Microarray Sub-Grid Detection: A Novel Algorithm (code: SI-ZW)

Daniel Morris, Zidong Wang* and Xiaohui Liu

Abstract— In this paper, a novel algorithm for detecting microarray sub-grids is proposed. The only input to the proposed algorithm is the raw microarray image, which can be at any resolution, and the sub-grid detection is performed with no prior assumptions. The algorithm consists of a series of methods of spot shape detection, spot filtering, spot spacing estimation, and sub-grid shape detection. The algorithm is shown to be successful in dividing images of varying quality into sub-grid regions with no manual interaction. The algorithm is robust against high levels of noise, and high percentages of poorly expressed or missing spots. In addition, the developed algorithm is proven to be effective in locating regular groupings of primitives in a set of non-microarray images, suggesting that the algorithm may have application potential in the general area of image processing.

Index terms— Microarray, Gridding, Image filter, Shape detection, Sub-grid detection

I. INTRODUCTION

Microarray technology has provided the biology community with a powerful tool for exploring the genome. A robotic arrayer prints thousands of DNA sequences onto glass slides. Each slide will typically contain several sub-grids: two dimensional arrays of DNA spots. The slides are hybridised with mRNA from two sources that will be typically dyed red and green. After hybridisation the slides are scanned with a red and a green laser, the difference in fluorescence between the two colour channels shows the relative difference of gene expression between the two sources (Brown and Botstein, 1999). For more information on microarray technology please see Brown and Botstein (1999). Gridding is the process of identifying the sub-grids within an image and then the rows and columns of spots within each sub-grid. Gridding is the first main step in extracting data from a microarray image. After spot locations have been identified values for each spot are extracted and analysis of the difference in the levels of gene expression between the two sources can begin. Gridding is clearly the most critical stage of obtaining data from a microarray image, as misidentifying which spots represent which genes will significantly mislead the results analysis.

The task of gridding is often complicated by the following factors. (1) The spot shapes within an image are not consistently uniform. Within the gridding literature spots are often categorised as circular or donut shaped, however due to the creation process of a microarray slide, spots can be almost any shape. (2) Often many spots within an image will be poorly expressed, which can make even manual grid row and column identification difficult. (3) The grids within an image can be rotated with respect to the image edges. (4) Due to the wet lab creation of microarray slides, there are often noise artefacts and scratches on the slides. These can completely hide some spots from view, or partially obscure them. (5) The image background is often uneven with some areas significantly higher or lower than others. This can severely hinder threshold based

D. Morris, Z. Wang and X. Liu are with the Department of Information Systems and Computing, Brunel University, Uxbridge, Middlesex, UB8 3PH, U.K. Email: {Daniel.Morris, Zidong.Wang, Xiaohui.Liu}@brunel.ac.uk. Fax: 0044 1895 251686.

* Corresponding author.

techniques, as a threshold may return purely spot pixels in one area of an image, and return both spot and background pixels in another. (6) Spots and sub-grids can be printed slightly away from their ideal locations.

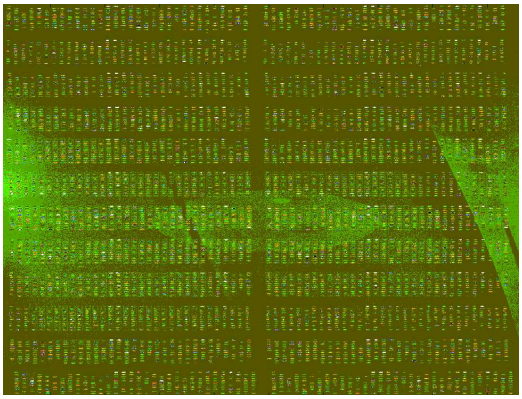

Figure 1, a typical microarray image. This image features 24 sub-grids.

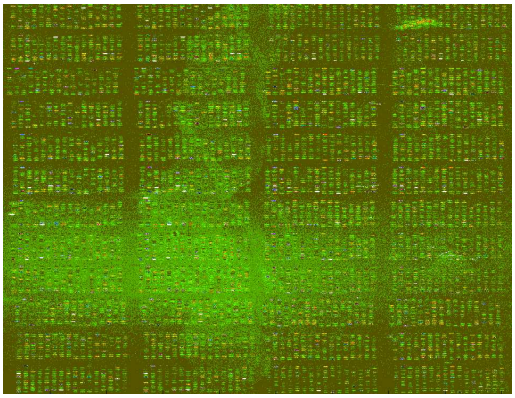

Figure 2, a typical microarray image. This image features 48 sub-grids.

Figure 1 shows a typical microarray image featuring 24 sub-grids, each sub-grid contains 12 rows and 32 columns of spots. Figure 2 shows another typical image, but with a different structure: 48 sub-grids, each one with 15 rows and 16 columns. Both images also feature clearly visible noise covering large regions.

As discussed in Morris and Wang (2006), initial Microarray Image Processing tools were completely manual, with users having to define the location of every spot within the image. Due to the number of spots on the average microarray slide numbering in the thousands and the demand for high throughput; semi-automated tools were created. Users could specify the number of rows and columns and then position a template grid over the image. Over time, these semi-automated tools have become more sophisticated and have required less and less user interaction. While there has been a large amount of work carried out recently into automated microarray gridding, many of the approaches require some degree of human intervention. Although a great deal of time has been spent researching automated microarray gridding, the focus has tended to be away from the task of *sub-grid detection*. Many relevant papers in the literature do not offer a solution for this critical phase of microarray image processing and begin with the subsequent task of sub-grid addressing (identifying the rows and columns within an individual sub-grid). The main purpose of this paper is therefore to shorten such a gap by exclusively tackling the sub-grid identification problems and developing novel algorithms that differ from the existing ones.

