An Automatic and Robust Method for Microarray Image Analysis and the Related Information Retrieval for Microarray Databases

Wei-Bang Chen[1], Chengcui Zhang[1], and Wen-Lin Liu[2]
[1]*Department of Computer and Information Sciences,*
[2]*Department of Management, Marketing, and Industrial Distribution,*
University of Alabama at Birmingham, Birmingham, Alabama 35294, USA
{wbc0522, czhang02, wenlin}@uab.edu

Abstract

DNA microarray is an increasingly important technique that allows biologists to monitor expression levels of thousands of genes in parallel. This technique is widely used in biological research for studying genomic aberrations in cancer and other genetic diseases, and therefore, it has great potential for clinical diagnostics in the future. However, there are still several critical problems with this technology. In this study, we focus on two important issues. The first issue is related to automatic gridding and spot segmentation for microarray images. It has been reported that the quality of spot segmentation significantly influences data precision in the subsequent data analysis. However, nowadays microarray image analysis software still requires users' fine tuning to obtain acceptable results. Another important issue is how to automatically collect related information regarding all genes on the microarray slide for subsequent data analysis and data mining. To relieve researchers from manually correcting image processing results and manually collecting the related information for genes, in this paper, we proposed an automatic and robust method for microarray image analysis and the related information retrieval module which is integrated with the proposed database schema for microarray image data.

1. Introduction

DNA microarray is a promising technology that enables biologists to examine expression levels of thousands of genes concurrently. This powerful tool was introduced in 1999 by Patrick Brown and Vishwanath Iyer to allow biologists to perform many hybridization experiments simultaneously in a single chip [1]. The central dogma of microarray technique is the binding, measurement, and analysis of specimen and its complementary probes [2]. It has been widely used in biological research for studying genomic aberrations in cancer and other genetic diseases and has great potential for clinical diagnostics in the future.

A microarrayer produces a microarray slide by repetitively moving the spotting pins and printing the DNA sample onto the slide with a specific pattern. A typical cDNA microarray slide is shown in Figure 1. The typical layout of a microarray image consists of one or more blocks. Ideally, these blocks are of equal size, and the distance between two blocks is uniform. Each block contains an array of spots which ideally are identical in size and shape with equal distance between them in both vertical and horizontal directions. Each spot contains one specific probe which is a DNA fragment of a gene. For example, the microarray image in Figure 1 is composed of 48 (12×4) blocks, and each block consists of 900 (30×30) spots.

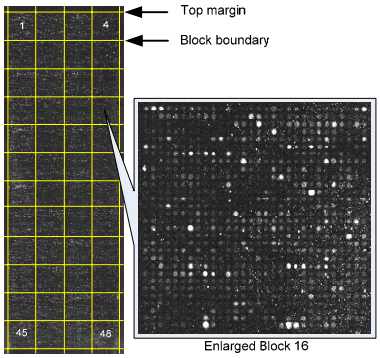


Figure 1. The layout of a typical microarray image

A microarray experiment includes the processes of hybridization, image processing and data analysis. The hybridization is a laboratorial operation which allows specimen binding to its complementary probes, and finally makes spots labeled with different fluorescent signal. After the fluorescent signal is obtained from microarray scanner, analysis software is used to recognize the location of each block and spot on the image, and then to determine the boundary of each spot for estimating the fluorescent intensity of that spot [3]. However, this

1-4244-0832-6/07/$20.00 ©2007 IEEE.

process is not fully automatic and requires intensive manual effort from operators due to the noise and contaminations in microarray images. The fluorescent intensity of a spot represents the expression level of the associated gene. Prior to the data analysis and mining process, we want to store not only the normalized fluorescent intensity of a gene in a database, but also its related meta information such as the known function, the organism, the location on chromosome, etc. for subsequent data analysis and mining.

There exist several critical problems in microarray experiments. The first problem occurs in the image processing stage. In real world situations, microarray images often come with manufacturing defects, such as inner holes, artifacts, and blanks. In addition, the hybridization process may cause scratches, noises, and uneven background due to improper operations. All of these often make the automatic gridding and segmentation results unsatisfactory [4] and require manual effort to correct them. In 2004, Ahmed *et al*. [5] reported that the methods used for microarray image segmentation could significantly influence the data precision in the subsequent data analysis. Since gridding and segmentation are both challenging tasks, they have been typically dealt with separately in the literature [6].

