Summer 2001

A dynamic programming-based approach to microarray image comparison and registration

Sunil Bosco Rodrigues

A DYNAMIC PROGRAMMING-BASED APPROACH TO MICROARRAY IMAGE COMPARISON AND REGISTRATION

by

SUNIL BOSCO RODRIGUES

A THESIS

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI-ROLLA

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN COMPUTER ENGINEERING

2001

Approved by

R. Joe Stanley
Dr. R. Joe Stanley, Advisor

Randy H. Moss
Dr. Randy H. Moss

Dr. Cihan H. Dagli

ABSTRACT

Microarray technology is increasingly used as a means of high throughput analysis of human, non-human and plant genomes. Manual methods of array production have inherent imperfections and variations of the quality of output data derived from these arrays. A dynamic programming-based approach is presented to validate the quality of such data prior to insertion into evolving databases in order to assign reliably, genetic aberrations to bio-medical functions. Dynamic programming is used for comparing normal and tumor microarray image pairs and a cost function is developed which gives a value representing the difference between the images. Quality of the image pairs is determined from this cost value. Dynamic programming makes it possible to have a backward-solution, which is used as the basis for developing the image registration algorithm. The image registration technique provides for an optimal alignment of the image pairs. The aligned image pairs help in further analysis of the spot-to-spot comparison between the two images to detect specific genetic expressions that could be related to bio-medical functions.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

Table Page

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my academic advisor Dr. Joe Stanley for his constant support, motivation, ideas and advice all along my graduate studies at UMR. I am indebted to all his invaluable time, patience and energy spent in countless discussions and meetings, some even during weekends and odd hours both during the research work and preparation of this thesis report.

I express my thanks to Dr. Cihan Dagli and Dr. Randy H. Moss for willingly agreeing to serve on my advisory committee.

I express my deepest gratitude to my mother Maria Rodrigues, father Antonio Rodrigues, brother Sanjit Rodrigues, sister Dr. Sharmila Mascarenhas and brother-in-law Joseph Mascarenhas for their love, prayers, blessings and good wishes. Although far in distance, their spirit kept me moving all along the challenges and trying times. I dedicate this thesis to my family.

I would like to thank my research teammate and friend Mahesh for giving me good company all along and especially for his spirited work towards the end. I also thank my other colleagues and friends, Ian Downard and Bill Seiver for the occasional helping hand they gave me during programming. Thanks to friends Meena D'souza, Bernard D'souza and Anil D'souza (all from Goa-India) for their constant goodwill and support. I also would like to thank other friends Raaj Shinde, Dipti Velingker, Sachin Lawande, Amit Raikar and Mario Creado for all their concern and moral support.

I thank all other colleagues, relatives and friends, names of whom I might have missed or not included due to space constraints.

1. INTRODUCTION

1.1. OVERALL SPECIFIC OBJECTIVES AND SIGNIFICANCE

Cancer is a complicated disease in which multiple molecular alterations occur inside a cell. The Cancer Genome Anatomy Project of the National Institutes of Health has begun to determine the complete DNA (deoxyribonucleic acid) sequence and gene expression profiles of certain tumors. Equally important are cancer epigenetic alterations, such as DNA hypermethylation, which silence tumor suppressor genes. One of the greatest intellectual challenges to investigations, such as this, is to extract the fullest meanings and implications of all available data. Implicit in this is the challenge to develop methods and tools to validate the quality of the genetic data for a meaningful analysis. The current five-year plan for the Human Genome Project (years 1999-2003) includes eight major goals, one of which involves bioinformatics and computational biology [1]. Two broad categories of bioinformatics needs were addressed: databases and development of analytical tools.

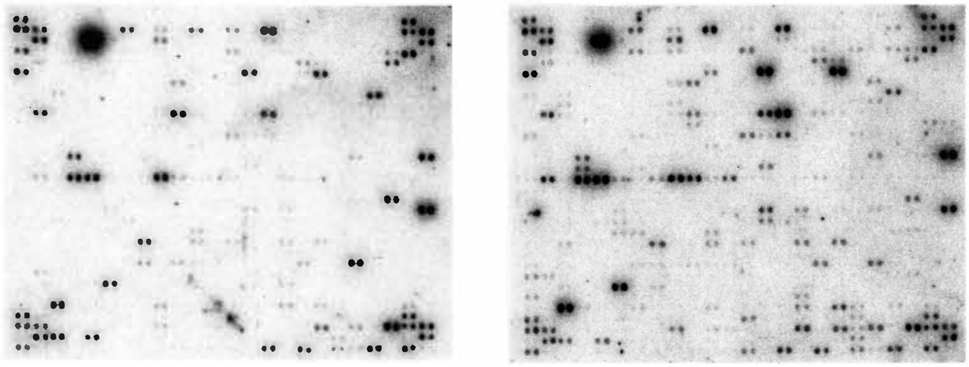
Traditionally, hypermethylation studies have been carried out by southern hybridization to assess known gene sequences. More recently, whole genome scanning using microarray technology has become feasible. The uses of microarray technology for studies of DNA methylation have been described by Huang, et al. [2,3]. This technology has enormous implications for high-throughput genetic studies.

Many laboratories are now using microarray technology as a means of high-throughput analysis of human, non-human, and plant genomes. Much of this work is performed using manual methods of array production, with inherent imperfections and variations in the quality of output image data derived from these arrays. It is therefore important to this endeavor to develop tools and methods that can validate the quality of such data prior to insertion into evolving genetic databases in order to assign reliably genetic aberrations to bio-medical functions.

There is an ongoing research project on breast cancer hypermethylation at the University of Missouri-Rolla and the University of Missouri-Columbia. From this research, there is a need to develop new visualization and analysis tools for high-throughput genetic microarray technology for ensuring data quality and reliability, before entering into a tumor-specific database. Developing image processing and analysis algorithms of DNA methylation microarray data will facilitate: 1) validation of data quality and 2) interpretation of data by biologists. The tools and methods developed will be broadly applicable to other laboratories, which use various methods and study different cancers, as well as, additional genomic projects such as maize and swine.

1.2. MICROARRAY IMAGE PAIR ACQUISITION

The microarray images used in this project were obtained as follows. First, an array of complimentary DNAs (cDNAs) is printed to a membrane and hybridized with a radiolabeled "normal" DNA sample. Second, a grayscale image of the signal intensity is produced from a phosphoimager [2,3]. Third, the "normal" DNA is stripped. Fourth, the same film is rehybridized with "tumor" DNA. Finally, a grayscale image of the signal intensity is produced from a phosphoimager [2,3]. Figure 1.1 contains an example of a normal and tumor image pair obtained from the currently used phosphoimager at Ellis Fischel Cancer Center at the University of Missouri-Columbia. DNA sequences in the microarray images always occur in pairs. Stripping of the normal DNA during the hybridization process causes the spots to shift from their original position.



(a) “normal” image

(b) Corresponding “tumor” image

Figure 1.1. Sample image pair acquired from phosphoimager

Hence, after re-hybridization of the array with the tumor DNA, a shift takes place in the corresponding spots in this image. Realigning these shifted spots is the main reason and requirement for image registration. The focus of this research was to develop a dynamic programming-based image registration technique that would facilitate data quality assessment and comparison of "normal" and "tumor" images.

