significantly wider than that of a gap between spot rows, we build a histogram of the band widths with two bins. The bin with a larger value of width will be used to eliminate the bands containing lines between spot rows.

### 2.5. Gridding

There is yet another step prior to gridding – generating a bounding box for each group of spatially connected signals. A group containing too few pixels will be ignored. Ideally, a group is a rough estimate of the corresponding spot. However, some may contain more than one. Gridding is to draw a line between every two rows and every two columns of spots, such that the position of each spot within the grids can be determined. This is the so-called ‘addressing’ process. In order to automatically generate grids without knowing the number of rows and columns in a block, we implement the following algorithm.

![Illustration of eligible bounding boxes](image)

**Figure 4. Illustration of eligible bounding boxes**

First we notice that the majority of the bounding boxes contain only one spot and can be used to generate the average height and width of a spot. For that purpose, the statistic distribution of the heights and the widths of all bounding boxes is examined to find the mode ($Mo$) for heights and for widths. The mode is defined as the most frequently occurring value in a given set of data. Since the majority of bounding boxes containing only one spot, the $Mo$ should be capable of representing the average height and width. To allow a little variation in the data, we extend the selection range and use $Mo \pm 3$ pixels as the criterion to select the boxes within. Bounding boxes with heights and/or widths falling into that range are considered to be ‘eligible boxes’ and can be selected for estimating grid lines. Figure 4 shows some examples of those ‘eligible boxes.’

In addition, the centroids of the bounding boxes in a row, using their x-coordinates only, show a large distance value (in terms of their x-coordinates) between rows ($\Delta X$) and a small distance value within the same row ($Ax$), as shown in Figure 5. Therefore, we also take the advantage of this pattern to separate the rows and columns in a block. Take row detecting for example, we first sort the selected boxes in the list by the x-coordinates of their centroids and then compute the difference of the x-coordinates of two adjacent boxes. For the reason aforementioned, those difference values within the same row should be small since spots are usually well aligned. Based on this observation, we group those bounding boxes with $Ax$ less than a threshold and then use the centroids of those boxes to generate a center line for that spot row represented by the group.

![The differences of x-coordinates between rows and within a row](image)

**Figure 5. The differences of x-coordinates between rows and within a row**

Finally, after we obtain the center line for each spot row, the grid lines are automatically generated by finding the mid-line between each pair of adjacent center lines.

### 2.6. Segmentation of spots

When the gridding/addressing has been done, the locations of spot centers and their bounding boxes are already known. In addition, after background removal and noise pre-elimination, an initial labeling of signal pixels is available for further/finer spot segmentation. To this end, we propose a simple yet effective segmentation method that utilizes Otsu’s thresholding algorithm [14] in a progressive manner. For each target spot region, a local threshold is chosen to minimize the intra-class variance and the between-class variance of the thresholded ‘black’ and ‘white’ pixels. The threshold $th$ is positioned midway between the means of the two classes (‘black’ and ‘white’).

Let the means of the gray levels in the two classes be $m_b$ (mean of the ‘black’ class) and $m_w$ (mean of the ‘white’ class), respectively, and the number of pixels with values greater than $th$ be $N_{th}$. $N_{th}$ is actually the number of pixels in the ‘white’ class. $th$ is then set to the value at which

$$N_{th} (N - N_{th}) (m_b - m_w)^2$$

is maximized. $N$ is the total number of pixels in that spot region.

Within a single spot region, the pixels that are real signals (or background) should have similar intensities. Pixels that belong to artifacts such as noise or a scratch will tend to have much higher intensities that are different
from either. In addition, pixels within an inner hole (in presence of donut-shaped spots) or on the outer rim of a spot may belong to background or form a class of its own but with relatively low intensities. As a result, when we apply the two-class clustering to the pixel classified as ‘foreground’ in the initial labeling, there are usually two cases:
1. \( N_0=N \): this case corresponds to the situation where all the pixels pre-labeled as ‘foreground’ are real signals.
2. \( N_0<N \): this case indicates that part of the ‘foreground’ pixels belong to noise, inner holes, or outer rims. While noise pixels usually have high intensities and occupy a small part of the ‘foreground’ region, pixels in inner holes or on outer rims typically have relatively low intensities and may occupy a significant part of the ‘foreground’ region. As a result, we have the following two sub-cases:
   a. \((N_0/N) < \varphi\) where \( \varphi \) is a small value around 0.2: this actually corresponds to the presence of noise with local high intensities. In this case, the pixels identified as ‘white’ are considered noise and removed from the foreground, followed by another round of two-class clustering towards the downsized set of ‘foreground’ pixels. This is the way we apply the clustering algorithm in a progressive manner to remove noise from spots.
   b. \((N_0/N) \geq \varphi\) in this case, only the pixels in the ‘white’ class will be considered as real signals.

We compare our method with the well-known software GenePix [15] which uses a circle with adaptive radius. Figure 7 show the results for different individual spots.

The 1st row in Figure 6 shows a case with an irregularly shaped spot and noise pixels surrounded by signal pixels. The proposed method successfully removes the noise while GenePix fails. In addition, the adaptive circle method of GenePix is inaccurate, missing some of the signals and including some of the background in it. The second row in Figure 6 shows another contaminated spot, with noise located on the bottom-right of the bounding box. There GenePix and our method both perform well, but GenePix misses some of the signals.

