GRIDLINE: AUTOMATIC GRID ALIGNMENT IN DNA MICROARRAY SCANS

Peter Bajcsy

National Center for Supercomputing Applications

605 East Springfield Avenue, Champaign, IL 61820

Phone: 217-265-5387, Fax: 217-244-7396, Email: pbajcsy@ncsa.uiuc.edu

ABSTRACT

We present a new automatic grid alignment algorithm for detecting two-dimensional (2D) arrays of spots in DNA microarray images. Our motivation for this work is the lack of automation in high-throughput microarray data analysis that leads to (a) spatial inaccuracy of located spots and hence inaccuracy of extracted information from a spot, and (b) inconsistency of extracted features due to manual selection of grid alignment parameters. The proposed grid alignment algorithm is novel in the sense that (1) it can detect irregularly row- and column-spaced spots in a 2D array, (2) it is independent of spot color and size, (3) it is general to localize a grid of other primitive shapes than the spot shapes, (4) it can perform grid alignment on any number of input channels, (5) it reduces the number of free parameters to minimum by data driven optimization of most algorithmic parameters and (6) it has a built-in speed versus accuracy tradeoff mechanism to accommodate user's requirements on performance time and accuracy of the results. The developed algorithm also automatically identifies multiple blocks of 2D arrays, as it is the case in microarray images, and compensates for grid rotations in addition to grid translations.

Key words: DNA microarray, spot alignment, image analysis, simulation, and quality control.

EDICS: 2-ANAL

1 INTRODUCTION

In general, the objective of any microarray data analysis is to draw biologically meaningful conclusions [1], [20]. There are many microarray analysis steps before a conclusion is made. A grid alignment (also known as addressing or gridding [23]) is one of the processing steps in microarray image analysis that registers a set of unevenly spaced, parallel and perpendicular lines (a template) with the image content representing a two-dimensional (2D) array of spots. The registration objective of the grid alignment step is to find all template descriptors, such as, line coordinates and their orientations, so that pairs of perpendicular lines intersect at the locations of a 2D array of spots in a microarray scan. Furthermore, this step has to identify any number of distinct grids of spots in one image. We denote a 2D array of spots as a grid of spots, and one array of spots among multiple 2D arrays as a block or a sub-array of spots [18]. In order to automate microarray analysis, the grid alignment step is needed before any spot segmentation and spot information extraction take place.

Without providing too much background on microarray technology (see [1] Chapter 12, [2], [3], [4]), microarray images are generated by scanners using confocal laser microscopes that scan a microarray slide with several blocks of 2D arrays. The goal of microarray image analysis is to extract absolute or relative intensity values from each spot that represent gene expression levels or features. Biological conclusions are drawn based on the results from data mining and statistical analysis of all extracted features. We will describe in a great detail the spot localization process and some of its challenges.

Before designing any automatic spot localization algorithm, one should consider a microarray slide preparation as a source of variations in spot locations [14], [16], [23]. For example, if a spotting machine with several dipping pins prints multiple 2D arrays of spots, then the dipping pins might bend over time and cause irregularity in a 2D arrangement of the printed spots [14]. Similarly, any rotational offset of a slide or dipping pins will cause a rotated 2D grid in a microarray image with respect to the image edge.

Based on the cDNA labeling type used during microarray slide preparation (hybridization), one can obtain, for instance, single-, double- or multi-fluorescent images. Most microarray data contain double-fluorescent images from scanners that operate at two wavelengths, e.g., 532nm (red) and 632nm (green) wavelengths forming two channels shown in Figure 1 left. In general, microarray image data can consist of any number channels.

Furthermore, while the fluorescent labeling type leads to microarray images with dark background and bright spots (signal), other labeling types with or without radio-isotopic labels lead to images with bright background and dark spots (see Figure 1 right). A slide material introduces another color variation, for example, coated glass slides or nylon membrane or 2D gel materials. As more novel techniques will be developed, there will be no guarantee of fixed background and foreground colors in all microarray scans. It is also noticeable in Figure 1 that the spot size varies and many spots are missing from a 2D array (or low signal is detected). Current microarray slides, as well as future miniaturized microarray slides, require accommodating these variations in a design of automatic grid alignment methods.

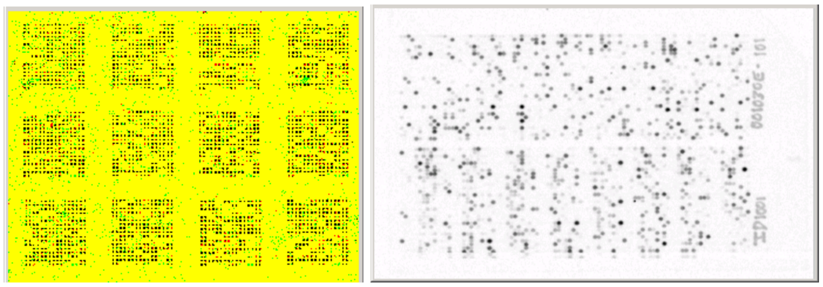


Figure 1: Examples of microarray images with double-fluorescent (left) and radioactive (right) dyes. The image on left contains two channels presented in red and green enhanced pseudo-color.

Another issue to mention is the shape of a primitive in microarray 2D arrays. Although the majority of current microarray imagery is produced with circular spots as shape primitives, we have encountered references to other primitive shapes, e.g., lines or rectangles (see the

CLONDIAG chip [9] or Affymetrix chips [5]). We believe and foresee that other primitive shapes than a spot shape will be used in microarray technology in future and the need for shape independent grid alignment will arise immediately.

We would also like to introduce the issue of grid alignment repeatability. As many other microarray analysis steps, the grid alignment step is desired to be "parameter free" so that the same algorithm can be applied repeatedly without any bias with respect to a user's parameter selection. Thus, any manual positioning of a grid template is not only tedious and time-consuming but also undesirable since the alignment step cannot then be repeated easily. The same logic applies to algorithms with many free parameters that are set by a user.

Lastly, one is concerned with processing an exponentially growing amount of microarray data in future. From the grid alignment algorithm viewpoint, finding a grid is always a tradeoff between computational resources (memory and speed/time) and alignment accuracy given a large number of microarray images. While this issue might be resolved without any accuracy loss by using either supercomputers or distributed parallel computing with grid-based technology [15], [21], it might be still beneficial to build into a grid alignment algorithm such a tradeoff in a short term.

In a nutshell, the problem formulation of an ideal grid alignment algorithm should include the following design requirements. First, a grid alignment algorithm should find irregularly row- and column-spaced 2D arrays with translational and rotational offsets. Second, it should perform alignment on images with any number of input channels. Next, it should be color and spot size independent and independent of any chosen primitive shape. Lastly, it should be parameter "free" and should accommodate speed versus accuracy tradeoffs. We will present our proposed grid alignment method that strives to meet the aforementioned requirements.

The proposed algorithm called GridLine [6] is built as a tool for aligning regular or irregular grid lines over grayscale or color (1 to N bands) DNA microarray images that contain a 2D array of spots with varying radii and deviating spot's locations from perfect grid placements.