1.3. OVERVIEW OF DNA METHYLATION

The master code that encodes all of the genes from microbes to man is made up of 4 bases that are abbreviated as G, A, T, and C. These are present in a type of molecule called DNA, which is like a very long ladder with pairs of these bases making up each of the "rungs" of the ladder. The base G pairs with C and A with T.

Medical terms related to the description of DNA methylation are defined here.

Base pair (bp): Two nitrogenous (purine or pyrimidine) bases (adenine and thymine or guanine and cytosine) are held together by weak hydrogen bonds. Two strands of DNA are held together in the shape of a double helix by the bonds between base pairs. The number of base pairs is often used as a measure of length of a DNA segment, e.g. 500 bp.

Purine: A nitrogen-containing, double-ring, basic compound that occurs in nucleic acids. The purines in DNA and RNA are adenine and guanine.

Pyrimidine: A nitrogen-containing, single-ring, basic compound that occurs in nucleic acids. The pyramidines in DNA are cytosine and thymine.

Adenine (A): Adenine is a purine base, is a constituent of nucleotides, and is one member of the base pair A-T (adenine-thymine) in DNA.

Thymine (T): Thymine is a pyramidine base and constituent of nucleotides and is one member of the base pair A-T (adenine-thymine) in DNA.

Guanine (G): Guanine is a purine base and constituent of nucleotides and is one member of the base pair G-C (guanine and cytosine)

Cytosine(C): Cytosine is a pyrimidine base and constituent of nucleotides and is one member of the base-pair G-C (guanine and cytosine)

Deoxyribonucleic acid (DNA): DNA is a molecule that encodes genetic information. DNA is a double-stranded molecule held together by weak bonds between base pairs of nucleotides. The four nucleotides in DNA contain the bases: adenine (A), guanine (G), cytosine(C), and thymine (T). In nature, base pairs form only between A and T and between G and C; thus the base sequence of each single strand can be deduced from that of its partner.

Complementary DNA (cDNA): DNA is synthesized from a messenger RNA template that corresponds to expressed sequences of genomic DNA. The term complementary

DNA may also refer to DNA that is complementary to a particular DNA sequence. The single stranded form is often used as a probe in physical mapping.

Ribonucleic acid (RNA): RNA is a chemical found in the nucleus and cytoplasm of cells; it plays an important role in protein synthesis and other chemical activities of the cell. The structure of RNA is similar to that of DNA. There are several classes of RNA molecules, including messenger RNA, transfer RNA, ribosomal RNA and other small RNAs, each serving different purposes.

Double helix: Double helix is the shape that two linear strands of DNA assume when bonded together.

Every cell in our bodies contains two copies of every one of our genes encoded in a DNA ladder, with one copy of each gene coming from our mother and one copy from our father. (The only exceptions to this rule are genes that determine whether we develop as a male or a female.) There are a limited number of genes that change in sequence during life, these are the specialized genes that encode antibodies. These genes need to be able to change in order to help our bodies fight off new bacterial or viral infections. Not every gene should be expressed in every cell of our bodies. For example, we do not want our brain cells to make hemoglobin, the protein required to carry oxygen around in our blood. Only those cells that will ultimately make red blood cells should make hemoglobin. If brain cells contain an intact copy (in fact 2 copies) of the gene for the hemoglobin protein, why do not brain cells also make hemoglobin? In fact, the level of the intermediate in hemoglobin protein production (a short copy of the hemoglobin gene in another molecule called RNA) is at least a million times lower in brain cells than in the cells that are in the process of turning into red blood cells. Hence, these are the processes that control what gene is expressed in what cell type, and in fact determine what type of cell it is. These processes can produce differences in activity of at least a million fold.

To regulate the level at which any gene is expressed, there are complex sets of regulatory proteins that bind to parts of the DNA encoding each gene. In complex organisms such as ourselves, many control factors have to be acting together to achieve the levels of power and refinement of gene regulation needed. One of these levels of control is provided by adding a small "tag" called a methyl group onto one of these letters that go to make up the DNA code, the C. The methyl group tagged Cs can be written as mC. Methyl group tags in the DNA of humans and other mammals plays an important role in the normal development and functioning of the organism. While there may be direct interactions between methyl group tags and regulatory proteins for different genes, it appears that the number and placement of methyl tags is an important signal for determining just how tightly folded that section of the DNA becomes.

It is now becoming clear that DNA methylation can play an important role in some diseases. With most genes, it probably does not matter that both copies of the gene (the one from the mother plus the one from the father) are both active. However, with a few genes, only one copy is normally active. This could be the copy from the mother or the father and which one of it is active is specific for that particular gene. Some babies are born with abnormalities due to both copies of the gene being active. This has been shown to be due to a failure in the establishment of the normal pattern of methyl group tags that blocks the activity of one of the copies of the gene. In addition, there is evidence that, in some cancers, genes that control the proliferation of cells can be inactivated by the abnormal addition of methyl group tags, which results in uncontrolled cell division.

Understanding the many complex roles of DNA methylation is an active field of research involving laboratories throughout the world. The DNA Methylation Society is an association formed by scientists interested in the many different aspects of this area of fundamental and applied research [4].

1.4. MICROARRAY HYBRIDIZATION

Microarray technology allows researchers to look at many genes at once, and determine which are expressed in a particular cell type. DNA molecules representing many genes are placed in discrete spots on a microscope slide; this is called a microarray. Thousands of individual genes can be spotted on a single square inch slide. Next, messenger RNA, the working copies of genes within cells, is purified from cells of a particular type. The RNA molecules are then "labeled" by attaching a florescent dye that allows us to see them under a microscope, and added to the DNA dots on the microarray. Due to the phenomenon termed as base pairing, RNA will stick to the gene it came from. After washing away all of the unstuck RNA, the microarray can be inspected under a microscope and see which RNA remains stuck to the DNA spots. Which of the gene each spot represents is known, and RNA only sticks to the gene that encoded it. Hence, it can be determined which genes are turned on in the cells. Some researchers are using this technology to learn which genes are turned on or off in diseased and healthy human tissues. The genes that are expressed differently in the two tissues may be involved in causing the disease. This is one of the precise applications of this project, to be able to determine the difference in the expressed genes related to tumor cells and the healthy (normal) cells in a patient having cancer.

While predisposition to some cancers is hereditary, many others result from genetic changes that occur throughout life. Using the techniques of modern molecular biology, researchers in the Cancer Genetics Branch (CGB) of the National Institute of Health (NIH), seek to define the genetic changes that lead to the initiation and progression of cancer. The CGB is involved in a range of experimental studies including the identification of inherited mutations predisposing family members to malignant melanoma, prostate and breast cancer. Microarray technology is used in the development and application of technologies for whole genome scanning of genetic variation and gene expressions [5].

1.5. OVERVIEW OF THE SECTIONS

Section 2 gives the background and overview of the basis on which the complete formulation of the cost function and the image registration algorithm are built. The basic idea and the central theme of sequence comparison are discussed. Furthermore, conceptual background on dynamic programming is presented. The advantages and preference of using the dynamic programming approach over other approaches is discussed in the literature review section.