Several methods, such as Markov random field (MRF) [7, 8], template matching and seeded region growing method [9], and the axis projections of image intensities [10], have been reported for automatic gridding. These approaches generally require user provide information such as numbers of rows and columns as mandatory input parameters.

In the meanwhile, several segmentation methods have been proposed in this field. (1) The fixed or adaptive circle segmentation methods use either a fixed radius circle or adjustable radius circle to fit a spot [6]. (2) The clustering-based segmentation method uses *k*-means clustering and partitioning around medoids (PAM) to generate binary partition of the pixels based on the distribution of their intensities [11]. (3) Adaptive shape segmentation uses watershed method or seeded region growing method which relies heavily on the selection of a seed point [6, 12]. (4) Adaptive thresholding method computes a threshold based on the Mann-Whitney (MW) test [13]. (5) Histogram methods use the histogram of a masked area to determine foreground and background [6]. (6) A method based on Markov random field modeling is proposed in [6]. (7) Gradient-based spot segmentation uses morphological analysis [14] for spot segmentation.

The second problem arises when we want to store other related annotation information for each gene together with its gene expression level in the database for subsequent data analysis and mining. Although commercial microarray products provide some annotations, that information may be outdated and there is a need to always keep the most up-to-date information for subsequent data analysis. However, it is not feasible to collect information manually since a typical microarray slide may contain more than hundred-thousand of genes and the related information may reside in different databases. Therefore, there is a need to automate the information retrieval task and to design an efficient database schema for storing microarray image processing results and related information.

In this paper, for microarray image analysis, we proposed a three-step approach which includes (1) background identification and noise removal, (2) fully automatic gridding, and (3) spot segmentation. In the first step, the goal is to preserve all possible pixels in spots (foreground), and eliminate as much the pixels outside the spots (background) as possible. We first apply a local-global thresholding method to distinguish background pixels from the foreground pixels, and then use a voting method based on spatial connectivity to remove most of the noise. In the second step, we first extract those more obvious blocks by identifying significant gaps between blocks, and then use the estimated block size to find those not-so-obvious block boundaries. After that, the bounding box for each group of spatially connected signal pixels i.e. a potential spot is generated, and the automatic gridding can be done by the proposed statistical analysis on the sizes of potential spots. The third step in our approach is spot segmentation. We developed a simple yet effective method based on a progressive two-class clustering scheme. Although more complex spot segmentation methods exist, our spot segmentation method is much time-efficient and has been found effective in our case. We compare our method with widely used commercial software GenePix. The experimental results show the three-step approach is more robust than GenePix.

In this study, we also implement a Perl program to retrieve the name of the probe set which was previously imported from a text file in the database, and use these probe set names to perform an automatic search on public databases like Gene and MeSH, available at NCBI's website. Furthermore, the Perl script parses the query result, extracts necessary information, and then stores the related information into the database according to the proposed database schema as described in Section 2.

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

2. Methods

For each microarray slide, it will generate two images, corresponding to two different fluoresce dyes. 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. The background intensity of the microarray image might be uneven due to poor slide quality or improper operation in the counter stains process. We exemplified the uneven background in Figure 2. In general, there are two approaches to identifying 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 smaller areas and finds a proper threshold for each small area locally. The local approach in general 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 pixels in that region are background pixels. In such cases, the 'real' signal pixels may be treated as noise and thus will be discarded.

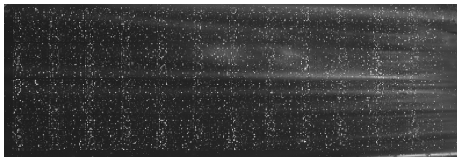


Figure 2. Uneven background of a microarray image

Our proposed approach takes the advantages of both global threshold and local threshold to identify background pixels. To find a proper global threshold, we first divide the entire slide into a set of smaller areas A.

$$A \in \Re^{m \times n} \Leftrightarrow A = (a_{ij}) = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \quad a_{ij} \in \Re$$

where m and n are the dimensions of the small area, and a_{ij} are the pixels within that area. For each small area, we calculate the average intensity of all pixels ($\overline{A}$) in that area and use that value 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.