Microarray gridding (or sub-gridding) approaches are generally viewed as either being template-based or data-driven with various levels of automation (Bajcsy, 2006). Template based approaches can only be applied to images that do not feature any significant level of deviation from the expected model so are not suitable for wide spread use (Axon Instruments Inc.). The vast majority of data-driven approaches are based around the use of 1D projections, so much so that the term ‘data-driven’ has almost become synonymous with ‘1D projection’ in regard to microarray image processing. 1D projections are typically created by summing an image along its horizontal and vertical axes. The summation may not necessarily be of the raw image values but can be of some transform of the image, for instance the output of an edge detector (Bajcsy, 2004). For examples of the widespread use of 1D Projection analysis in microarray image processing see Bajcsy, 2004; Li *et. al.*, 2005; Hirata *et. al.*, 2001; Lonardi and Luo, 2004; Steinfath *et. al.*, 2001; Brändle *et. al.*, 2000; Wang and Huang, 2004; Bozinov, 2003; Fabbri *et. al.*, 2002; Blekas *et. al.*, 2003. Unfortunately, there appears to be several problems when the 1D Projection algorithms are applied to microarray image processing.

The first problem with using 1D projection for microarray image analysis is that microarray images can feature varying degrees of rotation; sub-grid edges are not always parallel to the image edges. This problem can be countered by calculating the angle of rotation and either rotating the image back through this angle or

using the calculated angle value as a parameter in subsequent gridding steps (Bajcsy, 2004; Hirata *et. al.*, 2001; Steinfath *et. al.*, 2001; Brändle *et. al.*, 2000; Wang and Huang, 2004; Bozinov, 2003). Many of the methods used to calculate the rotation angle require multiple rotations of the image, which due to the typical dimensions of a microarray image, is computationally expensive. Rotating the image back to solve the problem is also undesirable as it will alter spot morphology.

The second problem with using basic 1D projections is that this inherently assumes that a microarray image can be divided into its constituent sub-grids with straight lines drawn from one side of the image to another. As stated above, some microarray printing devices are capable of misprinting sub-grids away from their desired locations, so it is not always possible to successfully segment an image using only straight lines. Figures 3 & 4 illustrate this problem. Figure 3 shows a microarray image (included in the test set for the new approach described in this paper) which features a clearly visible sub-grid 'drift'. Figure 4 shows the same image with a vertical line placed alongside the left edge of the top right sub-grid, this line intersects with the bottom left sub-grid, illustrating that with this image sub-grid separation is not possible using straight vertical and horizontal lines.

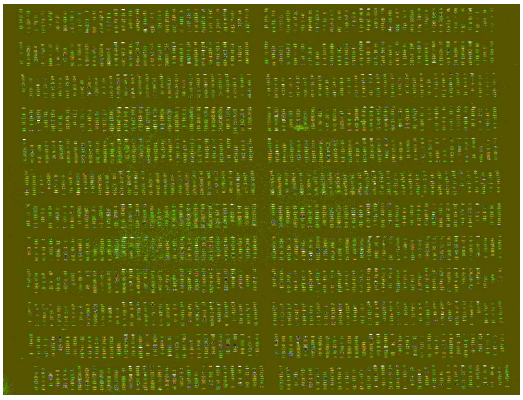

Figure 3, An image featuring significant sub-grid 'drift'

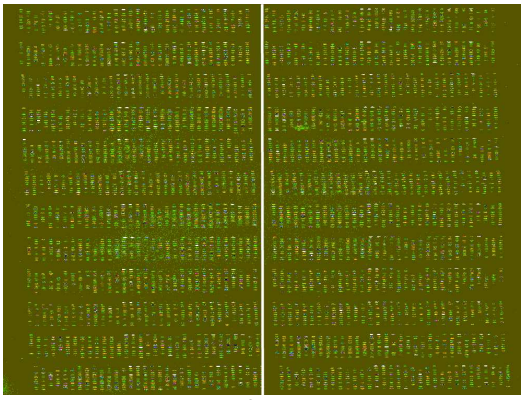

Figure 4, the same image from Figure 3, with a vertical line illustrating the difficulty of dividing the image into its' component sub-grids using straight lines.

An important assumption of using 1D projections to locate sub-grids is that the grids within the image are arranged in a regularly spaced 2D grid structure. This is a safe assumption to make, as currently the most commonly found microarray technology does print sub-grids in a 2D structure. However as microarray technology evolves it is conceivable that the structure of the images will change, so it is desirable to have a sub-grid detection algorithm that is future proof – in the sense that it can detect sub-grids of spots whether they are arranged in a 2D structure or in any other arrangement.

Another issue to be addressed in this paper is spot/sub-grid shape detection. The majority of documented techniques for locating spots/sub-grids have some predefined notion of the spot/sub-grid shape/size. For example in Antoniol and Ceccarelli (2005) the assumption is made that the spots within the image to be processed are circular and have a radius that is within predefined values. As these values do not appear to be calculated automatically, the assumption must be made that they are hard coded. In Steinfath *et. al.* (2001), the method of spot detection relies upon a "theoretical spot size", which, again, must be assumed to be hard coded as no calculation method has been provided. In Lonardi and Luo (2004), smoothing windows have been set to 25 and 4 pixels, when attempting to locate sub-grids and spots, but no calculation method for these values has been given, so this technique will only work with images featuring sub-grids and spots within a specific size range. The approach documented in this paper includes steps for detecting the size and shape of the spots within the image as well as the approximate dimensions of the image's sub-grids (based on previous work in Morris and Wang (2006)). These shapes and the values obtained with them can be of further use after the sub-grid detection phase, as demonstrated in our previous work (Morris and Wang, 2006). Detecting spot shape and sub-grid shape also helps to future proof the process. Developments in

microarray technology resulting in different spot shapes can already be seen in technologies such as the Affymetrix GeneChip (http://www.affymetrix.com).

In this paper, we deal with the microarray sub-grid detection problem. A novel algorithm is proposed which consists of a series of methods including; spot shape detection, spot filtering, spot spacing estimation, sub-grid shape detection and finally sub-grid detection. The main contributions of this paper can be summarised as follows: 1) a new method of sub-grid shape detection is developed, which builds on previous work in Morris and Wang (2006); 2) a new method of measuring the reliability of the detected shape is proposed, which allows for a mechanism of recalculation if an unsuitable shape is returned; 3) a new method of sub-grid detection, utilising multiple transformations of the original image is developed.