Section 3 is devoted to the main algorithm. The pseudo-code for this algorithm is discussed and the complete mathematical formulation of the same is presented in this section. Section 4 describes the experiments and the results obtained based on the base dynamic programming-based image registration algorithm. Results are presented from the local image dataset available from the University of Missouri Ellis Fischel Cancer Center image database. Drawbacks and limitations of the base algorithm image registration are discussed. Section 5 presents a cascaded approach to image registration.

This is a modified version of the algorithm that is changed to adapt and overcome the drawbacks of the original algorithm in Section 3. Specific experimental results are shown and compared to show the improvement in the results.

2. BACKGROUND LITERATURE REVIEW

2.1. INTRODUCTION

The main objective of this research was to develop a quality measure for the microarray image pairs, and to concurrently register those images. The two images have to be invariably compared as a means to solve the above problem. Sequence comparison was chosen as the basis to compare the two images since it can also provide a measure of image quality. Dynamic Programming (DP) was used as a two-pronged approach for sequence comparison and it addresses both problems.

2.2. BASIC APPROACH TO SEQUENCE COMPARISON

It is often necessary to compare and contrast two or more sequences, or strings, or vectors, or continuous functions of time, etc. Comparison is used for identification, for error-correction, and for determining relationships and finding patterns. In many situations, there is a natural correspondence between the elements, or components, or coordinates, or points of time, etc., in one sequence and those in the other, and the only sensible comparison is between the corresponding elements. In such situations, it is easy to make the comparison. Sequence comparison deals with the more difficult comparisons that arise when the correspondence is not known in advance. Perhaps because some underlying correspondence has been disturbed by the gain or loss of elements in one or both sequences, or simply because one sequence is greater than the other is. Comparison is made even more complex and undefined when there are totally unrelated sequences. This research applied the concepts of one-dimensional DP into a two-dimensional approach to compare images (microarray) of unequal size, which had some degree of correspondence between its elements. The algorithm developed has been built based on the existing one-dimensional approach applied to human chromosome recognition [6]. Many existing methods for comparing sequences are applied to sequences of equal length, based on comparing corresponding elements. Examples of some well-known methods for comparing N element sequences **a** and **b** are Euclidean distance $[\sum_{i=1}^{N} (a_i-b_i)^2]^{1/2}$, city block distance $\sum_{i=1}^{N} |a_i-b_i|$, and Hamming distance, which is simply the number of positions in which the corresponding elements are different. Here a_i corresponds to b_i, and the comparison between them is expressed in terms of $(a_i-b_i)^2$ or the term $|a_i-b_i|$ or, in the case of hamming distance, a term that is 1 if $a_i \neq b_i$ and 0 if $a_i=b_i$.

The sequence comparison technique explored in this research incorporated city block distance into a two-dimensional dynamic programming-based approach. One sequence was taken as the reference and the other was taken as the comparator. The reference sequence was always taken as the shorter of the two sequences. Every element from the reference sequence was taken individually and compared to a group of pixels from the comparator. The general rule adopted was to seek the appropriate correspondence by optimizing over all possible correspondences that satisfy suitable conditions, such as preserving the order of the elements in the sequence, etc.

Three areas need to be addressed for performing sequence comparisons.

First, distance functions are used as and when appropriate when there is no natural correspondence of elements. Whenever there is an optimum one-to-one comparison between elements the distance function results in a zero penalty. However, when a group of elements in the comparator image (after averaging) gives the best optimum match to a single element in the reference image, it is selected to be the best optimum comparison with the numerical difference in the elements becoming part of the penalty given by the distance function [7].

Second, all the possible combinations, from a group of relevant elements in the comparator image, are used to find the best match with a single element from the reference image. Hence, the central idea is to combinatorially optimize and select the correspondences between the elements [7].

Third, the above methods are combined to give the basis to the dynamic programming algorithm developed for calculating distances and optimum correspondences [7].

2.3. LITERATURE REVIEW

Muramatsn, et al. [8] have researched an image pattern search method for detecting accurate pattern positions based on template matching between a template image and an input image. In their approach, 2-D images are transformed into 1-D data by matching vertical and/or horizontal projection profiles. Only feature points extracted from reference projection profiles are used in the DP matching, this was done in order to make the matching process fast. This approach is more designed towards accurate pattern position detection.

Another effective new class of algorithms, developed iteratively, applies dynamic programming to partially aligned sequences to improve their alignment quality [9]. Many other conventional algorithms are available that are dedicated to search patterns, string matching based algorithms in computer science, etc. Popular among those are the sequence graph or tree based with an edge representing a pair-wise alignment [10-11]. Grouping sequences according to their structural similarity or species origin and then conducting intra-cluster and inter-cluster alignments is researched and presented by Miller et al.[11]. Many other papers and literature related to molecular biology are available which focus on the alignment of multiple sequences. Some of the methods could be categorized into a) sequences approaches, b) tree approaches, c) clustering approaches and d) template approaches. Chan et al. lists all these methods in a comprehensive survey of multiple sequence comparison methods [12].

Ching Zang and Andrew K.C Wong have presented a genetic algorithm for multiple molecular sequence alignment [13]. They have proposed an approach, based on genetic algorithms, which is different than the other conventional algorithms mentioned earlier.

Essentially genetic algorithms are a set of stochastic algorithms for efficient and robust searching. They are developed in a way to simulate a biological evolutionary process and genetic operations on chromosomes. In genetic algorithms, the matching process starts from an array of points (states) in the problem space, instead of a single point in each iteration of the search. A better point is generated in each iteration of the search. Genetic algorithms are suitable for problems with large complexity and poorly understood search patterns [13].

Work in image registration is widely done in the area of medical imaging. A number of digital imaging techniques are available in medicine and these require the combinations of multiple images. Using these techniques, it is essential that the images be adequately aligned. Althof, et al. [14] describes an alignment routine developed to register an image of a fixed object containing a global offset error, rotation error and magnification error relative to a second image.

Sequence comparison approaches, as discussed earlier are essentially 1-D problem solutions or searching algorithms. Hence, none of these were directly suitable for the problem under consideration in this research. Most DP based approaches used for 2-D matching were essentially modified using 1-D methods. Genetic algorithm based methods related to molecular biology invariably used deletions and insertions or substitutions in their real sense. Although various methods are available in both 2-D and 3-D registration [15-17], they are all related to a specific problem domain and cannot be used directly for any other domain. Most of them are related to global alignment, and are essentially linear solutions based on the relative displacement of the images in terms of angle of orientation or displacement. The microarray image pairs examined for image registration requires a nonlinear-based approach due to the random shifts of the spots during the hybridization process. The image registration problem is explored using a DP-based approach such that the solution gives a minimum cost function and simultaneously allows a backward solution that helps in reconstruction of the registered image.

2.4. OVERVIEW AND CONCEPT OF DYNAMIC PROGRAMMING

Dynamic programming was the brainchild of an American mathematician, Richard Bellman, who described a way of solving problems where you need to find the best decisions one after another [18]. In the forty-odd years since this development, the number of uses and applications of dynamic programming has increased enormously.

The word programming in “dynamic programming” has nothing to do with writing computer programs. Mathematicians use the word to describe a set of rules that anyone can follow to solve a problem [18]. They do not have to be written in a computer language.