$$E \in \Re^{r \times c} \Leftrightarrow E = (\overline{A}_{ij}) = \begin{bmatrix} \overline{A}_{11} & \cdots & \overline{A}_{1c} \\ \vdots & & \vdots \\ \overline{A}_{r1} & \cdots & \overline{A}_{rc} \end{bmatrix} \quad \overline{A}_{ij} \in \Re$$

where $r \times c$ is the total number of small areas. In other words, the entire slide has been divided into r rows and c columns of such small areas. The width and the height of each A are set to be smaller than the typical width and height of margins. As shown in Figure 1, the margin areas outside the blocks should contain no spot but background pixels only. Therefore, in each row and column, 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_G and the standard deviation σ_G as follows:

$$H_i = E(i,:) = \begin{bmatrix} \overline{A}_{i1} & \overline{A}_{i2} & \cdots & \overline{A}_{ic} \end{bmatrix}$$

$$V_j = E(:,j) = \begin{bmatrix} \overline{A}_{1j} & \overline{A}_{2j} & \cdots & \overline{A}_{rj} \end{bmatrix}^{\mathrm{T}}$$

$$m_G = \frac{1}{(r+c)} \times \left(\sum_{i=1}^{r} \min(H_i) + \sum_{j=1}^{c} \min(V_j) \right)$$

$$T_G = m_G \pm 3 \times \sigma_G$$

The local approach is similar to the global one. Within each small 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 σ_L are computed similarly to m_G and σ_G. 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_L here is usually below the average intensity of the background. This is especially true for the sample slide shown in Figure 2 in which background intensities increases towards the bottom (right) of the slide. Therefore, we use a relatively higher threshold $m_L + 4 \times \sigma_L$ to eliminate not only the background pixels with low intensity, but also some of the background pixels with low to 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 identified and removed from the foreground.

2.2. Noise reduction

After the majority of the background pixels being removed, the next issue is noise, which usually have intensity levels similar to or greater than that of signals. To reduce noise, we observe that signal pixels usually group together locally to form a round-shaped spot, while noise pixels usually form much smaller local groups containing only a couple of pixels. Therefore, if a pixel has its intensity greater than the threshold of the

background but does not connect well to its neighbors i.e. it does not have a good local spatial connectivity with its neighbors, then it could be a noise pixel. Based on such assumption, we implement the following method to reduce noise.

Our method examines a target pixel in a 5 by 5 lookup window as shown in Figure 3. The center of the window is the target pixel that we want to examine. Each target pixel is surrounded by its 24 neighbors. As mentioned above, since signals usually group together locally to form a round-shaped spot, if a target pixel belongs to a real signal spot, it should be neighbored with more signal pixels within its lookup window. On the other hand, noise pixels will have a higher possibility of being surrounded by background pixels and are spatially isolated as a 'small island' in its lookup window.

In particular, we first label each neighbor pixel in the window as either 1 or 0, indicating if it is a foreground (1) pixel or not (0). In the second step, we check the labels of each pair of pixels that are symmetric to the center target pixel. For example, in Figure 3, P_1 & P_1' and P_2 & P_2' are two such pairs of symmetric pixels in a 5×5 lookup window.

P_1	P_2	P_3	P_4	P_5
P_6	P_7	P_8	P_9	P_{10}
P_{11}	P_{12}	Target	P_{12}'	P_{11}'
P_{10}'	P_9'	P_8'	P_7'	P_6'
P_5'	P_4'	P_3'	P_2'	P_1'

0	0	0	0	1
0	1	1	0	1
0	0	Target	0	0
0	0	0	0	0
0	0	0	0	0

Figure 3. A 5×5 lookup window

If both pixels in a pair are zeros (meaning both are background pixels), then we have one vote for the target pixel being a noise pixel. When the number of noise votes reaches 5, we say the target pixel is a noise. Figure 4 shows two examples. The dark cells indicate background pixels and the white cells represent signal pixels. In Figure 4(A), we have 8 votes (P1 & P1', P2 & P2', P3 & P3', P4 & P4', P6 & P6', P9 & P9', P11 & P11', and P12 & P12') for noise. Since the number of votes for noise is 8, the target pixel is considered to be a noise. Similarly, in Figure 4(B), since the number of votes for noise is 2, the target pixel is considered to be a signal pixel. The threshold value 8 is an empirical value. As the last step, all the identified noise pixels will be removed from the foreground.