It is worth mentioning that, unlike many of the traditional gridding approaches, the sub-grid detection method developed in this paper does not utilise 1D projections. Our new methods are proven experimentally to be very robust against high levels of noise, high percentages of poorly expressed spots and all of the other problems associated with microarray sub-grid detection. Furthermore, while the algorithm is developed and tested for use with microarray images, its applications are far wider. Essentially this paper presents an algorithm for locating reoccurring shaped groupings of objects within an image. The groupings when applied to microarray images are the sub-grids and the objects are the spots. As well as being successfully tested with 50 real microarray images of varying quality, the algorithm has been tested on a series of artificial images and has proven experimentally to be more likely to be future proof against any changes in microarray gridding technology than previous sub-grid detection techniques.

The rest of this paper is organised as follows. Section 2, describes the first two contributions of this paper: a new method of sub-grid shape detection and the method of obtaining reliability score for the detected shape. Section 3 documents the major contribution of this paper, a method for searching a microarray image for sub-grids. Section 4 presents the results of various tests performed with the algorithm on 50 microarray images and a series of artificial images.

II. SUB-GRID SHAPE DETECTION AND SHAPE RELIABILITY CALCULATION

To scan an image for sub-grids it is first necessary to identify the shape of the sub-grids within the image. In this section a method of sub-grid shape detection is documented. The only inputs to this process are a real valued spot filtered image (*SFI*) and the horizontal and vertical spot spacing distances (in pixels) (*hSpacing* & *vSpacing*). The *SFI* and the *hSpacing* & *vSpacing* values can be calculated from the raw microarray image using techniques documented in our previous work in Morris and Wang (2006).

The process uses two key variables: *thresh* & *dilateMultiplier*. Initial values for *thresh* & *dilateMultiplier* can be randomly set, with *thresh* taking any value between 0 and 1 and *dilateMultiplier* taking any value between 0.5 and 2.

The key idea behind this stage is that if working with a perfect image (no noise and all spots regularly shaped and expressed above the background level of the image) where the distance between spots in the same grid is smaller than the distance of any two spots in adjacent clusters. If the *SFI* is subjected to a threshold and then dilated with a structuring element with dimensions equal to the spot spacing values. The resulting binary image should feature connected components that identify each of the sub-grids within the image.

Figure 5a, shows a manually created ‘perfect image’ with 4 sub-grids of 5 rows and 5 columns and no noise. Figure 5b shows the result of dilating the image with a rectangular structuring element with dimensions equal to the calculated *hSpacing* & *vSpacing* values for this image.

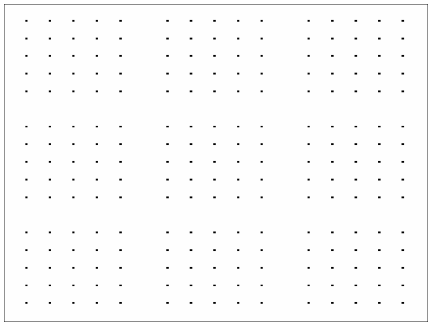

Figure 5a, an artificial 'perfect image', featuring 9 sub-grids.

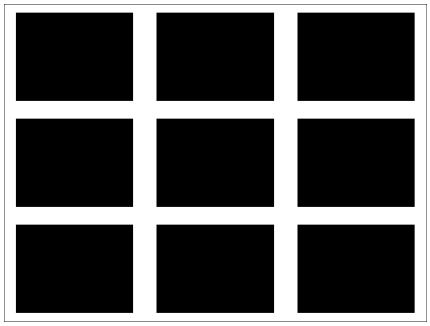

Figure 5b, Figure 5a dilated with a rectangular structuring element.

In reality it is not always possible with the *SFI* created from most microarray images to perform a single threshold and dilation to identify every sub-grid. This is mainly due to high numbers of poorly expressed spots which will disappear after the threshold. In addition if the threshold is set too low then false positives between sub-grids will merge them together during the dilation operation. But it is typically possible to threshold & dilate a microarray image's *SFI* so that many sub-grids are successfully identified. Figure 6a shows an area from a typical image's *SFI*. This image is subjected to a binary threshold (*thresh* = 0.5) and dilated using a rectangular structuring element, the dimensions of which are calculated by multiplying the *hSpacing* & *vSpacing* values by the *dilationMultiplier* (*dilationMultiplier* = 1.5), the resulting image is shown in figure 6b. It is clear in figure 6b that nine large connected components covering the majority of the nine sub-grids from the original image have been formed by the dilation.

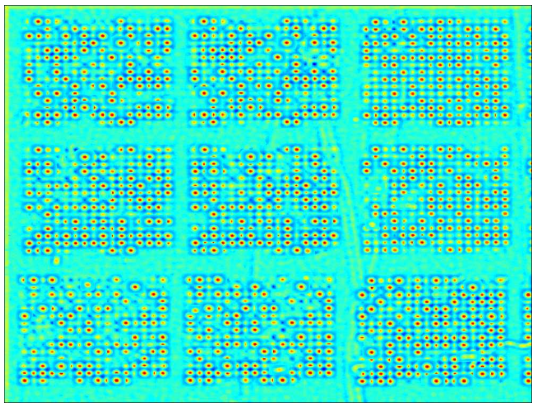

Figure 6a, An area from the *SFI* of a typical microarray image.

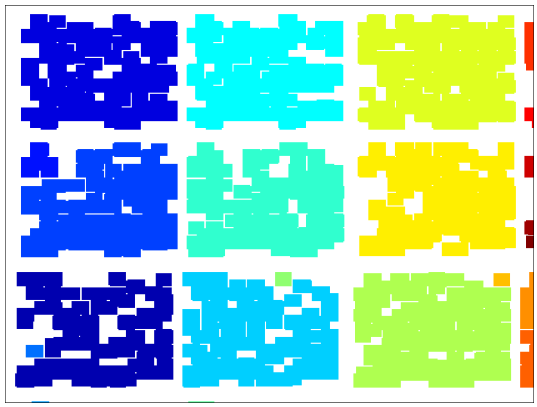

Figure 6b, Same area from Figure 6a having been subjected a threshold and a dilation. Each connected object is displayed in a different colour / shade

To obtain the approximate shape of the sub-grids within the image the mass of each object within the image is calculated, any objects with a mass smaller than 25% of the largest object are disregarded. This will remove the small objects which will often be created by isolated spots (spots whose neighbours were removed during the threshold operation) or sometimes false positives. The remaining objects are compared to each other - there are many methods of calculating a similarity value between objects, the decided measure was arbitrarily chosen as height and width – the object that is found to closely resemble the most objects in the image and all of the objects that closely resemble it are merged together to form an estimate of the sub-grid shape. Figure 7 shows the output of this process on the image in Figure 6a.