The GridLine algorithm is data driven and its goal is to find a set of mutually perpendicular lines that intersect at a center of each grid cell. A grid cell is defined as an area enclosing one spot and is part of a 2D array of spots. From a processing standpoint, the algorithm assumes that sought lines intersect a large number of high contrast and low contrast areas in contrary to the background that is assumed to be homogeneous in color with some superimposed additive noise. In addition, line directions are not constrained and therefore any translated and rotated 2D array of spots can be detected. Furthermore, multiple distinct 2D sub-arrays (grids) can be found by partitioning an input image into sub-images and aligning each distinct grid separately. If the speed is an important issue then it is addressed by down-sampling input images. By running the GridLine algorithm on a smaller sized image (downsampled image), the speed could be improved with some sacrifice of alignment accuracy. The GridLine tool is a part of the microarray image analysis tool set described in [6].

2 PREVIOUS WORK

In the past, the problem of grid alignment has been addressed in two ways. First, the problem was simplified by using a very accurate technology, for example, in the case of Affymetrix chips [5]. Although the problem was simplified and hence the grid alignment became more accurate, the Affymetrix technology has been much more expensive than the technology with coated glass slides and thus the need for solving the grid alignment problem has remained.

Second, the alignment problem was tackled with template-based and data-driven approaches. The template-based approach is the most prevalent in the previous literature and existing software packages, e.g., GenePix Pro by Axon Instruments [7], ScanAlyze [12] or GridOnArray by Scanalytics [8]. To our knowledge, the data-driven approach has been based on statistical analysis of 1D image projections [13], [17], [18] or used as part of image segmentation algorithms [19], [24]. While most of the currently available software packages enable manual template matching [7], [12], [14] by adjusting spot size, spot spacing and grid location, some software products already incorporate an automatic refinement search for a grid location given size and spacing of spots [7], [10]. Nonetheless, irregular grids cannot be found with template-based approaches unless the template is manually adjusted to fit pre-defined distortions [8]. The data-driven approaches are capable of finding irregular grids but are prone to misalignment due to spurious or missing spots and are also dependent on many parameters.

Our proposed grid alignment is data-driven and uses statistical analysis of 1D projections of directional edge feature images. In comparison with other data-driven methods, our proposed method is not based on any segmentation optimization [19] and it is different from the methods described in [17], [18] by analyzing 1D projections of directional edge feature images as opposed to original intensity image. This difference makes the algorithm color independent since the edge features are color free. In addition, some parameters for statistical analysis are optimized automatically and it is our goal to design a completely parameter "free" algorithm in future.

3 GRID ALIGNMENT METHOD

The proposed grid alignment method is executed in several steps as illustrated in Figure 2. The method is based on detecting intensity discontinuities at all spot locations of a 2D array and detecting homogeneous intensities in the background area of the spots. Thus, the underlying model of a microarray image assumes a homogeneous background region and heterogeneous foreground regions (spots). We present a data flow diagram in Figure 2 instead of a pseudo code and explain each module of the data flow next.

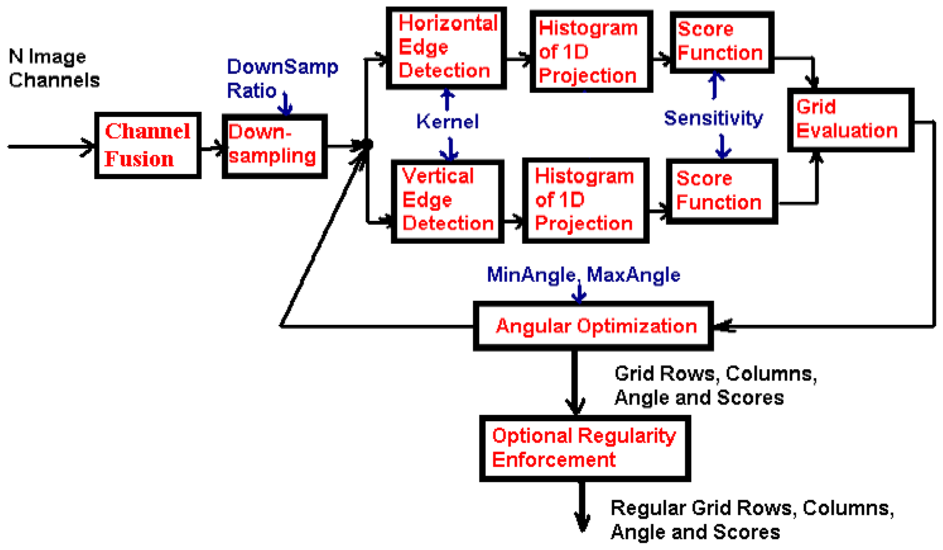
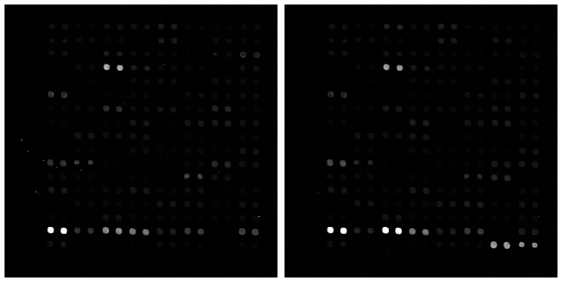

N Image Channels
Channel Fusion
DownSamp Ratio
Down-sampling
Horizontal Edge Detection
Kernel
Vertical Edge Detection
Histogram of 1D Projection
Histogram of 1D Projection
Score Function
Sensitivity
Score Function
Grid Evaluation
MinAngle, MaxAngle
Angular Optimization
Grid Rows, Columns, Angle and Scores
Optional Regularity Enforcement
Regular Grid Rows, Columns, Angle and Scores

Figure 2: Overall schema of the grid alignment algorithm.

3.1 2D Array Processing

First, in order to accommodate an arbitrary number of channels, all image channels are fused into one image by performing a logic OR operation (see "Channel Fusion" module in Figure 2). It is apparent that one would like to detect one common grid for all channels. However, it is very often that a grid cell would not contain any intensity variation indicating a spot in some input image channels. The fusion of all channels with logic Boolean OR operator will propagate foreground and background intensity variations into the grid alignment algorithm and increase its robustness assuming that there is little spurious variation in the background. Furthermore, fusing all channels reduces multi-channel computation and avoids the problem of merging multiple grids detected per each channel.



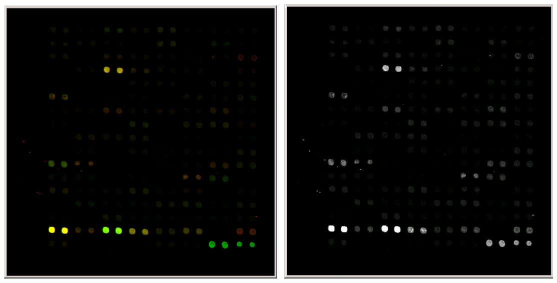


Figure 3: Microarray images of red (upper left) and green (upper right) channels that can be visualized in pseudo-color by combining channels (lower left) and are fused by Boolean OR function before processing (lower right).