P_1	P_2	P_3	P_4	P_5
P_6	P_7	P_8	P_9	P_{10}
P_{11}	P_{12}	Noise	P_{12}'	P_{11}'
P_{10}'	P_9'	P_8'	P_7'	P_6'
P_5'	P_4'	P_3'	P_2'	P_1'

(A) Target pixel with 8 noise votes

P_1	P_2	P_3	P_4	P_5
P_6	P_7	P_8	P_9	P_{10}
P_{11}	P_{12}	Signal	P_{12}'	P_{11}'
P_{10}'	P_9'	P_8'	P_7'	P_6'
P_5'	P_4'	P_3'	P_2'	P_1'

(B) Target pixel with 2 noise votes

Figure 4. Two examples for noise reduction

2.3. Detecting margins and blocks

After eliminating most of the background and noise pixels by the above processes, the slide will have a much cleaner look with higher contrast corresponding to the signal spots. However, before we can start gridding and extracting the spots, we have to detect the boundaries for each block and divide the entire slide into blocks. As shown in Figure 1, a microarray image typically contains more than one block. It is also known that the spots are better aligned within a block than across blocks. It would be more robust to generate grids at the block level rather than at the slide level.

To detect the boundary for each block without knowing the number of blocks in each row/column, we scan the entire slide row by row and column by column at the pixel level. By counting the number of signal pixels in each row and each column, we obtain the frequency chart as shown in Figure 5.

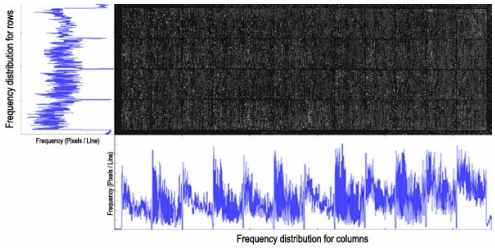


Figure 5. Distribution charts of signal pixels across rows and columns

We can observe from these charts that those rows and columns that really contain spots have high counts of signal pixels. Therefore, we calculate the mean and the standard deviation of each chart and identify all the rows and columns which contain signal pixels less than a threshold *T*, which is calculated as follows:

$$T = \mu - k \times \sigma$$

$$S_T \geq L \times 0.05$$

where μ and σ are the mean and the standard deviation of the number of signal pixels, respectively. L is the number of rows or columns of the image, and S_T is the number of selected rows or columns whose number of contained signal pixels is less than T. k is the coefficient and its actual value is selected in a way that assures that the S_T selected rows or columns constitute more than 5% of the total rows or columns as shown in the above equation. A smaller k will always satisfy this requirement; however, it could also cause the value of T too high to be appropriate.

With our assumption that those background rows and columns should contain signal pixels no more than 5% of the total row/column length (in terms of number of pixels per row/column), we further select the maximum value of k which meets the abovementioned requirement and also makes the value of T less than 5% of the row/column length. In our case, such a k can always be found.

In this way, we can obtain a series of groups of adjacent rows or columns corresponding to gap areas (with no signals) on the slide. With the assumption that a typical block gap should be wider than 5 pixels, we collect the width value for each group of adjacent rows and columns and exclude all those groups with their width values less than 5. By excluding all of such, the remained groups of adjacent rows or columns are what we believe the gaps between blocks, as shown in Figure 6.

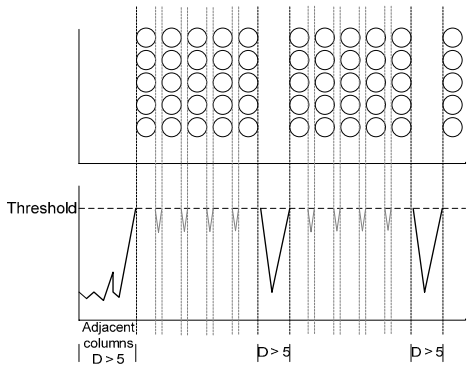


Figure 6. Detect block gaps

For the remained groups of adjacent rows or columns, we then find out the central line for each row/column group as the dividing lines between blocks. With all these identified dividing lines, we can further obtain the dominating distance between two successive dividing lines and use that as the estimated block size. This information is then used to examine whether the blocks are correctly divided and to interpolate missing dividing lines or to remove false positives. Figure 7 shows how to use the estimated block size to discover missing block dividing lines.