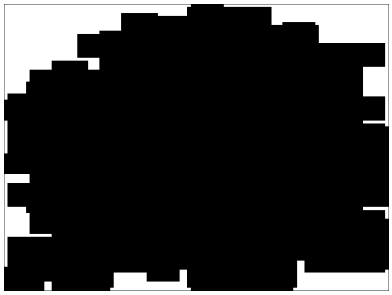

Figure 7, the approximate sub-grid shape.

A reliability measure for the output shape can be calculated by dividing the mass of all objects that resemble the representative object by the total mass of the image after the initial dilation. If the reliability measure is below a predefined value (set to 0.5 in the below code) then the process can be repeated with a different value for *thresh* and/or a different value for *dilationMultiplier*. If the reliability measure does not meet the predefined value after several iterations (set to 10 in the below code), then the shape with the highest reliability score can be taken as the sub-grid shape. The detected shape is stored into the variable *subGridShape*.

The outline of the whole sub-grid shape detection process is as follows:

```
Input: SFI, hSpacing, vSpacing

bestScore = 0
bestShape = null
rScore = 0
loopCounter = 0

while rScore < 0.5 AND loopCounter < 10
    loopCounter = loopCounter + 1
    dilateMultiplier = random value between 0.5 and 2
    thresh = random value between 0 and 1

    Create a rectangular structuring element se with dimensions of hSpacing*dilateMultiplier by
        ySpacing*dilateMultiplier.

    temp = (SFI > thresh) + (SFI < thresh*-1)
    dilate temp with se
    mass1 = sum(Temp)

    find the mass of the largest connected object in temp
    remove any objects that are smaller than 25% of this value

    tempShape = the most frequently occurring shape in temp
    mass2 = sum the mass of all objects in temp that resemble tempShape
    rScore = mass2/mass1
    if rScore > bestScore then
        bestScore = rScore
        bestShape = merge all shapes in temp that resemble tempShape
    end if
end while

output: rScore, bestShape
```

III. SUB-GRID DETECTION

Using the *SFI*, *subGridShape* and the row and column spot spacing values (*hSpacing* & *vSpacing*) it is possible to search the image for sub-grids. As was illustrated in section 2, subjecting the *SFI* to a binary threshold then dilating the result will often identify many sub-grids within the image. But it is often not possible to identify a single pair of values for *thresh* and *dilationMultiplier* that will successfully highlight every individual sub-grid in the image.

Therefore a threshold & dilation based approach used to locate all sub-grids within the image must take multiple views of the image created with a range values for *thresh* & *dilationMultiplier*. The set of threshold values used in our experiment were between 0.3 and 1, *dilationMultiplier* was set to take values between 0.5 and 2. The notion that multiple views of a microarray image are often required to obtain better knowledge of the image was previously exploited in Fraser *et. al.* (2004).

After each threshold and dilation operation the output image is searched for sub-grids. This is accomplished by comparing each of the connected objects within the new image to the known sub-grid shape. Any objects that closely resemble *subGridShape* are added to the final output image *finalOutput* provided that they do not significantly overlap any other objects which have previously been added to the output image.

Figures 8a,8b,8c & 8d show the state of the *finalOutput* over 4 iterations when the algorithm was used on the image from Figure 3. The initial processing of the image is performed with *thresh* = 0.3 & *dilateMultiplier* = 2 (see figure 8a) and correctly identifies 16 of the 24 sub-grids within the image. Figure 8b shows the second iteration with *thresh* = 0.3 & *dilateMultiplier* = 1.5, 2 more sub-grids are added to the *finalOutput*. Figure 8c shows the *finalOutput* after *thresh* = 0.3 & *dilateMultiplier* = 1, 4 more sub-grids identified. And in 8d with *thresh* = 0.5 & *dilateMultiplier* = 2 the remaining 2 sub-grids are added to the *finalOutput*.