A mathematical way of thinking about dynamic programming is to look at what you should do at the end of a multiple-choice solution, if you get to that stage. Therefore, you think about the best decision with the last potential partner (which you must choose) and then the second to last one and so on. This way of solving the problem backwards is dynamic programming. Dynamic programming (DP) has the concept and principle of optimality. The other features of DP are to permit a backward solution, have a additive cost function and get a combinatorial complex solution.

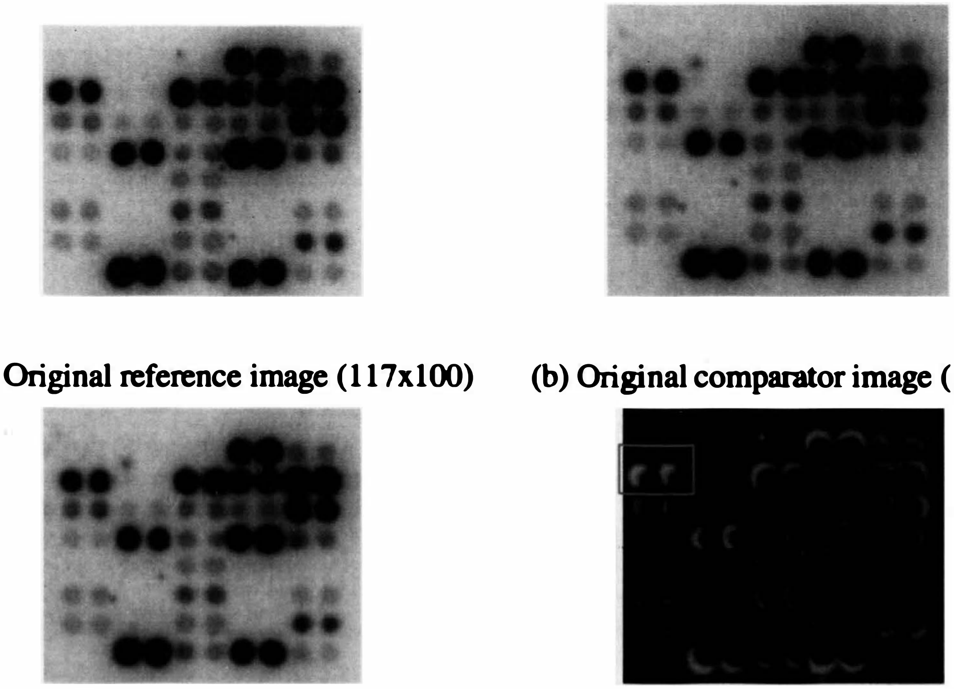
Dynamic Programming is conventionally used to find a minimal cost editing sequence to map a string into a path in a given network [19]. Costs are assigned to deletions, insertions, substitutions, and matches on each of the elements in a string [20]. The computation and math involved can be viewed as filling a cost matrix such that a trace from the upper left entry to the lower right entry defines an optimal editing of the string. In the context of the problem in this research, the direction matrix is tracked which is also similar in nature to the cost matrix. The direction matrix keeps track of the combination of the elements that represent the cost in the cost matrix for all the multiple cost combinations. Once the optimum solution of matching the two images is obtained, the direction matrix has all the information necessary to know which of the elements from the comparator image were recombined to get the optimum cost or the best match. Knowing this combination to obtain the best matching sequence is known as back solving. The direction matrix is therefore required for back solving the image registration problem since the best optimal cost of matching the entire image using DP is obtained at the very end. All the required information is stored in the direction matrix. The final cost that is obtained gives a definite measure on how closely the sequences are related. If two identical sequences are used as inputs, then we will have a cost score of zero. Totally unrelated sequences will give a relatively high cost. Hence, in this research the cost will be a reflection of the quality measure of the two images in terms of its background, noise, and relative shift of the spots.

3. ALGORITHM USED FOR COST FUNCTION DETERMINATION AND IMAGE REGISTRATION

3.1. INTRODUCTION

The main aim of this research is to develop a dynamic programming-based image registration technique, which will facilitate data quality assessment and alignment of normal and tumor images. The cost function for the dynamic programming algorithm provides the basis for determining the quality measure of the image pairs, while facilitating image registration. During microarray hybridization, the normal DNA is stripped before re-hybridizing the array with the tumor DNA. Due to this stripping, the individual spots corresponding to the two images shift with respect to each other. The primary requirement is that all these spots must be aligned with one another so that spot-to-spot comparisons may be performed. Due to the shift in the spots, the normal and the tumor images obtained are of unequal size and hence they need to be registered with each other.

Linear methods of resizing the images do not align the shift in the spots. Hence, a non-linear approach is required such that the images are resized with one another and at the same time aligned with respect to the spots in the two images. A small part of an image pair is cropped and shown in Figure 3.1. This figure demonstrates why a linear technique of image resizing does not have any improvement in the alignment of the spots. Figure 3.1(a) and (b) show the cropped original reference and comparator images. The spots in these images should have exactly matched with each other under normal circumstances. However, a shift in the spots occurs due to the manual methods of array production and the stripping of normal DNA that was explained earlier. The larger image from Figure 3.1(b) was resized using Paint Shop Pro^{TM} and is shown in Figure 3.1(c). As seen in Figure 3.1(d) the arithmetic pixel difference taken between the reference (Figure 3.1(a)) and the resized comparator image (Figure 3.1(c)) show the spots misaligned. Blurred spots are seen in this image because they are not perfectly aligned with each other.



(a) Original reference image (117x100)

(b) Original comparator image (124x108)

(c) Tumor image resized (size)

(d) Arithmetic pixel difference between (a) and (c)

Figure 3.1 Image example highlighting misaligned spots

Spots cannot be distinctly identified since there is misalignment between the individual corresponding spots in the two images. A region of such a misaligned spot is highlighted with a red border in Figure 3.1(d). Hence a non-linear technique is required, to align the spots between the two images.

3.2. DESCRIPTION OF THE ALGORITHM

The image registration process was performed between corresponding grayscale normal and tumor image pairs. Image registration requires determining the reference image and the comparator image. The reference image is defined as the image that is always referenced during dynamic programming-based image comparison. In the context of this research, the smaller image (width and height) was taken as the reference image. The reference image hence could be either the normal or the tumor image. Next, the images were normalized between 0 and 1 using the maximum pixel value from the reference image to obtain a linear mapping of the pixel gray levels from [0,255] to [0,1].

Image pairs, where the height of the first image is greater than the height of the second image, while the width of the former image is smaller than that of the latter image, cannot be processed using this algorithm. These image pairs are considered hybrid-cases, which require a different approach for image comparison and registration.

In the formulation of the dynamic programming-based technique for image registration, several operational definitions and rules were applied. The dynamic programming technique that was adopted is an extension of the sequence comparison presented by Stanley et al. [6]. This concept is used for 2-D images. The sequences are made up of the row elements from the reference and the comparator images.