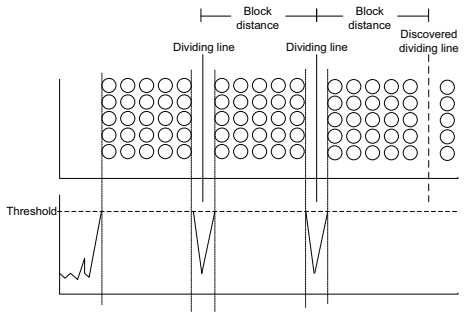


Figure 7. Discover missing dividing lines

2.4. Gridding

With all the blocks on a slide are correctly identified, our next step is to find the grids within each block. We first generate a bounding box for each spot i.e. a group of spatially connected foreground pixels. A spot containing too few pixels will be ignored. Ideally, a bounding box will contain one spot. However, some may contain more than one spot, forming a group of spatially connected spots. In order to automatically generate grids without knowing the number of rows and columns in a block, we implement the following algorithm.

First we notice that the majority of the bounding boxes contain only one spot and can be used to estimate the average height and width of a spot. Therefore, the mode (*Mo*) in the statistic distribution of the bounding boxes' heights and widths is used as the estimate of the typical height and width of a spot. 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 able to represent the average height and width of a spot. Based on this assumption, we then select only those bounding boxes with the height or width within the range of *Mo* ± 3 pixels which allows a little variation in the data. Those selected bounding boxes are considered 'eligible boxes' and can be use to generate vertical and/or horizontal grid lines automatically. Figure 8 shows some examples of those 'eligible boxes'.

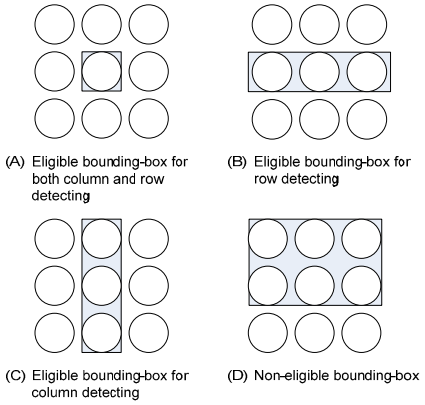


Figure 8. Illustration of eligible bounding boxes

Gridding is the process of identifying the dividing line between each two consecutive rows/columns of spots, such that the position of each spot within the grids can be determined. This is also the so-called 'addressing' process. We first detect the centerline of each row/column by examining the bounding boxes' centroids on that row/column. The centroid of each bounding box contains both the values of its x-coordinate and y-coordinate. Our detection algorithm is based on the observation as illustrated in Figure 9. For all the bounding boxes in a row, if we calculate the distance value (in terms of their x-coordinates) between each two neighboring bounding boxes in the same row (Δx) and that between rows (ΔX), we can see that ΔX >> Δx (see Figure 9).

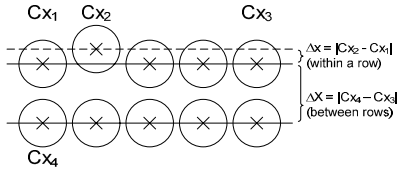


Figure 9. The differences of x-coordinates between rows and within a row

Therefore, we can take the advantage of this pattern to separate the rows and columns in a block. Take row detecting for example, we group those bounding boxes with Δx less than a threshold and assume those boxes are in a row. Then we use the centroids of those boxes to generate a center line for that row of spots (see Figure 9). Once the center line is obtained for each row, the horizontal grid lines are automatically generated by finding the mid-line between each pair of adjacent center lines. The vertical grid lines are generated in a similar way.