The second module denoted as "Down-sampling" addresses the issue of speed versus alignment accuracy. It is well known that the speed of most image-processing algorithm is linearly proportional to the number of pixels since every pixel has to be accessed at least once and processed in some way. Nonetheless, a grid alignment algorithm should detect 2D array of spots regardless of the spot's radii. If two microarray images of the same size would contain NxM spots of radii R1 (image 1) and R2 (image 2), such that R1<R2, then the alignment of image 2 with spots of radius R2 could be performed faster by sub-sampling without any loss of accuracy with respect to the alignment performed on image 1. It follows that the tradeoff of speed (or computational requirements) versus grid alignment accuracy is also a function of spot size. In practice, down-sampling (or local averaging) is preferred instead of sub-sampling in order to preserve local spot information that could be completely eliminated by sub-sampling.

The two branches in Figure 2 with the three modules denoted as "Edge Detection", Histogram of 1D Projection" and "Score Function" show a separate treatment of horizontal and vertical grid lines. The reason for having a separate detection of horizontal and vertical grid lines is that many grids do not have evenly spaced rows and columns. The objective of finding horizontal and vertical grid lines is to identify all grid line intersections defining a spot location. The set of mutually perpendicular grid lines also defines grid cells that form a rectangular around each spot. The functionality of the three modules can be described as (a) feature formation ("Edge

Detection"), (b) statistical analysis of features ("Histogram of 1D Projection") and (c) metric-based ranking of line candidates ("Score Function"). The feature formation step is performed by using a differential gradient operator [22, Chapter 5.8] defined in Equation 1 for vertical and horizontal directions, where Kernel is the spatial extend of directional edge detection and I(i, j) is the image intensity value at image location (row, column) = (i, j).

$$Edge^{vertical}(row,col) = \sum_{\text{i=row}-\text{kernel}}^{\text{row}} \sum_{\text{j=col-kernel}}^{\text{j=col+kernel}} I(i,j) - \sum_{\text{i=row}}^{\text{row+kernel}} \sum_{\text{j=col-kernel}}^{\text{j=col+kernel}} I(i,j)$$

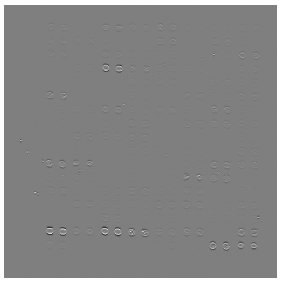
$$Edge^{horizontal}(row,col) = \sum_{\text{j=col}-\text{kernel}}^{\text{col}} \sum_{\text{i=row-kernel}}^{\text{i=row+kernel}} I(i,j) - \sum_{\text{j=col}}^{\text{col+kernel}} \sum_{\text{i=row-kernel}}^{\text{i=row+kernel}} I(i,j)$$

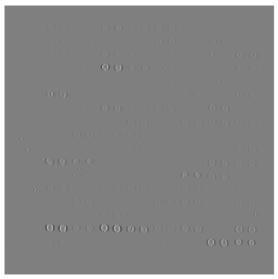
Equation 1: Directional edge definitions.

After each directional edge detector has been applied (see Figure 4), edge feature images are projected into 1D to measure frequency of edges (1D histogram) along each direction. A score for each row and column (denoted Line) is computed from the two histograms according to the formula in Equation 2. The Sensitivity parameter determines what contrast values are considered to be due to signal as opposed to due to background noise variation. The score computation generates two curves of score as a function of row or column shown in Figure 4.

$$Score(Line) = \sum_{\|i\|>Sensitivity} HistValue(Line,i)$$

Equation 2: Score definition





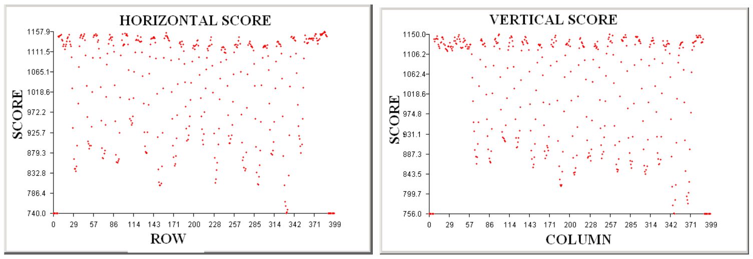


Figure 4: Feature images after horizontal (upper left) and vertical (upper right) edge detection and the formed corresponding score functions for detecting rows (lower left) and columns (lower right).

The final set of mutually perpendicular lines is reported from the module "Grid Evaluation" and it is based on (a) the minimum score parameter and (b) the number of searched lines. For example, if the minimum score parameter is equal to zero then the resulting number of searched lines will include the top ranked lines. If there are fewer lines above the minimum score parameter than the number of searched lines then the result will contain only lines that meet the minimum score requirement.

The grid alignment described in this Section so far has assumed that there is no grid rotation involved. Despite a well-controlled environment during a microarray image formation, an undesirable grid rotation occurs in practice. The problem of grid rotation is addressed by searching the space of expected rotational angles. The angular rotation becomes a grid variable that has to be found by conducting a grid alignment under a set of possible rotations (see module "Angular Optimization"). A user specifies the minimum and maximum angular rotation to narrow the search space since in general it can be any value between -45 and 45 degrees. An initial angular estimate can be made by analyzing four edges of a 2D array [17]. The overall score of each grid is computed as a sum of all line scores in Equation 2 and it is used for selecting the optimal grid rotation (Angle) by maximizing the TotalScore value.

$$TotalScore(Angle) = \sum_{i=0}^{N2FindRow} Score(Row_i, Angle) + \sum_{j=0}^{N2FindCol} Score(Col_j, Angle)$$

Equation 3: The overall grid score used for optimizing grid rotation.

In practice, there might be a strict regularity requirement on a grid. Any imposed regularity requirement can be performed by using the "Optional Regularity Enforcement" module in Figure 2. There are four types of grid that one can seek; irregular, regular rows, regular columns and square grid. The problem of imposing a regularity of an irregular set of lines (unevenly spaced lines) is resolved by computing a histogram of distances between adjacent lines and selecting the most frequent distance as the most likely correct line spacing. The final regular set of lines (evenly spaced lines) is determined with respect to the line with the highest score since the score indicates the most confidence in its location. A square grid is obtained by imposing the regularity requirement on both horizontal and vertical sets of lines.

3.2 Parameter Optimization

We should also mention that a parameter optimization is an integral part of the grid alignment algorithm. The basic set of parameters for running the GridLine algorithm consists of the following input variables: N2FindRow or N2FindCol is the number of expected rows or columns in each grid, MinAngle and MaxAngle define the range of expected grid rotations in degrees, DownSamp represents the down-sampling ratio if it is desirable to reduce computational requirements, Accept is the minimum score value from the range [0,100] for reporting a valid grid line, Sensitivity is the threshold value for separating background noise from signal and Kernel defines the spatial extend of directional edge detection.