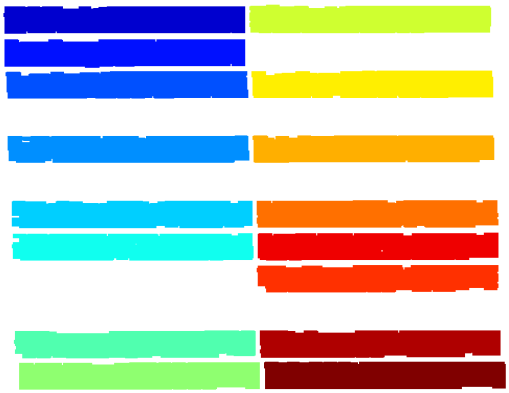

Figure 8a, *thresh* = 0.3 & *dilateMultiplier* = 2

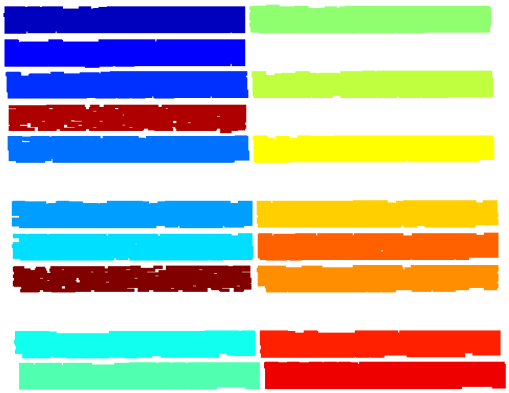

Figure 8b, *thresh* = 0.3 & *dilateMultiplier* = 1.5

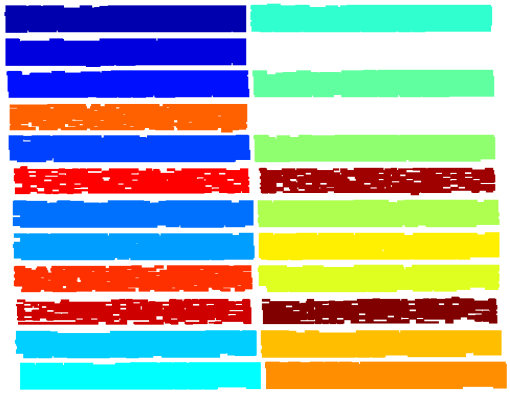

Figure 8c, *thresh* = 0.3 & *dilateMultiplier* = 1

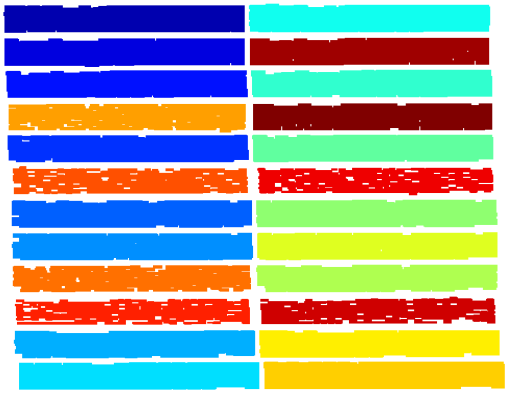

Figure 8d, *thresh* = 0.5 & *dilateMultiplier* = 2

The algorithmic description of the process is as follows:

```
Input: SFI, subGridShape, hSpacing, vSpacing

finalOutput = empty image, the same size as SFI

terminate = false

thresh = 0.3
while thresh < 1 AND terminate = false
    dilateMultiplier = 0.5
    while dilateMultiplier <= 2 AND terminate = false
        Create a rectangular structuring element se with dimensions of hSpacing*dilateMultiplier by
            ySpacing*dilateMultiplier.

        temp = (SFI > thresh) + (SFI < thresh*-1)
        dilate temp with se

        find any shapes in temp that resemble subGridShape
        add shapes to outputImage, provided they don't overlap any shapes already in the image

        terminate = imageFull (finalOutput, subGridShape)

    end while
    thresh = thresh + 0.1
end while

Output: finalOutput
```

A time saving device has been built into the algorithm, where if the output image can no longer fit any more sub-grid shapes (without overlapping shapes that are already in the image) then the process will terminate. This can be accomplished by eroding an inverse of the binary output image with a structuring element that is the same shape as the *subGridShape.* The process is in the function imageFull and is described as follows:

```
Function imageFull
    Input: finalOutput, se

    temp = finalOutput > 0
    temp = temp * -1 + 1

    test = erode temp by se

    if sum(test) == 0
```

```
            then x = true
            else x = false

        Output: x
    End Function
```

Once the process has terminated it has been shown beneficial to subsequent stages to place the known *subGridShape* over each of the detected sub-grids. The most obvious benefit being that many of the detected sub-grids will contain gaps and holes that will be filled by placing the *subGridShape* on top. Figure 9a shows an example of the *finalOutput* from a typical image, many of the sub-grids featuring small holes and chips around the edges. The 6th sub-grid down in the right hand column features a very noticeable missing region. Figure 9b shows the same image after the *subGridShape* was added on top of each sub-grid. The improvement is clear.

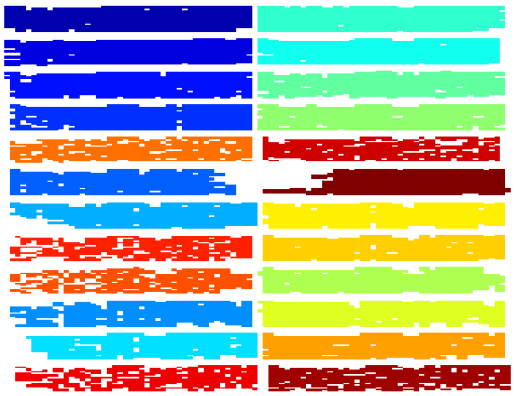

Figure 9a, *finalOutput* image featuring many holes and one sub-grid with a very noticeable chunk missing.

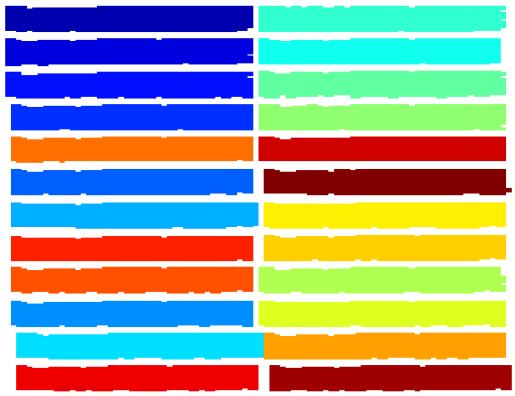

Figure 9b, The image from figure A1 having had the *subGridShape* placed over each sub-grid.

Some sub-grids may be slightly smaller in either width or height than the known *subGridShape* due to entire rows/columns of poorly expressed spots. When the known *subGridShape* is wider that a detected sub-grid the output image is tested to see whether the placement of the known *subGridShape* against the detected sub-grid's right or left edge will overlap any other grids. If placement is possible without any overlap then it is made. The same operation is performed with respect to the sub-grid's height and the top and bottom edges. Figure 10a shows the *finalOutput* from another image in the test set, this image includes several sub-grids which have several missing columns on their right hand sides (bottom right corner). Figure 10b, shows the finalOutput after the edge placement process is performed.

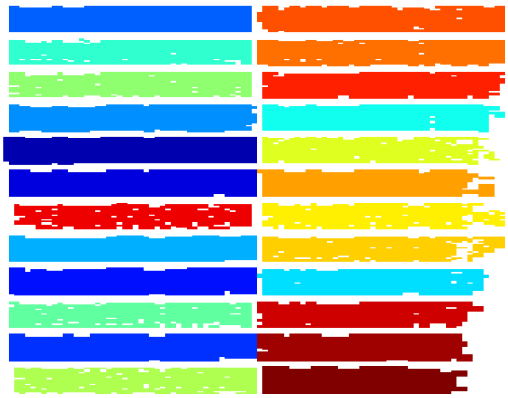

Figure 10a, finalOutput from a typical image featuring several sub-grids with missing columns.