2.5. Spot segmentation

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 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. $N_{th}=N$: this case corresponds to the situation where all the pixels pre-labeled as 'foreground' are real signals.
2. $N_{th}<N$: 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. $(N_{th}/N)\leq\varphi$ where φ 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. $(N_{th}/N)>\varphi$: 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 which uses a circle with adaptive radius. Figure 10 shows the results for different individual spots.

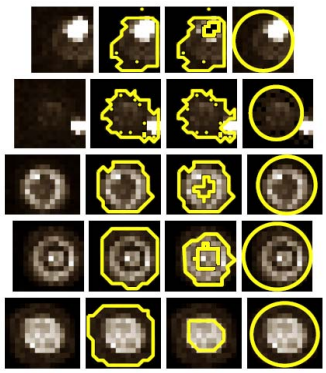


Figure 10. Segmentation results for some sample spots. In each row, from left to right – 1st: the original spot; 2nd: the pre-labeled spot with the segment boundary superimposed; 3rd: the spot segmentation result and the segment boundaries; 4th: GenePix result.

The 1st row in Figure 10 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 10 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 10 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 10 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 comprehensive evaluation of these two methods is presented in Section 3 with related discussions.

2.6. Automatic information retrieval

Once the signals on the image are successfully extracted, we may want to collect related information of all genes and store it in the database for subsequent data analysis and mining. Even though commercial microarray products provide some annotation files, there is still a need for an automatic information retrieval facility which can automatically collect the most up-to-date information for microarray data analysis. For example, it would be infeasible to collect information manually since a typical microarray slide may contain hundred-thousand of genes and the related information may reside in different databases. Therefore, there is a need to automate the information retrieval tasks and design an efficient database schema for storing image processing results and related information.

In response to this need, in this study, we implemented a prototype system of our proposed information retrieval module with Perl. Perl provides many useful modules for developing network applications. Perl also enables us to use regular expression. With this powerful syntax, we can do complex string comparisons, selections, and replacements. Therefore, parsing the query results for obtaining the related information becomes very easy.

The concept of automatic information retrieval is quite simple. When users submit their query via a web search interface, the browser such as IE just simply sends a form with all the requests to the web search server. After processing the query, the server returns query results in HTML format. Thus, we adopt the web user agent class to create an `LWP::UserAgent` object for dispatching web requests with relevant protocol, and the server will return the query results in HTML format and store the content in an `HTTP::Response` object. Following is the sample code for a query with 'ITGB2' as the keyword searched via MeSH database available at NCBI.

```
#-----------------------------------------------------------------
use LWP::UserAgent;

#parameters
$url='http://www.ncbi.nlm.nih.gov/entrez/query.fcgi';
$database='MeSH';
$term='ITGB2';

#construct a hash for storing user inputs
%qryFrm = (db=>$database,term=>$term,);

#create an useragent object
$user_agent = new LWP::UserAgent;

#dispatch the request and receive the returned results
$response = $user_agent->post($url, \%qryFrm);

#retrieve content stored in the response object.
$content = $response->content;
#-----------------------------------------------------------------
```

It is easy to parse the retrieved content with regular expressions. We exemplified how to parse the returned HTML content with Perl as below:

```
#-------------------------------------------------------

if($content=~/<div class="ResultSet">((.|\n)*?)<\/div>/g) {
  #select the result set in the HTML file
  $ResultSet = $1;

  #parse the content
  if($ResultSet=~/<input.*><b>.*<\/b><b>(.*)<\/b>[\w\W]*?<dd>
  (.+?)<br>Year introduced: (\d+)<br>[\w\W]*?<p>Entry Terms:(
  [\w\W]*?)<\/p>/) {

    print "MeSH Term       : $1\n";
    print "Description     : $2\n";
    print "Year introduced : $3\n";
    print "Entry Terms     : $4\n";

    $EntryTermList = $4;
    while ($EntryTermList=~/<li>(.*)<\/li>/g){
      print "                  - $1\n";
    }
  }
}

#---------------------------------------------------------------
```

2.7. Database schema

The schema of the proposed microarray image database consists of the following six tables with their attributes described as follows. Basically, it models almost every aspect about microarray image data and its associated meta-information (e.g. experiment information, probe set, etc.) and organizes it in a hierarchical way, which corresponds to the hierarchical processing of microarray image data (slide level→block level→grid level→spot level). It is worth mentioning that the proposed DB schema is efficient in supporting subsequent analysis such as gene co-expression analysis by providing quick locating of specific genes and their meta-information. In particular, each spot in the database connects with the corresponding information by REF_ID. This will eliminate some data redundancy since many spots in the database represent the same gene. In addition, we store the upper left and bottom right coordinates of the bounding box of each spot for efficiently retrieval of its subimage from the raw image of the microarray slide. This could also save the storage needed for saving individual spot images in the database.