As in any other algorithm, a parameter optimization is accomplished by searching a space of parameters and selecting the "best" set of parameters given an evaluation metric. While the benefit of parameter optimization is a fully automated grid alignment tool, the drawback of optimization is the need for more computation and hence slower execution speed. In our case, the grid rotation parameter is a parameter that has to be searched in the entire space of grid rotations unless a user specifies smaller by setting the values of MinAngle and MaxAngle. Similarly, the optimal values of Kernel and DownSamp parameters can be estimated by knowing grid characteristics, such as a spot size and spot spacing. Had the grid characteristics been given, Kernel parameter would be set to a smaller value than the edge-to-edge spot distance and DownSamp to a value that preserves the spot size and spot spacing for a robust grid detection. To achieve fully automatic algorithm, one has to optimize grid rotation and these two parameters simultaneously. The current implementation optimizes the grid rotation, Kernel and DownSamp parameters automatically for a range of user specified values.

However, it is possible to estimate other parameters, for instance, the Sensitivity parameter, without an exhaustive search by analyzing the global distribution of an original image. If a user entered an invalid value for the Sensitivity parameter then the algorithm would compute a 1D image histogram and determine the value for Sensitivity as a percentage of the distribution spread.

From the list of basic parameters, the last three parameters to optimize are N2FindRow, N2FindCol and Accept. These parameters can be optimized in an unsupervised way by detecting a sharp discontinuity in the score function. Without specifying the pair of N2FindRow and N2FindCol, or the value of Accept, or both, the output of data-driven analysis is a rank-ordered list of lines. A sharp discontinuity in the rank-ordered list of lines will determine (a) the values of N2FindRow and N2FindCol for each line set as the number of lines before the sharp discontinuity and (b) the value of Accept as the minimum score of the selected set of line in (a).

3.3 Processing Multiple 2D Arrays

It is very common in the case of DNA microarray image analysis that multiple distinct 2D sub-arrays of spots (grids) are present in one microarray image. These distinct grids are also arranged in a 2D array format thus the number of expected distinct grids can be defined by the number of grids along horizontal (row) and vertical (column) axes. The corresponding input parameters are denoted as N2FindGridRow and N2FindGridCol.

It is possible to search for multiple distinct 2D arrays of spots (grids) in one image with the proposed grid alignment algorithm. The goal of this part of the algorithm is to partition the original image into sub-areas containing individual grids. Due to the nature of most frequently occurring microarray images, it is performed by dividing the original image into rectangular sub-areas based on N2FindGridRow and N2FindGridCol parameters and process each sub-area separately.

If N2FindGridRow and N2FindGridCol are not available as input parameters then there is a need to optimize these input parameters as well. One way how to approach this problem is by searching for all irregular lines in the entire image and then analyzing the spacing of all found mutually perpendicular grid lines. Every large discontinuity in the line spacing will indicate the end of one and beginning of another block (2D arrays of spots). An example result is shown in Figure 5.

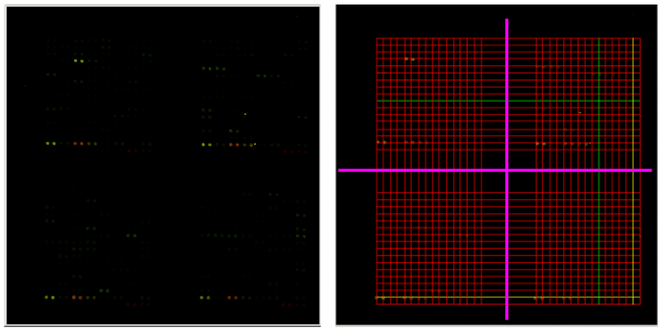


Figure 5: An example result of processing the original image (left) with the proposed algorithm and analyzing discontinuities in line spacing (right) to partition the original image into sub-images containing one sub-array per sub-image.

Another approach to this optimization problem is to low pass (filter) the original image such that all 2D arrays of spots become primitives of a new grid formed by blocks. The filtered image can be analyzed the same way as any image with a single grid as it is shown in Figure 6 left. If most of the sub-array spots are much brighter than the background intensities then each original sub-array becomes a rectangular primitive of a newly created 2D array in the filtered image (see Figure 6 right). The analysis of the created image with one 2D array will determine N2FindGridRow and N2FindGridCol parameters. However, this approach is less robust than the previous one because of missing spots. Due to missing spots, a newly formed grid will contain primitives with holes of background color that cause a location ambiguity of primitives.

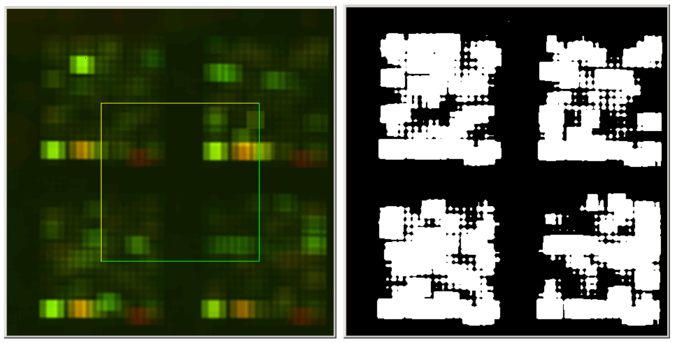


Figure 6: The alignment result (left) obtained by filtering the image in Figure 5 (left) and searching for 2x2 grid lines with the proposed algorithm. A thresholded image (right) of the filtered image (left) demonstrates shape irregularity of primitives due to missing spots.

In the later approach, one would immediately raise a question of grid shape primitives that has changed from spot shape (original image) to rectangular shape (filtered image). Fortunately, the proposed grid alignment method is capable of identifying grids with variable shape of grid primitives. It turns out that although the shape of grid primitives is important for a grid alignment task, the horizontal and vertical edge features used by the proposed method can model an arbitrary shape of the underlying grid primitive as long as the primitives are sufficiently separated. This capability of the proposed method can be utilized not only for finding the number of blocks in one image but also for other types of microarray images using other than spot shape primitives, for example, a rectangular shape used in CLONDIAG chips [9].

4 EXPERIMENTAL RESULTS

This section demonstrates several capabilities of the proposed grid alignment algorithm and describes our robustness and performance evaluations of the proposed method. We conducted all experiments with measured microarray images consisting of two channels and synthetic microarray images derived from measured images. The set of synthetic images was generated with multiple variations in spot size, color, shape and amount of intensity blur, as well as, with variations in 2D array arrangement, such as, rotated arrays, downsampled arrays or arrays with missing spots. Examples of measured microarray images are shown in Figure 1, Figure 3 and Figure 5. Synthetic images are used throughout this section.

4.1 Capabilities of Alignment Method

First, we investigated the impact of a number of input channels on alignment accuracy. We ran the grid alignment algorithm on each channel separately and then on a combined image with the logic Boolean OR operator. To illustrate spatial misalignment of all three resulting grids, we generated a mask with spots (radius = 5 pixels) at each grid cell for each grid result and combined the three generated masks into one color image (red contains the mask obtained from channel 1, green contains the mask obtained from channel 2 and blue contains the mask obtained from both channels at the same time). The color image with three masks and the difference between grid masks are shown in Figure 7. We evaluated quantitatively the total misalignment according to the formula in Equation 4

$$E_{Misalignment} = \frac{Misaligned\ Pixel\ Count}{Total\ Signal\ Area} * 100$$

Equation 4: Total misalignment error.

and presented the experimental values in Table 1. The MisalignedPixelCount value is equal to a half of the white pixels shown in Figure 7 (middle and right) since any misalignment of estimated versus original spots has signal-to-background and background-to-signal components that should be counted only once. This experiment demonstrates measured grid alignment errors as a function of the number of processed channels. Unfortunately, a data set with more microarray channels was not available for this experiment.