The 3rd row in Figure 6 is an example of a donut-shaped spot with one inner hole. The adaptive circle method takes the signals to be all the pixels inside the outer contour of the donut shape. This includes the inner hole pixels which are darker than the signals. The 4th row shows another donut-shaped spot with signals on both the outer circle and within the central circle. Again the proposed method successfully identifies the signals by removing the darker pixels on the intermediate circle, while GenePix misrepresents the actual signals in that spot. The 5th row in Figure 6 includes a regularly shaped spot with real signals surrounded by non-signal pixels (significantly darker) located on the outer rim of the spot. Our method correctly identifies the central signal spot.

GenePix takes the signals to be all the pixels on and inside the outer rim of that spot, including the darker pixels on the outer rim.

A more global evaluation of these two methods is presented in Section 3 with related discussions.

![Figure 6](image1)

**Figure 6.** Segmentation results for some sample spots. In each row, from left to right – 1st: the original spot; 2nd: the pre-labeled spot with the segment boundary superimposed; 3rd: the spot segmentation result and the segment boundaries; 4th: GenePix result.

![Figure 7](image2)

**Figure 7.** Noise elimination results. (a) Before applying noise elimination method; (b) After applying noise elimination method.

### 3. Experimental results

In order to test the performance of our proposed method in handling uneven background and noise contamination, we select a slide in poor condition and evaluate results by using the measures of recall and precision. The recall is defined as the ratio of the number of relevant items retrieved to the total number of relevant items in the dataset. The precision is defined as the ratio of the number of relevant items retrieved to the total number of irrelevant and relevant items retrieved.

#### 3.1. Background identification and noise removal

In Figure 1, we demonstrate a poor-conditioned slide with uneven background and tons of noise. In this slide, the intensities of background pixels are extremely uneven as the background intensity is low on the top and high on the bottom. We applied our noise elimination method to this slide and Figure 7 shows part of the result for the block No.48. In Figure 7, pixels with intensities greater than 0 are outlined with white colors. Figure 7(a) shows...
the slide before noise elimination. Figure 7(b) demonstrates the result after eliminating the noise and the margin areas. The results show that our proposed method is effective in removing background (including margin areas) and noise.

3.2. Block boundary detection and gridding

Based on our experiments on five different slides (48 blocks each, 240 blocks in total), the proposed block detection method returns 224 blocks. Among them, 224 blocks were correctly identified while 16 blocks were missed. Thus, the recall value and precision value of the proposed methods are 93% and 100%, respectively.

In our experiment to test the performance of automatic gridding, our method correctly generates the grids of 30 rows by 30 columns for almost all the 48 blocks except one (block No.43). In block No.43, an extra row line was generated due to severe noise contamination in that block. It is worth mentioning that for block No.36, the manual gridding of GenePix cannot locate its grid lines correctly. Therefore, the precision value and recall value of our method are 99.97% and 100%, respectively.

Table 1. Results for cleaning contaminated spots

<table>
<thead>
<tr>
<th>Total number of contaminated spots</th>
<th># of spots correctly cleaned</th>
<th>Our method</th>
<th>GenePix</th>
</tr>
</thead>
<tbody>
<tr>
<td>191</td>
<td>172</td>
<td>103</td>
<td></td>
</tr>
<tr>
<td>Success Rate:</td>
<td></td>
<td>90%</td>
<td>54%</td>
</tr>
</tbody>
</table>

3.3. Segmentation of spots

As a final assessment, we carried out a subjective evaluation of whether the proposed spot segmentation method was successfully identifying donut-shaped spots and noise-contaminated spots. Since human eyes are typically better at segmenting spots than computers, we compared the results from our method with those from a subjective evaluation by one of the authors under the supervision of our collaborators from UAB Comprehensive Cancer Center. The slide shown in Figure 1 was examined without prior knowledge of the segmentation result, and 187 donut-shaped spots and 191 noise-contaminated spots were randomly selected as the ground truth. To assess the effectiveness of the method in identifying the above two types of spots, the same observer subjectively examined the segmentation results for the selected spots and gave either positive or negative feedback for each spot. Our method correctly segmented 168 donut-shaped spots out of 187 cases with a success rate of 89.84%. A comparison with GenePix is not appropriate here as the adaptive circle method of GenePix cannot represent donut-shaped spots correctly. Table 1 shows the comparison result of our method and GenePix with regard to their performance on noise removal for contaminated spots. As shown in Table 1, the proposed method outperforms GenePix by a large margin.

4. Conclusions and future work

In this paper, we proposed a method to detect the block boundaries and generate grids for microarray images in a fully automatic manner. The proposed method can handle microarray images with uneven background and severe noise contamination. A simple yet effective method for spot segmentation is also proposed and applied to the output of automatic gridding which is a cleaned image with initial labeling. The experimental results show that the proposed method is robust and outperforms GenePix, a popular microarray image analysis tool. In addition, the proposed algorithm is highly parallelizable and thus part of our future work will be to make use of parallel computing in order to further improve the performance.

5. References