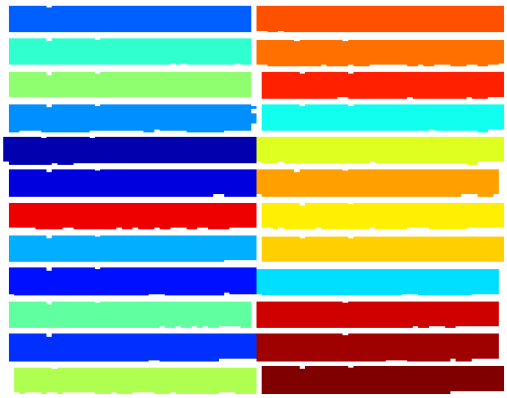

Figure 10b, Figure 10a after edge placement process.

It is desirable to assign every pixel in the output image to a specific sub-grid. This means that the output image can then be used as a sub-grid map for the original microarray image. This can be accomplished by dilating every sub-grid iteratively by a single pixel using a 3x3 structuring element. Any sub-grid pixels created by the dilation which overlap any other sub-grid are removed. Figure 10c shows the result of these dilations on the image from Figure 10b.

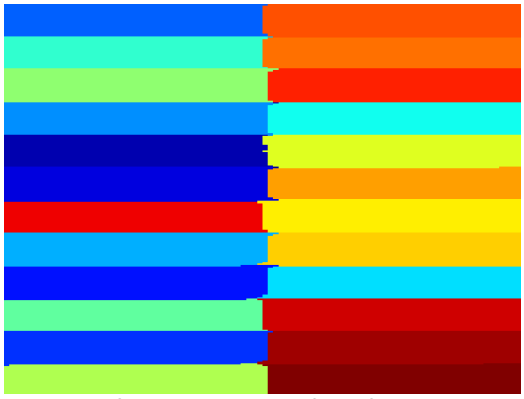

Figure 10c, Shows the result of passing the *finalOutput* image from figure 10b through the sub-grid dilation process.

IV. RESULTS

The most logical, and also the simplest way of testing the approach was to apply it to a set of real microarray images. The only input given to the algorithm for each image was the image itself and no data was carried over between images. The algorithm was tested with a set of 50 images, comprising two types of image. The first featuring 24 sub-grids arranged in a 12x2 format (see figure 1) the second containing 48 sub-grids arranged in 12x4 structure (see figure 2). The quality of the images were typical of the varying quality of images seen in microarray technology; some relatively clean, others with very high levels of noise and/or high numbers of poorly expressed spots. In all 50 images the algorithm successfully divided the images into their constituent sub-grids, with all spots correctly contained in the appropriate grid.

Some of the poorest quality images are shown in this section to illustrate the robustness of this approach. Figure 11a shows a microarray image which features some very sparse sub-grids. Each sub-grid in the image contains 384 spots (12 rows x 32 columns). In the top left sub-grid only 22 spots are visible – less than 6%. The top left sub-grid is shown close up in figure 11b. One of the lower sub-grids is shown in figure 11c for contrast. The approach was still able to successfully detect all 24 sub-grids within this image. Figure 11b shows the grid map for the image.

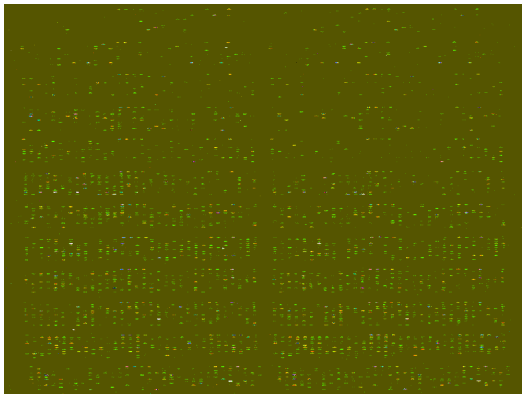

Figure 11a, an image featuring several sub-grids with very few visible spots

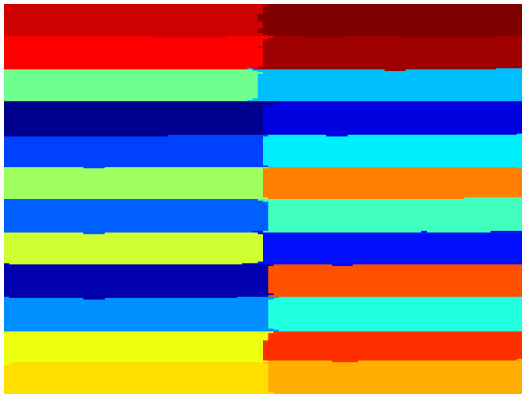

Figure 11b, the grid map for the image from figure 11a.

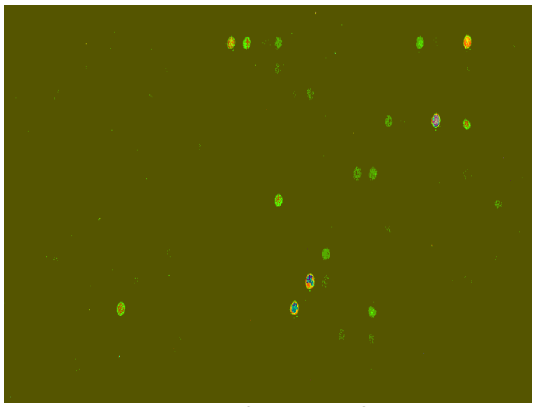

Figure 11c, the top left sub-grid from the image shown in figure 11a

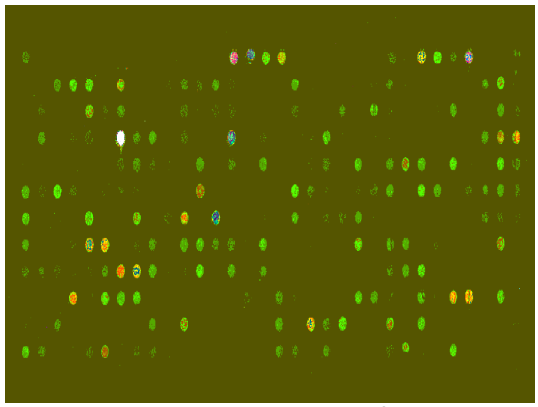

Figure 11d, a more visible sub-grid from the image shown in figure 11a

Figures 12a & 13a show images with very high levels of noise. Figures 12b & 13b show the corresponding grid maps created using the approach. All sub-grids were successfully identified, and valid boundaries created. It should be noted that in the image featured in Figure 12a (and several other images in the test set) that the level of background noise did severely hinder / prevent correct gridding using some of the traditional 1D approaches described in the first part of this paper.