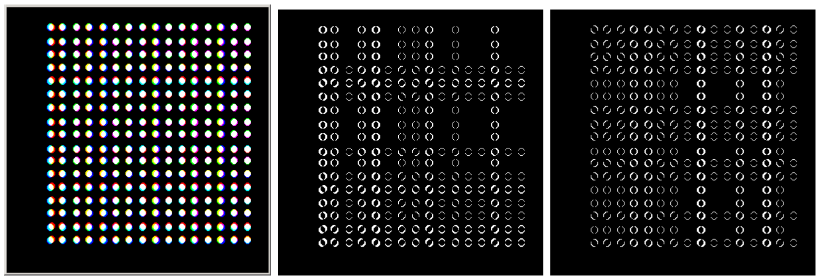


Figure 7: A color image showing three grid masks obtained from channel 1 (red), channel 2 (green) and both channels at the same time (blue) on the left side. The spatial misalignment between the grids obtained from both channels and (a) channel 1 (middle) or (b) channel 2 (right).

Table 1: Misalignment error due to processing channels separately of together.

Misalignment	Channel 1	Channel 2	Channel 1 & 2
Channel 1	0	9.3%	10.6%
Channel 2	9.3%	0	9.8%
Channel 1 & 2	10.6%	9.8%	0

Second, we ran the grid alignment method on both channels with and without regularity requirements imposed on both grid rows and grid columns. The results are presented in Figure 8. This experiment demonstrates the disadvantage of template-based methods in comparison with data-driven methods. The better alignment in Figure 8 (left) than in Figure 8 (middle) is perceptually apparent and the total misalignment computed according to Equation 4 represents 15.5%.

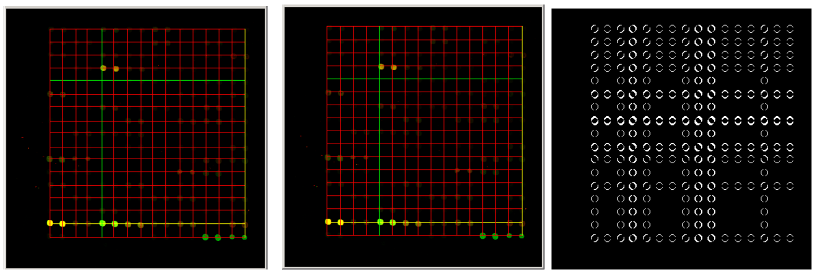


Figure 8: Grid alignment results without (left) and with (middle) regularity requirements imposed on both rows and columns, and the difference between the corresponding two grid masks (right).

Next, we have conducted experiments to demonstrate an invariance of the proposed grid alignment algorithm to spot color and grid primitive shape. The invariance of the proposed algorithm to spot colors comes from the fact that the horizontal and vertical edge features are color independent. Figure 9 shows that the alignment results are identical for images with two simulated spot colors. Observing the grid line colors can lead to a quick visual comparison of the results. The yellow lines in each direction denote the highest confidence lines (maximum score) and the green lines denote the lowest confidence lines (minimum score above Accept). Any other line is shown as red.

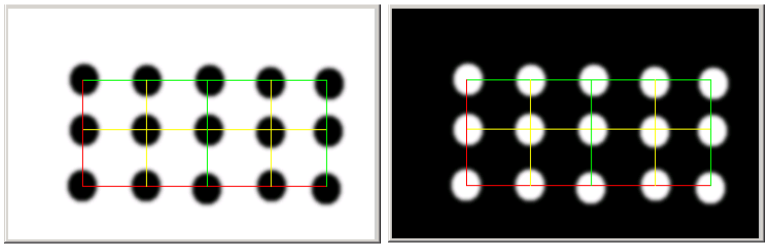


Figure 9: Black (left) and white (right) spot colors do not change the result of a grid alignment.

The invariance to grid primitive shape is demonstrated with two examples of 2D arrays in Figure 10. These examples of grid shape primitives other than spots are composed of blurred squares and triangles where the triangles vary in their orientation in addition to irregular grid locations. The overlaid line results in Figure 10 illustrate an additional capability of the proposed algorithm.

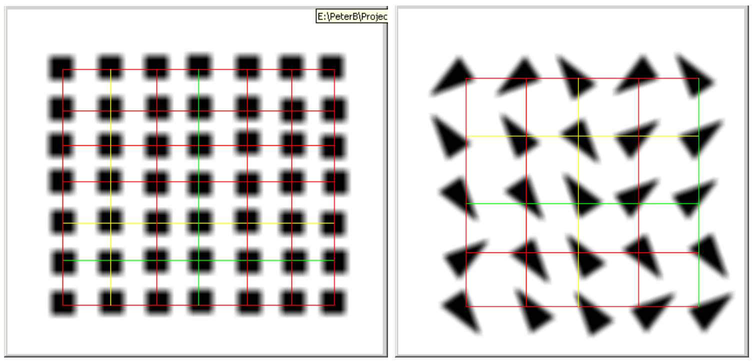


Figure 10: Grid alignment results for square (left) and triangular (right) grid primitives.

Additional capabilities were tested using multiple sub-arrays and rotated 2D arrays. Results obtained from measured and synthetic images are shown in Figure 11 and Figure 12. Any image with multiple sub-arrays was partitioned based on N2FindGridRow and N2FindGridCol input parameters and then optimized with respect to the parameters described in Section 3.2. Measured image in Figure 11 was obtained from a publicly accessible web site at http://stat-www.berkeley.edu/users/terry/zarray/Html/begin.html and the grid alignment results show the capability of processing multiple sub-arrays in one image.

Synthetic images in Figure 12 were used to verify the angular optimization functionality of the proposed algorithm. The verification test consisted of comparisons of correct angular values for single and multiple grids with estimated angular values given a range of acceptable angles (minimum and maximum angular values). We did not detect any angular errors for synthetic images. Nonetheless, one should be aware that there is some amount of spatial alignment inaccuracy in the results for rotated arrays in comparison with the arrays that are not rotated. This inaccuracy comes from rounding pixel locations during any rotation and hence offsetting spot locations.