EXP_INFO (Experiment information and record)

Field Name	Type	Size	Description
EXP_ID	Integer	10	Unique ID for an experiment
EXP_DATE	Date		Experiment data and time
EXP_DESC	Char	255	Experiment brief description
EXP_MO	Char	65535	Experiment memo

SLIDE_INFO (Slide information)

Field Name	Type	Size	Description
SLIDE_ID	Integer	10	Unique ID for a slide
SLIDE_MFG	Char	50	Slide manufacture
SLIDE_BAT	Char	30	Slide batch and serial number
SLIDE_IMG	BLOB		Raw Image of slide
SLIDE_PROBE	Char	30	Probe set used on this slide

BLOCK_INFO (Block information)

Field Name	Type	Size	Description
SLIDE_ID	Integer	10	Unique ID for a slide
ORIENT	Boolean	1	Yes: Horizontal / No: Vertical
LINE_SET	Char	255	Separation lines

GRID_INFO (Grid information)

Field Name	Type	Size	Description
BLOCK_ID	Integer	10	Unique ID for a block within a slide
ORIENT	Boolean	1	Yes: Horizontal / No: Vertical
LINE_SET	Char	255	Separation lines
SLIDE_ID	Integer	10	Unique ID for a slide

SPOT_INFO (Spot information)

Field Name	Type	Size	Description
SPOT_ID	Integer	10	Unique ID for a spot
SPOT_X1	Integer	10	x-coordinate of upper left corner
SPOT_Y1	Integer	10	y-coordinate of upper left corner
SPOT_X2	Integer	10	x-coordinate of lower right corner
SPOT_Y2	Integer	10	y-coordinate of lower right corner
CH635_INT	Double	5,3	Intensity of channel 635
CH532_INT	Double	5,3	Intensity of channel 532
PIX_CLASS	Char	255	Pixel index belong to signal
BLOCK_ID	Integer	10	Unique ID for a block within a slide
SLIDE_ID	Integer	10	Unique ID for a slide
REF_ID	Integer	10	Unique ID for a reference

CROSS_REF (Cross reference)

Field Name	Type	Size	Description
REF_ID	Integer	10	Unique ID for a reference
SYMBOL	Char	255	Official symbol of a gene
OFF_NAME	Char	255	Official name of a gene
OTHER_1	Char	255	Official name of a gene
…			

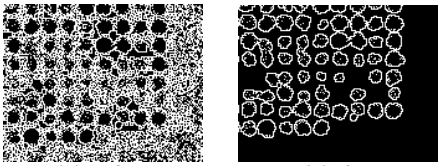
3. Experimental results

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 recognition and noise removal

In Figure 2, 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 11 shows part of the result for the block No.48. In Figure 11, pixels with intensities greater

than 0 are outlined with white colors. Figure 11(a) shows the slide before noise elimination. Figure 11(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.



(a) Before (b)After

Figure 11. Noise reduction result. (a) Before applying the noise reduction method; (b) After applying the noise reduction method.

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 240 blocks. Among them, 240 blocks were correctly identified. Thus, the recall value and precision value of the proposed methods are 100% and 100%, respectively. This experimental results show the robustness of our proposed block boundary detecting method. It is worth mention that our approach mimics human behavior for recognizing both apparent and unapparent block separation lines.

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.

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 2 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.

Table 1. Results for cleaning contaminated spots

Total number of contaminated spots	Number of spots correctly cleaned	
	Proposed method	GenePix
191	172	103
Success Rate:	90%	54%

4. Conclusions and future work

In this paper, we proposed a method which can automatically detect the block boundaries and generate grids for microarray images. Further more, our proposed method also have a good performance in dealing with the microarray slides with uneven background and lots of noise. The later is especially important and is one of the major contributions of this study to the filed of microarray image analysis. We also proposed a simple yet effective method for spot segmentation and applied it to the output result of automatic gridding. The experimental results show that the proposed method is robust and outperforms GenePix, a popular commercial 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.