Every pixel element from the reference image was compared with a group of pixels from the comparator image. No pixel from the reference image was deleted during registration of the image. Groups of pixels were averaged and substituted into the comparator image during registration. These groups of pixels were bounded by some rules that are considered logically appropriate and hence termed legal. The legal groups of pixels from the comparator image were bounded by the difference in the number of rows and columns between the reference and the comparator images.

A sample example is presented in Figure 3.2 demonstrating the number of pixel-group combinations that are taken into consideration while comparing a single pixel from the reference image with the comparator. This example is for an image pair with size difference of one row and two columns. The box with the letter 'X' in it represents the base pixel location in the reference image. Figure 3.2(a)-(f) shows all the possible combinations that are made for a single pixel from the comparator.

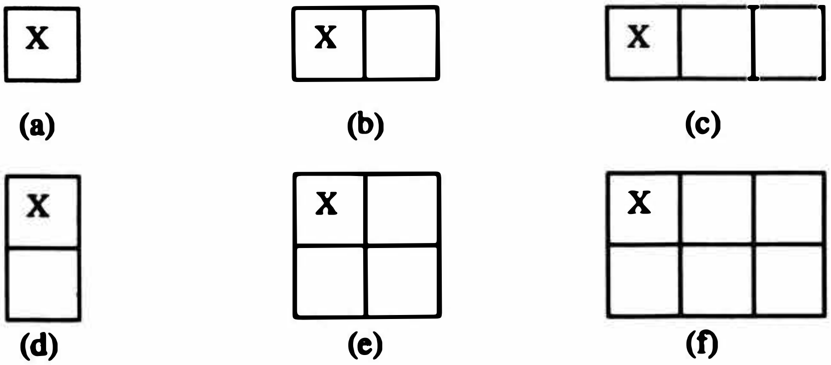


Figure 3.2. Illustration showing the combination of the legal pixel-group

To optimize the sequence comparison, only one row from the reference image was used for each run of the sequence comparison. However, multiple rows from the comparator image were used during this comparison. The maximum number of rows that can be combined is limited by the row difference between the reference and the comparator image. Layer combinations are the number of rows taken from the comparator image and compared with a single row from the reference image. Hence, when one row from the comparator image is taken and compared with one row from the reference image, it is termed as layer-1. When two rows from the comparator image are taken and compared with one row from the reference image, it is termed as layer-2 and so on. Figure 3.2 (a)-(c) represents a single pixel comparison for layer-1, while Figure 3.2 (d)-(f) represents pixel combinations for layer-2. As per the rules of the algorithm, it is possible to have as many layers of row combinations as there are row differences between the reference and the comparator images.

The cost function was developed based on the square of the pixel difference between the normalized value of the base reference pixel and the average value of the legal combination of pixels from the comparator image. A penalty was added to the above cost to account for the number of pixel combinations from the comparator image. This penalty was fixed to be one number less than the number of pixels from the legal group used in that particular comparison. The value of the cost function obtained is hence proportional to the size difference between the reference and the comparator images. For reference and comparator images of equal size, the cost is a pure reflection of quality of the images. Direct pixel-to-pixel comparison was done in such cases. The distance penalty cost was not added for comparison of equal size images. If the spots in the images are largely misaligned, the cost of matching will be higher since the spot-to-spot comparisons of the pixels will yield a greater difference.

The cost matrix height was initialized equal to the reference image width, and the cost matrix width was made equal to the comparator image width. The cost function calculates the incremental cost of combining the base pixels from the reference image with the comparator image. All the possible combinations that were made with every base pixel from the reference image were stored in the cost matrix along its rows. These entries are made starting from the upper-left corner to the lower right corner of the matrices in the form of diagonal elements. All elements below the diagonal are zeros, because those positions of the cost matrix imply deleted elements of the reference image. This was not done in this dynamic programming implementation. The direction matrix also has a similar structure and size and keeps track of the number of pixels in the legal pixel-groups that have been combined to get the best match. The direction matrix stores this information along its diagonal, exactly corresponding to the cost matrix value stored.

Another matrix that was used is the vertical direction matrix. This matrix keeps information about the row combinations within a given layer that gave the optimum cost for comparing the corresponding reference row. The vertical direction matrix has a number of rows equal to that of the reference image and a number of columns equal to the difference between the rows of the reference and the comparator image.

The dynamic programming-based approach, non-linearly determines the minimum cost for matching the normalized reference image to the normalized comparator image. Row-to-row comparisons and matching are performed for all legal combinations. Cost matrices were computed on a row-by-row basis using the cost data from prior rows for storing the incremental cost of matching all the legal combinations for the current row. Direction matrices were used during each row cost computation in parallel with every cost matrix to store the information about the number of the comparator elements that are combined during the comparison. Once the final minimum cost was found for comparing the entire image, the direction matrix was used to register the comparator image with the reference image. The pseudo-code for the dynamic programming-based algorithm developed is presented in Figure 3.3-3.5.

Figure 3.3 starts with the initialization of the image, cost and direction matrices. Normalization of the pixel values from [0,255] to [0,1] is performed here. This figure gives the pseudo-code for comparing the first row of the reference image with the comparator.

```
Read normal and tumor images
Determine reference and comparator images
Normalize reference and comparator images
Initialize cost and direction matrices
For first row of reference image
    For all layers
        For first column of reference image
            Match pixel of reference image with legal first pixel-group of comparator
            Store minimum cost in cost matrix corresponding to reference and comparator
                image comparison position
            Store comparator pixel combination yielding lowest cost match in direction
                matrix corresponding to reference and comparator image comparison
                position
        End for
        For columns 2 to second-last column of reference image
            Match pixel of reference image with legal pixel-group of comparator
            Store minimum cost in cost matrix corresponding to reference and comparator
                image comparison position (add previous pixel cost to get this minimum)
            Store comparator pixel combination yielding lowest cost match in direction
                matrix corresponding to reference and comparator image comparison
                position
        End for
        For last column of reference image
            Match pixel of reference image with legal last pixel-group of comparator
            Store minimum cost in cost matrix corresponding to reference and comparator
                image comparison position (add previous pixel cost to get this minimum)
            Store comparator pixel combination yielding lowest cost match in direction
                matrix corresponding to reference and comparator image comparison
                position
        End for
    End For
End For
```

Figure 3.3: Pseudo-code for first row reference image comparison algorithm

Pseudo-code for comparing middle rows from the 2nd to second-to-last row from the reference image is given in Figure 3.4. The difference in the procedure for first row comparison and middle rows is that the calculation of cost for middle rows includes the cost added from previous rows. No previous row cost is available for such an addition while calculating the cost for the first row.

```
For rows 2 to second-last row of reference image
    For all layers
        For first column of reference image
            Match pixel of reference image with legal first pixel-group of comparator
            Store minimum cost in cost matrix corresponding to reference and comparator image comparison position (add cost from previous row pixel to get this minimum)
            Store comparator pixel combination yielding lowest cost match in direction matrix corresponding to reference and comparator image comparison position
        EndFor
        For columns 2 to second-last column of reference image
            Match pixel of reference image with legal pixel-group of comparator
            Store minimum cost in cost matrix corresponding to reference and comparator image comparison position (add cost from previous pixel from current row and previous row to get this minimum)
            Store comparator pixel combination yielding lowest cost match in direction matrix corresponding to reference and comparator image comparison position
        EndFor For last column of reference image
            Match pixel of reference image with legal last pixel-group of comparator
            Store minimum cost in cost matrix corresponding to reference and comparator image comparison position (add cost from previous pixel from current row and previous row to get this minimum)
            Store comparator pixel combination yielding lowest cost match in direction matrix corresponding to reference and comparator image comparison position
        End For
    EndFor
EndFor
```