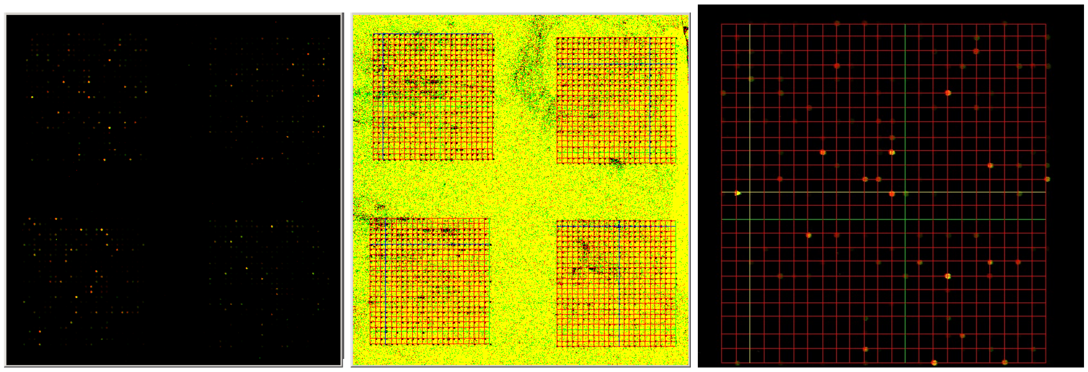


Figure 11: Measured microarray image with four sub-arrays (left) and the result of grid alignment (middle) overlaid on the color enhanced measured image to show the grid alignment. High-resolution view of the top left grid and its grid alignment result (right) demonstrate perceptual quality of the alignment.

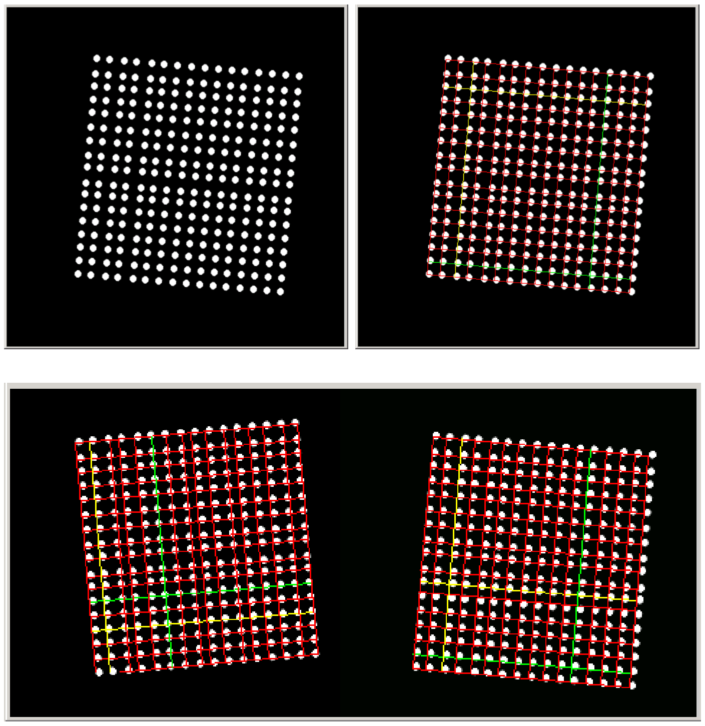


Figure 12: An example of a synthetic image with a rotated 2D array (upper left) by 5 degrees clockwise and the results of alignment for the case of one array (upper right) or multiple sub-arrays (bottom).

4.2 Evaluations of Alignment Method

As in many experimental studies, one would like to evaluate robustness and performance of the proposed method. Although there is an abundance of microarray images, there has not been any established data set to perform robustness evaluations. We have approached this problem by simulating microarray images (similar to the work of Balagurunathan *et al.,*[25]) with some variations observed in measured imagery. In this section, we simulated (a) varying spot size (radius), (b) blurred spots, (c) downsampled spots and (d) missing spots with the goal to test the automatic grid alignment capability[1].

A set of images with 2D arrays of spots was generated by detecting a grid in one of the measured microarray images (see Figure 3) and forming a mask image with estimated spot locations. The radii of spots in a series of mask images varied from 2 to 9 pixels inside of a grid cell with the dimensions along rows from [17, 22] and along columns from [18, 24]. These mask images simulate spatial variations found in measured microarray images but do not simulate intensity variations other than signal and background values. Four simulated images with radii equal to 2, 5, 9 and 12 are shown in Figure 13 and their corresponding alignment results are presented in Figure 14. We compared grid lines reported for every simulated image and there was no misalignment error as long as the spots were contained inside of each grid cell ($R<8$). Once the neighboring grid cells contained overlapping spots, we have observed alignment inaccuracy. An overlap less than about 28% of grid dimensions leads to a small inaccuracy, for instance, 0.5 pixel line offset for the image with spot radius equal to 9 pixels. However, any larger overlap will cause the algorithm to fail because the background regions in the corners of grid cells will be considered as a 2D array while the signal area will be viewed as a homogeneous background. This is demonstrated in Figure 13 and Figure 14 for spots with their radius equal to 12.

[1] We are not testing any spot segmentation, spot quality or spot information extraction functionality.

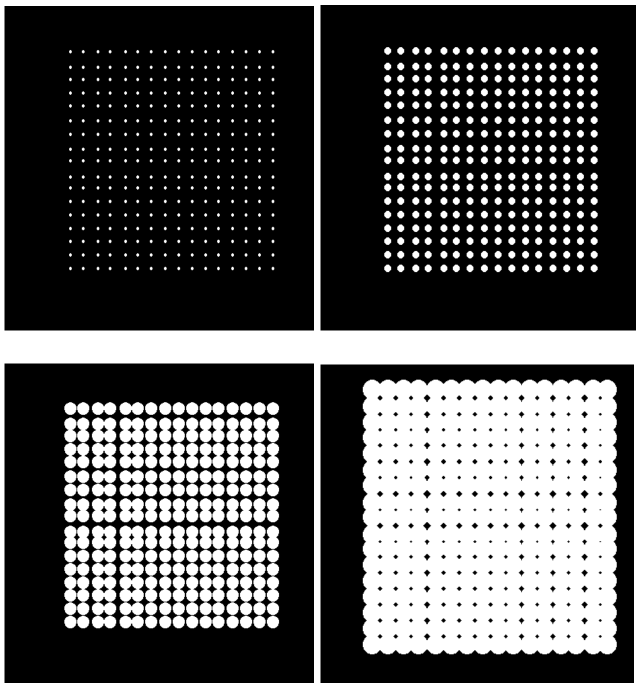


Figure 13: Simulated 2D arrays of spots with varying spot radii, R=2, 5, 9 and 12.

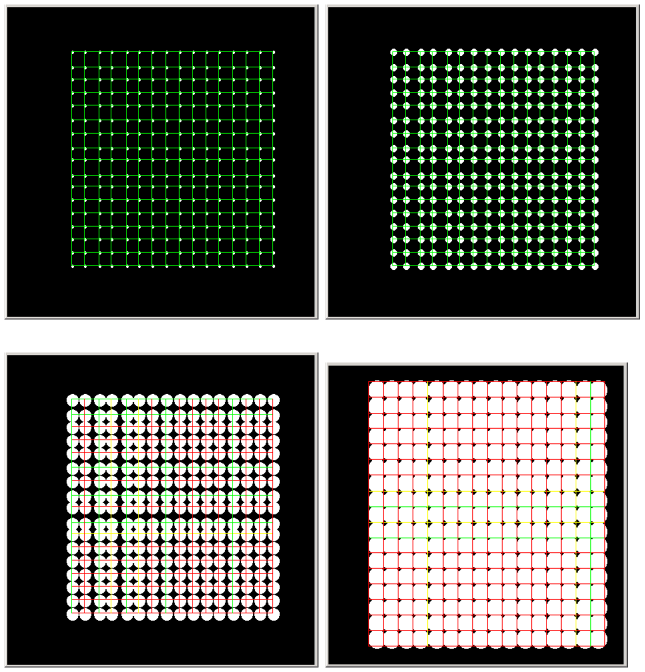


Figure 14: The grid alignment results for the images in Figure 13.