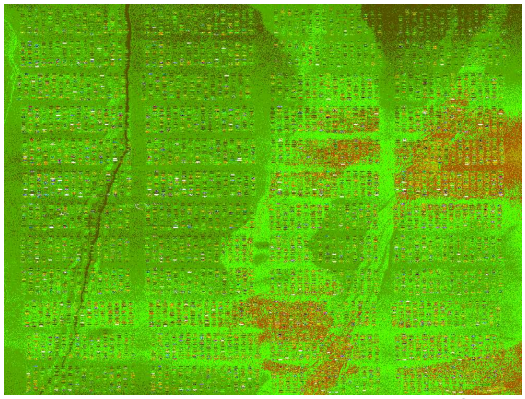

Figure 12a, an image with very high levels of noise.

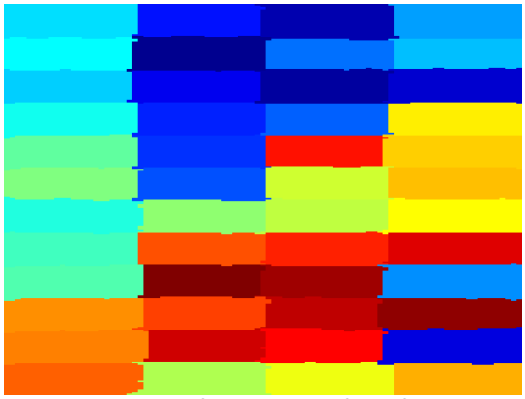

Figure 12b, the grid for the image from figure 12a, all 48 sub-grids successfully identified.

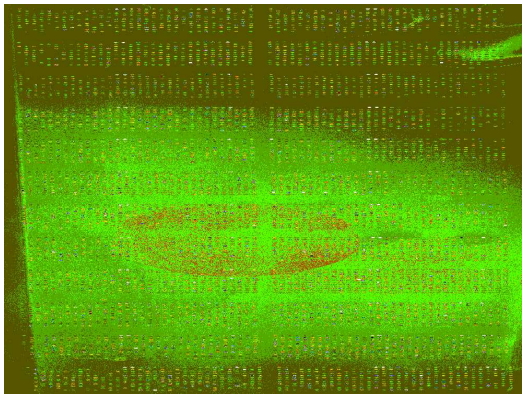

Figure 13a, another image with high levels of noise.

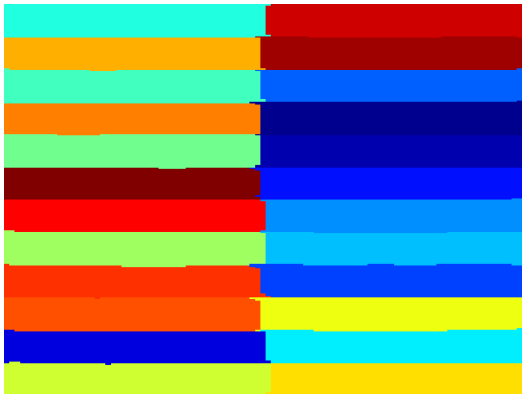

Figure 13b, the gridmap for the image in figure 13a.

After testing with all 50 images at their original size, the tests were repeated with resized versions of all 50 images at 50% and 25% of the original size. The results were identical, in all images the sub-grids were correctly identified, with all spots being contained within the correct region. As well as demonstrating the robustness of the approach this also provides a run time saving mechanism. By working with smaller images the run time was significantly reduced. Table 1, shows the run time of the entire process (*hSpacing* & *vSpacing* calculation, sub-grid shape detection and sub-grid detection) for a typical image at 4 different sizes.

Size Scale (%)	Run time (seconds)
100	1223
50	94
25	47
16.666	25

Table 1, run times for the entire process at various resized scales.

The algorithm was also tested with a series of artificial images, to test its' robustness against differing primitive shapes and differing arrangements of sub-grid primitives and sub-grid arrangements. Figure 14a shows one of the artificial test images, an image with regular groupings of triangular primitives, the groupings arranged randomly. Figure 14b shows the output of the approach, the algorithm successfully determined the shape of the primitive, the shape of the groupings of the primitives within the image, and then identified all of the groupings in the image.

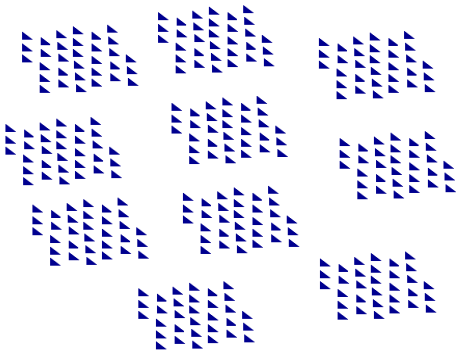

Figure 14a, an image with groupings of triangular primitives.

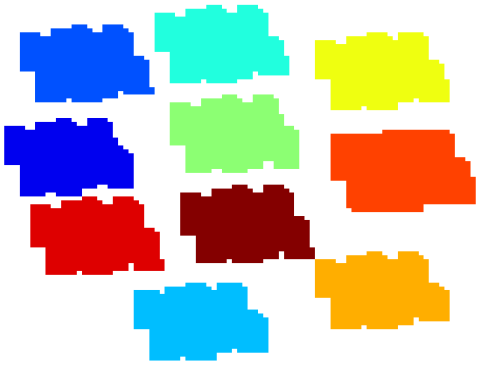

Figure 14b, output of the approach, all groupings clearly identified.