Figure 3.4. Pseudo-code for middle rows 2 to second-to-last row in the reference image

In Figure 3.5 the cost comparison for the final row from the reference image is calculated. Last row computation is calculatæd differently as compared to the first row and middle row computations. In this case, the last row from the reference image was compared with the last row from the comparator for calculation of the first layer. For second layer computation the last two rows from the comparator were taken and so on.

```
For last row of reference image
   For all layers
        For first column of reference image
           Match first column pixel of reference image with legal first pixel-group of
              comparator using cost from previous matches to find minimum cumulative
              cost
           Store minimum cost in cost matrix corresponding to reference and comparator
              image comparison position (add cost from the pixel of previous row to get
              this minimum)
           Store comparator pixel combination yielding lowest cost match in direction
              matrix corresponding to reference and comparator image comparison
              position
        EndFor
        For columns 2 to second-last column from reference image
           Match pixel of reference image with legal pixel-group of comparator
           Store minimum cost in cost matrix corresponding to reference and comparator
              image comparison position (add cost from previous pixel from current row
              and previous row to get this minimum)
           Store comparator pixel combination yielding lowest cost match in direction
              matrix corresponding to reference and comparator image comparison
              position
        EndFor
        For last column of reference image
           Match final reference column pixel with final legal comparator column pixel-
              groups using cost from previous rows
           Store minimum cost in cost matrix corresponding to reference and comparator
              image comparison position (add cost from previous pixel from current row
              and previous row to get this minimum)
           Store comparator pixel combination yielding lowest cost match in direction
              matrix corresponding to reference and comparator image comparison
              position
        End For
   EndFor
EndFor
Find minimum cost from among all the layers for the final row case
Identify layer with minimum final cost
Reconstruct comparator image using direction matrices (Separate Algorithm given in
Section 3.4 and 3.5)
```

Figure 3.5. Pseudo-code for the last row comparison from the reference image

After the final row cost computation, the layer with the minimum cost was identified as the winning layer. Reconstruction of the comparator image, to perform image registration is presented later in Section 3.4. The mathematical formulation of the above pseudo-code is presented in Section 3.3.

3.3. MATHEMATICAL FORMULATION OF THE ALGORITHM

The mathematical formulation presented in this section is derived from the pseudo-code algorithm given in Figures 3.3-3.5. The normal and tumor image matrices used in this section are assumed normalized using the maximum pixel value from the respective image.

The reference image matrix for an m_x by m_y image is defined as given below

$$Y= \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots\cdots\cdots & y_{1,m_y} \\ y_{2,1} & y_{2,2} & \cdots\cdots\cdots & y_{2,m_y} \\ | & | & \cdots\cdots\cdots\cdots\cdots & | \\ y_{m_x,1} & y_{m_x,2} & \cdots\cdots\cdots & y_{m_x,m_y} \end{bmatrix}$$

while the comparator image matrix for an n_x by n_y image is given as

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots\cdots\cdots & x_{1,n_y} \\ x_{2,1} & x_{2,2} & \cdots\cdots\cdots & x_{2,n_y} \\ | & | & \cdots\cdots\cdots & | \\ x_{n_x,1} & x_{n_x,2} & \cdots\cdots\cdots & x_{n_x,n_y} \end{bmatrix}$$

where $n_x \geq m_x$ and $n_y \geq m_y$

The maximum number of rows and columns for the comparator image are “m_x” and “m_y” respectively, while “n_x” and “n_y” represent the maximum number of rows and columns for the reference image.

Terms and variables used in the algorithm are defined as follows. The row number of the reference image for which the comparison is made is denoted as “r”, while “s” is denoted as the layer number. The number of layers is limited by the difference in the number of rows between the comparator image and the reference image plus one. If the reference and comparator images have an equal number of rows, there is only one layer (i.e., s=1). All the possible legal combinations within a layer are given by k_1. These are used in finding the minimum cost for each layer(s) after adding the layer cost value from the previous row (r-1). The column element from the comparator image is given by “i” while “j” gives the column element under comparison from the reference image,“i” also relates to the column of the cost and direction matrix while “j” corresponds to the row of the cost and direction matrix.

The current comparator element is defined as $x^*_{i,j,s,r}$ $d^*_{i,j,s,r}$ (ith comparator image column, jth reference image row; sth layer and rth reference image row) represents the distance-measure to be taken for matching the comparator image with the reference. The final minimum cost of comparing the above two image matrices Y and X is represented by ‘S_{m_x,m_y,n_x,n_y} ’. Element values of X and Y are the normalized image values. Each of the image matrices are normalized from [0,255] to [0,1].

The mathematical expression for the above pseudo-code is given in Figures 3.6-3.8. Figure 3.6 shows the cost calculation for matching the first row of the reference image with all the possible rows from the comparator. The calculation is done by layers of row combination from the comparator image with a single row from the reference image. The number of layers is defined as the difference in the number of rows between the comparator image and the reference image plus one. In this algorithm, each layer is represented by "s", where s ranges from 1 to n_x-m_x, where n_x and m_x are the maximum number of rows in the comparator and the reference image respectively.

Figure 3.7 represents the mathematical translation of the pseudo-code as given in Figure 3.4. This figure represents the combined cost calculation for all the other rows starting from 2 through n_x-1, where n is the number of rows in the reference image. The cost for all these subsequent rows is the cumulative cost and is calculated by adding the costs from the previous rows such that the cost calculated is optimized.

Figure 3.8 is a direct math version of pseudo-code from Figure 3.5, and shows the calculation of the final minimum cost for matching the final row of the reference image to the comparator. The cost obtained here is actually the final minimum cost for comparing the entire image. In the equations representing cost calculations, the distance-measure d^*-1 is used as the cost penalty for every comparison. The distance-measure is an approximation used to correctly reflect the higher cost involved for comparing a larger pixel-group from the comparator image verses a smaller pixel-group combination. If the pixel-group from the comparator image comprises of just one pixel, then d^*-1 works out to be zero and hence no distance-measure penalty is added for such a comparison.