Another set of simulated images was formed to test an impact of blurred spots on the algorithm performance. Figure 15 shows a simulated image and the corresponding grid alignment result. While signal blur is present in scanned digital images to some degree, the blur is also introduced when images are down-sampled for computational efficiency purposes. Apparently, any signal blur will decrease the signal contrast and increase the edge transition length between the signal and its background. From the grid alignment standpoint, smaller contrast means less discrimination power in the score function (max minus min range) and wider edge transition means larger spatial errors in determining grid lines (broader local minima in score function). These two facts can be seen from the score function plots in Figure 15. For the simulated array in Figure 15 (spot radius is 5) and varying amount of signal blur introduced by a low-pass filter with kernel values in [3,13], the total misalignment errors according to the Equation 4 are within the range [14.22%, 22.09%]. The alignment fails for images blurred with a low-pass kernel larger than 14.

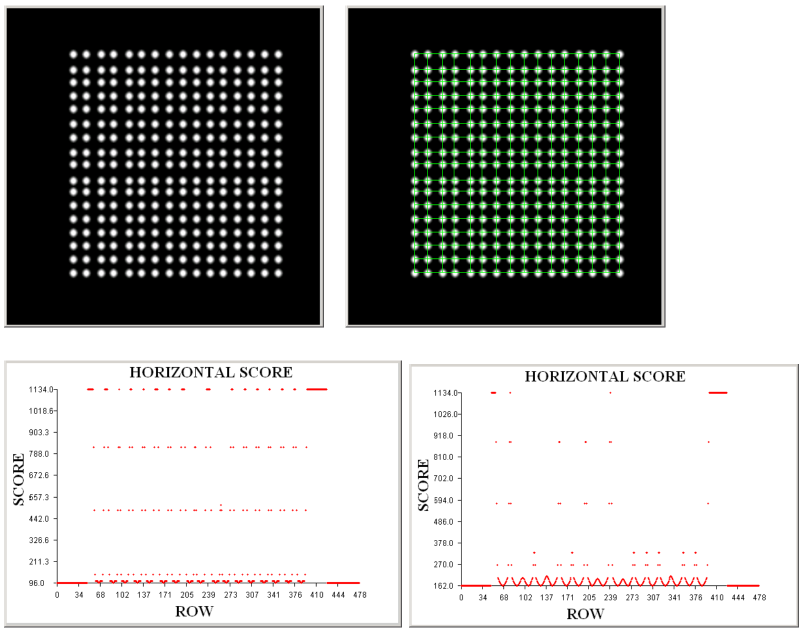


Figure 15: A simulated 2D array of spots (upper left) after blurring the spots with a low-pass filter (kernel = 5). The resulting grid (upper right) and the corresponding horizontal score function

(lower left) used for finding horizontal lines. The horizontal score function (lower right) for another simulated image with more blur (kernel = 10).

As it was stated before, image down-sampling will introduce not only signal blur but also reduction in spot size and possible partial overlap of spots in neighboring grid cells. These factors will impact spatial accuracy of the line detection. One can see visually the impact on spatial alignment accuracy in Figure 16. The spatial inaccuracy originates from the score functions due to spot overlaps and intensity blurs in down-sampled images, and these factors were investigated separately before. However, the benefit of analyzing down-sampled images is the computational speed summarized in Table 2. Table 2 illustrates a tradeoff between the misalignment error computed according to Equation 4 and the approximated execution time on a DELL PC with x86 processor. The time was approximated by averaging real time instead of CPU time since the code is written in Java that does not support CPU time measurements. Although the error for the case with no down-sampling might seem too high (6.4%), it is caused by a circular shape of the array primitive overlaid on a square image grid. Thus, a rectangular shape would lead to more accurate alignment results from an image-processing standpoint. Another observation can be made about the tradeoff by viewing Table 3. The numerical values in Table 3 suggest that the benefit of computational speedup is inversely proportional to the misalignment error albeit we cannot generalize this result because of the small number of simulations.

Table 2: Experimental results showing a tradeoff between the misalignment error computed according to Equation 4 and an approximated execution time for an image of size 479x460 and the run-time parameters Kernel = 2, Sensitivity = 0, MinAngle = MaxAngle = 0.

DownSamp Parameter	1	2	3
Misalignment Error	6.4%	14.6%	21.3%
Execution Time	175ms	65ms	45ms

Table 3: The misalignment and error ratios of down-sampled cases with respect to the case with no down-sampling.

Compared Cases of DownSamp	1 versus 2	2 versus 3
Inverse Error Ratio	2.28	1.46
Time Ratio	2.69	1.44

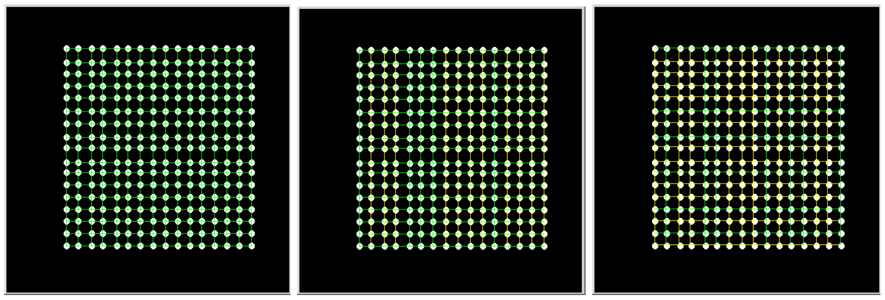


Figure 16: The results obtained without any down-sampling (left, DownSamp=1), and with down-sampling by a factor of 2 (middle) and 3 (right).

It is well known that in a measured microarray image, spots are missing quite often due to a variety of reasons [14], for example, printing error or low expression level. As in every data-driven approach, if there is no signal to support the presence of a grid then the proposed data-driven algorithm fails. In reality, it is very rare that an entire row or column would not have a single spot. If there would be obviously no information about lines in such rows or columns without spots, then this situation could be remedied by an editing tool that would allow the grid alignment quality inspector to insert a missing line. We have investigated more realistic scenarios when a few spots are missing in each line. Figure 17 shows a simulated image with a varying number of spots in each line. The plot in Figure 17 (right) illustrates a robustness related issue due to the lack of spots. Smaller number of spots in a line means less discrimination power in the score function. This agrees with our intuition that a grid alignment of 2D arrays with small number of spots in a line will be less robust to missing spots than an alignment of 2D arrays with large number of spots.