Furthermore, we implement an information retrieval module to retrieve related information from online public databases like MeSH and store it into a microarray image database by the proposed database schema.

In our future work, we shall focus on the subsequent data analysis and mining of the microarray experiments since this is the ultimate goal of the microarray experiments. Also, we will continue to implement a complete system for the proposed framework with a parallel computing component for improving the performance.

5. Acknowledgment

The work of Chengcui Zhang was supported in part by IBM UIMA Award and SBE-0245090 the UAB ADVANCE program of the Office for the Advancement of Women in Science and Engineering.

6. References

[1] V. R. Iyer, M. B. Eisen, D. T. Ross, G. Schuler, T. Moore, J. C. Lee, J. M. Trent, L. M. Staudt, J. Hudson, Jr., M. S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P. O. Brown, "The transcriptional program in the response of human fibroblasts to serum", *Science*, vol. 283, pp. 83-7, 1999.

[2] U. M. Braga-Neto and E. T. Marques, Jr., "From functional genomics to functional immunomics: new challenges, old problems, big rewards", *PLoS Comput Biol*, vol. 2, pp. e81, 2006.

[3] D. J. Duggan, M. Bittner, Y. Chen, P. Meltzer, and J. M. Trent, "Expression profiling using cDNA microarrays", *Nat Genet*, vol. 21, pp. 10-4, 1999.

[4] Q. Li, C. Fraley, R. E. Bumgarner, K. Y. Yeung, and A. E. Raftery, "Donuts, scratches and blanks: robust model-based segmentation of microarray images", *Bioinformatics*, vol. 21, pp. 2875-82, 2005.

[5] A. A. Ahmed, M. Vias, N. G. Iyer, C. Caldas, and J. D. Brenton, "Microarray segmentation methods significantly influence data precision", *Nucleic Acids Res*, vol. 32, pp. e50, 2004.

[6] O. Demirkaya, M. H. Asyali, and M. M. Shoukri, "Segmentation of cDNA microarray spots using markov random field modeling", *Bioinformatics*, vol. 21, pp. 2994-3000, 2005.

[7] M. Katzer, F. Kummert, and G. Sagerer, "Methods for automatic microarray image segmentation", *NanoBioscience, IEEE Transactions on*, vol. 2, pp. 202-214, 2003.

[8] M. Katzer, F. Kummert, and G. Sagerer, "A Markov Random Field model of microarray gridding", in *Proceedings of the 2003 ACM symposium on Applied computing*. Melbourne, Florida: ACM Press, 2003.

[9] Y. H. Yang, S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed, "Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation", *Nucleic Acids Res*, vol. 30, pp. e15, 2002.

[10] A. N. Jain, T. A. Tokuyasu, A. M. Snijders, R. Segraves, D. G. Albertson, and D. Pinkel, "Fully automatic quantification of microarray image data", *Genome Res*, vol. 12, pp. 325-32, 2002.

[11] H. Y. Jung and H. G. Cho, "An automatic block and spot indexing with k-nearest neighbors graph for microarray image analysis", *Bioinformatics*, vol. 18 Suppl 2, pp. S141-51, 2002.

[12] Y. H. Yang, M. J. Buckley, and T. P. Speed, "Analysis of cDNA microarray images", *Brief Bioinform*, vol. 2, pp. 341-9, 2001.

[13] Y. Chen, E. R. Dougherty, and M. L. Bittner, "Ratio-based decisions and the quantitative analysis of cDNA microarray images", *Journal of Biomedical Optics*, vol. 2, pp. 364-374, 1997.

[14] J. Buhler, T. Ideker, and D. Haynor, "Dapple: improved techniques for finding spots on DNA microarrays", University of Washington, Seattle, WA UW CSE Technical report UWTR 200-08-05, 2000.

[15] N. Otsu, "A threshold selection method from gray-level histogram", *IEEE Trans. Systems, Man, and Cybernetics*, vol. 9, pp. 62-66, 1979.