Equations 3.1(a), 3.1.1(a), 3.1.2(a), 3.2(a), 3.2.1(a), 3.2.2(a), 3.3(a), 3.3.1(a), 3.3.2(a) give the distance-measure for each of the cost comparisons undertaken in the entire algorithm. The distance-measure gives the number of pixels that are combined from the legal pixel-group from the comparator image. This distance-measure is added to the cost function as a penalty for each comparison and is seen reflected in the subsequent cost function equations.

```
For r = 1
    For s = 1,......,$n_x$-$m_x$
        For j = 1
            For i = j,......,j + $n_y$ – $m_y$ :
                $d^*_{j,i,s,r}$ = (i-j+1)*s                                     ........Eqn: 3.1(a)
                $x^*_{j,i,s,r} = \dfrac{\sum_{c_1=r}^{r-1}\sum_{c_2=j}^{i} x_{c_1,c_2}}{d^*_{j,i,s,r}}$      ........ Eqn: 3.1(b)
                $S_{1,i,s,1} = ( | x^*_{1,i,s,1} - y_{r,j} | + | d^*_{j,i,s,r} - 1| )^2$        ........Eqn: 3.1(c)

        For j = 2,.......,$m_y$-1
            For i = j,....,j + $n_y$ – $m_y$
                For k = j,.....,i :
                    $d^*_{j,i,s,r}$ = (i- k+1)*s                                .......Eqn: 3.1.1(a)
                    $x^*_{j,i,s,r} = \dfrac{\sum_{c_1=r}^{r+s-1}\sum_{c_2=j}^{k} x_{c_1,c_2}}{d^*_{j,i,s,r}}$   .......Eqn: 3.1.1(b)
                    $S_{j,i,s,1} = \min_k((S_{j-1,k-1,s,r} +( | x^*_{j,i,s,1} - y_{r,j} | + | d^*_{j,i,s,r} - 1| )^2)$
                                                                                ......Eqn: 3.1.1(c)
        For j = $m_x$
            For i = $n_x$
                For k = $m_x$,.....,$n_x$ :
                    $d^*_{j,i,s,r}$ = (i- k+1)*s                                ..... Eqn: 3.1.2(a)
                    $x^*_{j,i,s,r} = \dfrac{\sum_{c_1=r}^{r+s-1}\sum_{c_2=j}^{k} x_{c_1,c_2}}{d^*_{j,i,s,r}}$   .......Eqn: 3.1.2(b)
                    $S_{m_x,n_x,s,1} = \min_k (S_{(j-1,k-1,s,r)} +( | x^*_{j,i,s,1} - y_{r,j} | + | d^*_{j,i,s,r} - 1|)^2)$
                                                                                .....Eqn: 3.1.2 (c)
```

Figure 3.6. Cost function calculation for the first row (r=1)

```
For $r = 2,\ldots\ldots,m_x-1$                    // row #
    For $s=1,\ldots\ldots,n_x-m_x$                // layer #
        For $k1 = 1,\ldots\ldots, s$              // # of run combinations per layer
            For $j = 1$
                For $i = j,\ldots\ldots,j + n_y - m_y$
                    $d^*_{j,i,s,r} = (i-j+1)*s$                                   ..... Eqn 3.2 (a)
                    $x^*_{1,i,s,1} = \frac{\sum_{c_1=r}^{r+k1-1} \sum_{c_2=j}^{i} x_{c_1,c_2}}{d^*_{j,i,s,r}}$   ..... Eqn 3.2 (b)
                    $S_{j,i,s,r} = \min_{k1}( S_{1,i,s-1,r-1} + (| x^*_{1,i,s,r} - y_{r,j} | + |d^*_{j,i,s,r} - 1|)^2 )$
                                                                                  .... Eqn: 3.2(c)
            For $j = 2,\ldots\ldots,m_y-1$
                For $i = j,\ldots\ldots,j + n_y-m_y$
                    For $k = j,\ldots\ldots,i$
                        $d^*_{j,i,s,r} = (i-k+1)*s$                               .... Eqn 3.2.1(a)
                        $x^*_{j,i,s,r} = \frac{\sum_{c_1=r}^{r+k-1} \sum_{c_2=j}^{k} x_{c_1,c_2}}{d^*_{j,i,s,r}}$   ..... Eqn 3.2.1 (a)
                        $S_{j,i,s,r} = \min_{k1} \min_{k} (S_{j-1,k-1,s,r}+S_{j,i,s-1,r-1}+(| x^*j_{,i,s,r} - y_{r,j} |+| d^*_{j,i,s,r} - 1| )^2)$
                                                                                  ..... Eqn 3.2.1(a)
            For $j = m_x$
                For $i = n_x$
                    For $k = m_x,\ldots\ldots, n_x$ : compute
                        $d^*_{j,i,s,r} = (i-k+1)*s$                               .... Eqn 3.2.2 (a)
                        $x^*_{j,i,s,r} = \frac{\sum_{c_1=r}^{r+k1-1} \sum_{c_2=j}^{k} x_{c_1,c_2}}{d^*_{j,i,s,r}}$   ..... Eqn 3.2.2(b)
                        $S_{m_x,n_x,s,r} = \min_{k1} \min_{k}(S_{j-1,k-1,s,r}+S_{j,i,s-1,r-1}+(| x^*_{j,i,s,r}-y_{r,j} |+| d^*_{j,i,s,r} -1| )^2)$
                                                                                  ...... Eqn: 3.2.2(c)
```

Figure 3.7. Cost function calculation for the middle rows 2 through m_x-1

```
For r = $m_x$                    // row #
   For s=1,.....,$n_x$-$m_x$           // layer #
      For k1 = 1,......, s          // # of run combinations per layer
         For j = 1
            For i = j,......j + $n_y$ – $m_y$
               $d^*_{i,j,s,m_x}$ = (i-j+1)*s                              ..... Eqn 3.3 (a)
               $x^*_{i,j,s,m_x} = \dfrac{\sum_{c_1=r}^{r+k1-1} \sum_{c_2=j}^{i} x_{c_1,c_2}}{d^*_{i,j,s,m_x}}$   ...... Eqn 3.3(b)
               $S_{i,j,s,m_x} = \min_{k1}( S_{1,i,s-1,r-1} + (| x^*_{1,i,s,r} - y_{r,j} |+| d^*_{i,j,s,m_x} - 1|)^2 )$
                                                                          ..... Eqn: 3.3(c)
         For j = 2,.......,$m_y$-1
            For i = j,....j + $n_y$-$m_y$
               For k = j,.....,i :
                  $d^*_{i,j,s,m_x}$ = (i-k+1)*k1                         .....Eqn 3.3.1(a)
                  $x^*_{i,j,s,m_x} = \dfrac{\sum_{c1=r}^{r+k1-1} \sum_{c2=j}^{k} x_{c1,c2}}{d^*_{i,j,s,m_x}}$   ..... Eqn 3.3.1(b)
                  $S_{i,j,s,m_x} = \min_{k1} \min_{k} (S_{j-1,k-1,s,r} + S_{j,i,s-1,r-1} + | x^*_{j,i,s,r} - y_{r,j} | + |d^*_{j,i,s,r} - 1|)^2$
                                                                          .....Eqn : 3.3.1(c)

         For j = $m_x$
            For i = $n_x$
               For k = $m_x$,.....,$n_x$ :
                  $d^*_{i,j,s,m_x}$ = (i-k+1)*s                          .... Eqn 3.3.2 (a)
                  $x^*_{i,j,s,m_x} = \dfrac{\sum_{c_1=r}^{r+k1-1} \sum_{c_2=i}^{k} x_{c_1,c_2}}{d^*_{i,j,s,m_x}}$   ..... Eqn 3.3.2 (b)
                  $S_{m_x,n_x,s,m_x} = \min_{s} \min_{k1} \min_{k}(S_{j-1,k-1,1,r} + S_{j,i,k-1,r-1} + (|x^*_{j,i,s,r} - y_{r,j}| + | d^*_{i,j,s,m_x} - 1|)^2)$
                                                                          ...... Eqn 3.3.2 (c)
```

Figure 3.8. Cost function calculation for the final row ($r = m_x$)

Equations 3.1(b), 3.1.1(b), 3.1.2(b), 3.2(b), 3.2.1(b), 3.2.2(b), 3.3(b), 3.3.1(b), 3.3.2(b) give the average value of the pixel-groups from the comparator image that are compared with the base pixel from the reference image for every comparison done in the algorithm. Every reference pixel is compared with the legal group of pixels from the comparator image as explained earlier in section 3.2.