Figure 15a shows another artificial image from the test set. This image features triangular groupings of star shaped primitives. The groupings arranged in an ‘orange packing’ structure. Figure 15b shows the output of the approach, all groupings successfully identified.

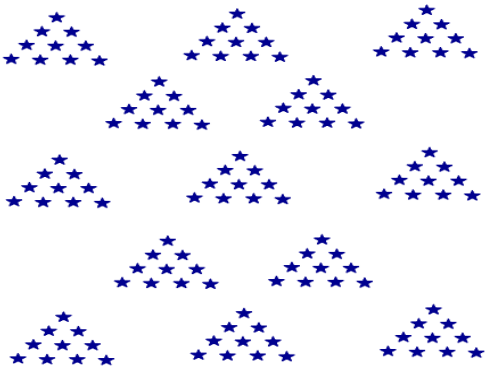

Figure 15a, an image with triangular groupings of star shaped primitives. The groupings arranged in an ‘orange packing’ structure.

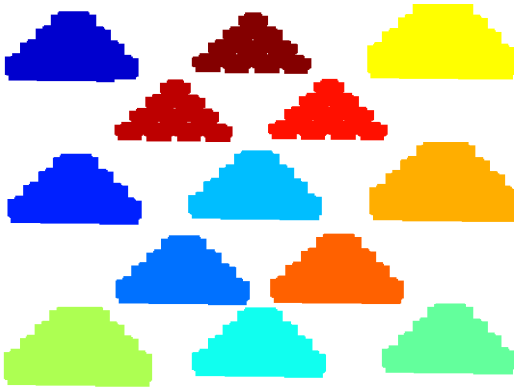

Figure 15b, output of the approach, again all groupings found.

For obvious reasons the more traditional 1D based approaches were unable to successfully demarcate the artificial images featured here.

V. CONCLUSIONS

In this paper it has been proven that the described approach for microarray sub-grid detection is robust against high levels of noise, high percentages of missing spots and all of the other factors that complicate the task of microarray sub-grid detection. The algorithm(s) tested have also proven to be potentially robust against future changes in microarray technology as they can cope with different shaped primitives, sub-grid configurations and sub-grid distributions. The approach can also work with a range of image resolutions, offering time saving benefits and proving robustness against an increase/decrease in primitive size. The documented method is clearly more robust than the common place 1D projection based approaches at future changes in microarray technology. The method may also offer application in other area’s, as proven with the artificial image set, providing a method of identifying and locating regularly appearing groupings of the same primitive within an image.

REFERENCES

G. Antoniol and M.Ceccarelli, “Microarray image addressing based on the radon transform”, Proc. IEEE International Conference on Image Processing, September 11-14, 2005, Genoa, Italy, 2005

Axon Instruments Inc., "GenePix Pro, Product Description" http://www.axon.com/GN_Genomics.html

P. Bajcsy, “Gridline: Automatic grid alignment in DNA microarray scans”, IEEE Transactions On Image Processing, vol. 13, no. 1, pp. 15-25, January 2004

P. Bajcsy, “An Overview of DNA Microarray Grid Alignment and Foreground Separation Approaches”, EURASIP Journal on Applied Signal Processing, vol. 2006, pp. 1-13, 2006

K. Blekas, N. Galatsanos and I. Georgiou, “An unsupervised artifact correction approach for the analysis of DNA microarray images”, IEEE ICIP (2) 2003, pp. 165-168, 2003

D. Bozinov, “Autonomous system for web based microarray image analysis”, *IEEE Trans. Nanobioscience*, vol. 2, no. 4, pp. 215-220, December 2003

N. Brändle, H. Lapp and H. Bischof, “Fully automatic grid fitting for genetic spot array images containing guide spots”, Technical Report (PRIP-TR-58), Pattern Recognition and Image Processing Group, Institute of Computer Aided Automation, Vienna University of Technology. December 2000

P.O. Brown and D. Botstein, “Exploring the new world of the genome with DNA microarrays”, Nature Genetics, vol. 21, pp. 33-37, 1999

R. Fabbri, L. Costa and J. Barrera, “Towards nonparametric gridding of microarray images”, In Proc. IEEE Digital Signal Processing 2002, vol. 2, pp. 623-626, July 2002, Santorini, Greece, 2002

K. Fraser, P. O’Neill, Z. Wang and X. Liu, “Copasetic analysis: A framework for the blind analysis of microarray imagery”, IEE Systems Biology, vol. 1, no. 1, pp. 190-196, 2004

R. Hirata, J. Barrera, R. Hashimoto and D. Dantas, “Microarray Gridding by Mathematical Morphology”, In Proc. Symposium on Computer Graphics and Image Processing, Florianpolis. IEEE, p.112-119, 2001

Q. Li, C. Fraley, R. Bumgarner, K. Yeung and A. Raftery, “Robust model-based segmentation of microarray images”. Bioinformatics, vol. 21, no. 12, pp 2875-2882, December 2005

S. Lonardi and Y. Luo, “Gridding and compression of Microarray Images”, In Proc. 2004 IEEE Computational Systems Bioinformatics Conference, August 2004, Stanford, CA, 2004

D. Morris and Z. Wang, “Truly blind microarray gridding”, IEEE Transactions on Systems, Man and cybernetics – Part C, accepted for publication, 2006.

J. Rahnenführer and D. Bozinov, “Hybrid clustering for microarray image analysis combining intensity and shape features”, *BMC Bioinformatics*, vol. 5, no. 47, 2004

M. Steinfath, W. Wruck, H. Seidel, H. Lehrach, U. Radelof and J. O’Brien, “Automated Image analysis for array hybridization experiments”, Bioinformatics, vol. 17, no. 7, pp. 634-641, 2001

Y.-K. Wang and C-W. Huang, “DNA microarray image analysis using active contour model”, *Proc.2004 IEEE Computational Systems BioInformatics Conference*, August 2004, Stanford, CA, pp. 549-550, 2004