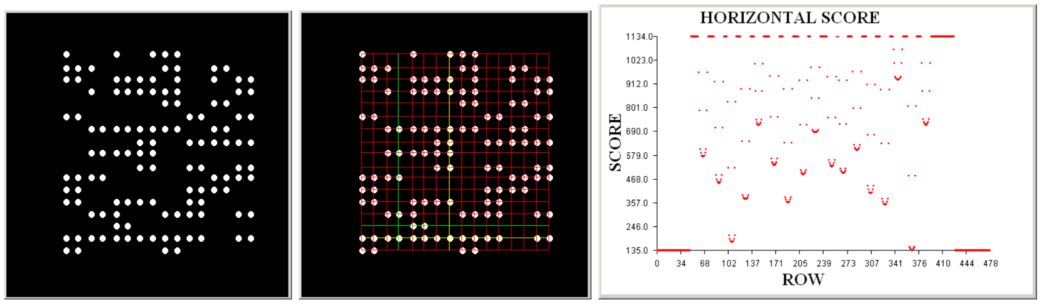


Figure 17: A simulated 2D array of spots (left) with varying number of spots in each row and column. The resulting grid (middle) and the corresponding horizontal score function (right) used for finding horizontal lines.

In general, all simulated variations can be present simultaneously in a microarray image. A decomposition of spot variation types revealed individual error contributions due to each variation.

5 SUMMARY

We have presented a novel data-driven grid alignment algorithm that (1) detects irregularly row- and column-spaced spots in a 2D array, (2) is independent of spot color and spot size, (3) localizes a grid of other primitive shapes than the spot shapes, (4) performs grid alignment on any number of image channels, (5) reduces the number of free parameters to minimum by data driven optimization of most algorithmic parameters and (6) has a built-in speed versus accuracy tradeoff mechanism to accommodate user's requirements on performance time and accuracy of the results. The 2D array processing, parameter optimization and processing of multiple grids were described in this work.

In the experimental part of this paper we demonstrated capabilities of the proposed grid alignment algorithm and evaluated its robustness and performance on a set of synthetic images. Among the capabilities, we have shown (1) grid alignment accuracy as a function of the number of channels, (2) inaccuracy of template-based methods in comparison with the proposed data-driven method, (3) alignment of 2D arrays with color and shape varying grid primitives, (4) alignment of multiple sub-arrays and (5) alignment of rotated sub-arrays. In the evaluation part, we have focused on the robustness and performance criteria, such as, grid alignment accuracy as a function of (1) spot radius, (2) spot blur, (3) down-sampling parameters and (4) missing spots.

In future, we would like to address other robustness issues including (1) misaligned spots in one line, (2) removal of background clutter, e.g., textual and bar code annotation on a slide (see Figure 1 right) and (3) alignment accuracy as a function of grid shape primitive.

ACKNOWLEDGEMENTS

The author would like to thank Dr. Lei Liu at the W. M. Keck Center for Comparative and Functional Genomics for providing microarray images for this work, as well as for having many fruitful discussions about microarray technology. He would also like to acknowledge many scientific interactions with Harris Lewin, Mark Band, Robin Evert and Zonglin Liu at the W. M. Keck Center for Comparative and Functional Genomics that led to this work.

REFERENCES

[1] P. Baldi and S. Brunak, “Bioinformatics, The Machine Learning Approach,” Second Edition, The MIT Press, Cambridge, Massachusetts, 2001.

[2] S. K. Moore, “ Understanding The Human Genome,” IEEE Spectrum, November 2000, pp. 33-42.

[3] J. Seo and B. Shneiderman, “Interactively Exploring Hierarchical Clustering Results,” Computer, July 2002, pp.80-86.

[4] S. H. Friend and R. B. Stoughton, “The Magic of Microarray,” Scientific American, February 2002, pp. 44-49.

[5] Affymetrix Inc., “Gene Chip Arrays,” Product Description at http://www.affymetrix.com/index.affx

[6] P. Bajcsy, “Image To Knowledge (I2K),” Software Documentation at http://alg.ncsa.uiuc.edu/tools/docs/i2k/manual/index.html.

[7] Axon Instruments Inc., “GenePix Pro,” Product Description at http://www.axon.com/GN_Genomics.html

[8] Scanalytics Inc., “MicroArray Suite,” Product Description at http://www.scanalytics.com/product/hts/microarray.html

[9] CLONDIAG Chip Technologies,” FluorIS: Array Standardization Tool,” Product Description at http://www.clondiag.com/products/dispo/fluoris/index.php

[10] Packard BioChip Technologies, LLC, “Quant Array Analysis Software,” Product Description at http://www.packardbioscience.com/products/521.asp.

[11] Imaging Research Inc, “Array Vision,” Product Description at http://www.imagingresearch.com/products/Genomics_Software.asp.

[12] M. Eisen, “ScanAlyze, ” Product Description at.http://rana.lbl.gov/EisenSoftware.htm

[13] CSIRO Mathematical and Informational Sciences, “Spot Image Analysis Software,” Product Documentation at http://experimental.act.cmis.csiro.au/Spot/index.php

[14] J. Buhler, T. Ideker, D. Haynor, “Dapple: Improved Techniques for Finding Spots on DNA Microarrays,” UV CSE Technical Report UWTR 2000-08-05.

[15] I. Foster and C. Kesselman. “Computational Grids,” *Chapter 2 of "The Grid: Blueprint for a New Computing Infrastructure"*, Morgan-Kaufman, 1999.

[16] A. B. Goryachev, P. F. MacGregor and A. M. Edwards, “Unfolding Microarray Data,” Journal of Computational Biology, Volume 8, Number 4, 2001, pp. 443-461.

[17] M. Steinfath, W. Wruck, H. Seidel, H. Lehrach, U. Radelof, and J. O’Brien, “Automated image analysis for array hybridization experiments,” Bioinformatics 2001 17: 634-641.

[18] A. N. Jain, T. A. Tokuyasu, A. M. Snijders, R. Segraves, D. G. Albertson and D. Pinkel, “Fully Automated Quantification of Microarray Image Data,” Genome Research, Vol. 12, Issue 2, February 2002, pp. 325-332.

[19] M. Katzer, F. Kummert and G. Sagerer, “Robust Automatic Microarray Image Analysis,” In Proceedings of the International Conference on Bioinformatics: North-South Networking, Bangkok, 2002.

[20] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M Loh, J.R. Downing, M.A. Caligiuri, “Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring,” *Science* 286: 1999, pp. 531-537.

[21] M. Karo, C. Dwan, J. Freeman, J. Weissman, M. Livny, E. Retzel, “Applying Grid technologies to bioinformatics,**”** *Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing*, 2001 pp. 441 –442.

[22] E.R. Davies, “Machine Vision: Theory, Algorithms, Practicalities,” Academic Press, 1997.

[23] Y.H. Yang, M. J. Buckley, S. Dudoit and T.P.Speed, “Comparison of Methods for Image Analysis on cDNA Microarray Data,” Technical Report #584, Department of Statistics, University of California at Berkeley, November 2000.

[24] Y. Chen, E. R. Dougherty, and M. L. Bittner, "Ratio-Based Decisions and the Quantitative Analysis of cDNA Microarray Images," Journal Of Biomedical Optics 2(4), 364–374, 1997.

[25] Y. Balagurunathan, E. R. Dougherty, Y. Chen, M. L. Bittner, and J. M. Trent, "Simulation of cDNA Microarrays via a Parameterized Random Signal Model," Journal of Biomedical Optics, 7(3), 2002.