Equation 3.1(c) gives the cost of matching the first pixel value from the first row and the first column in the reference image with the possible corresponding pixel-groups from the comparator image that are defined by the legal combinations of the pixel-groups discussed earlier. This equation also gives the cost for all the possible layers that correspond to the row difference plus one between the reference and the comparator images. Equation 3.1.1(c) gives the same cost as defined above for all the other pixel elements in the first row except the last pixel. However, this equation adds the cost from the previous pixels to give a cumulative cost for the row comparison for every layer. Equation 3.1.2(c) gives the generalized equation for the final minimum cost for all the layers for the first row.

Equations 3.2(c), 3.2.1(c) and 3.2.2(c) give the minimum costs as described above for rows 2 to m_x-1, where m_x is the number of rows in the reference image. These equations calculate the cost for all possible layers for each of the rows. To have the cost cumulatively combined all along the computation the above equations adds the appropriate cost from the appropriate layer in previous rows.

Equations 3.3(c), 3.3.1(c) and 3.3.2(c) are the costs related to the final row. Here the last row m_x from the reference image is compared with the last row n_x from the comparator image to compute the first layer cost. Costs from other layers are calculated by combining rows preceding n_x. Equation 3.3.2 (c) computes the final minimum single cost of comparing the complete reference image with the comparator image, taking into consideration and adding the combined cost from the previous row. The equation selects the minimum cost value from among the layer combinations. All these values are stored in the cost matrix as described earlier, while the distance-measure described earlier is stored in the direction matrix.

3.4. IMAGE REGISTRATION ALGORITHM

Image registration was done using the backward-solution technique based on the direction matrices obtained from the dynamic programming matching of the reference and comparator images. From the dynamic programming-based approach, direction matrices are determined in parallel with the cost matrices to provide information on the comparator pixel combinations that yielded the lowest cost. The comparator image is reconstructed to the size of the reference image using the direction matrix, which keeps track of the combined elements along the diagonal from the upper left corner of the matrix to the lower right corner of the matrix. Not all the rows that form a given layer need be used in the cost computation of that particular layer. A subset of rows that gives the minimum cost match may be selected to represent that layer. The vertical direction matrix was used to keep track of this subset of rows that are combined during each layer computation. Description and dimensions of the vertical direction matrix are discussed in Section 3.2. The vertical direction matrix was used to combine the correct row combinations that give optimum cost for the sequence comparison of every row in the reference image.

As explained earlier the dynamic programming-based cost computation was performed in layers. The number of layers per image pair comparison is dependent on the difference between the image heights of the two images plus one. At the end of the final row computation in the reference image, final minimum cost was chosen to be the value from the layer cost that has the least value. This layer with the minimum cost was chosen as the winning layer. Backward-solution starts first with the final (last) row setup and uses the direction matrix representing the winning layer. The last row is reconstructed by combining the pixels from the direction matrix corresponding to the winning layer. This direction matrix when backtracked gives the number of pixels that are combined horizontally along the row to obtain the minimum cost. The vertical direction matrix gives the exact rows from the comparator image that are combined to reconstruct each row in the registered image. Hence, both the direction matrix and the vertical direction matrix can determine the exact group of pixels used to compare the base element from the reference image. Once the final row is reconstructed, all the other rows (excluding the first row) are similarly reconstructed. The first row is reconstructed last to give the complete registered comparator image. The pseudo-code for the image registration algorithm is given in Figure 3.9; while its mathematical formulation is given in Figure 3.10.

The mathematical formulation for the image registration algorithm presented in this section was derived from the pseudo-code given in Figure 3.6. No combinations were done to the pixels from the reference image. Only the pixel-group combinations combined to form the minimum cost were averaged and replaced in the registered image. The gray value replaced at every pixel position in the registered image was computed as the average of the normalized value from the pixel-group combination multiplied by the maximum gray value found within the comparator image. The exact pixel-group combinations involved were obtained from the direction matrices.

```
For final row of reference image
    For all columns of reference image
         Recombine comparator image pixel values using direction matrix and vertical
            direction matrix
         Compute average of recombined pixel values
         Multiply average by maximum of comparator image to generate reconstructed
            image  pixel value
      End for
End for
For all middle rows of reference image
     For all columns of reference image
         Recombine comparator image pixel values using direction matrix and vertical
            direction matrix
         Compute average of  recombined pixel values
         Multiply average by maximum of comparator image to generate reconstructed
            image pixel value
        End for
  End for
  For first row of reference image
      For all columns of reference image
         Recombine comparator image pixel values using direction matrix and vertical
            direction matrix
         Compute average of  recombined pixel values
         Multiply average by maximum of comparator image to generate reconstructed
             image pixel value
       End for
  End for
```

Figure 3.9. Pseudo-code for the image registration algorithm

A registered image with gray levels from 0 to 255 was obtained. Pixel-groups that are legally selected during recombination comprise of the pixels selected from either a single row or a set of adjacent rows. Hence, groups of pixels from the same base row from the comparator image are combined during the image registration process to make up the row in the registered image.

The reference image matrix of m_x rows and m_y columns is defined as given below

$$Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots\cdots & y_{1,m_y} \\ y_{2,1} & y_{2,2} & \cdots\cdots & y_{2,m_y} \\ | & | & \cdots\cdots & | \\ y_{m_x,1} & y_{m_x,2} & \cdots\cdots & y_{m_x,m_y} \end{bmatrix}$$

while, the comparator image matrix of n_x rows and n_y columns is given as

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots\cdots & x_{1,n_y} \\ x_{2,1} & x_{2,2} & \cdots\cdots & x_{2,n_y} \\ | & | & \cdots\cdots & | \\ x_{n_x,1} & x_{n_x,2} & \cdots\cdots & x_{n_x,n_y} \end{bmatrix}$$

where, $n_x \geq m_x$ and $n_y \geq m_y$.

Maximum number of rows and columns for the comparator image are m_x and m_y respectively while, the maximum number of rows and columns for the reference image are n_x and n_y respectively.

The following algorithm shown in Figure 3.10, reconstructs the comparator image to the size of the reference image. Terms and variables used in the algorithm below are defined as follows:

The reconstructed normalized value of the registered image is given by $r^*_{m_x,i}$; $d^*_{i,j}$ is the value from the direction matrix, d[i][j][s][r] that gives information on which of the pixels from the pixel-group are combined horizontally from the comparator image to

```
For r = $m_x$
    s=v[r]                                                  ...... Eqn 3.4(a)
    For  i = $m_y$. . . . . 1                  // ( decrementing loop )
    $d^*_{m_x,1}$ = d[r][s][i][$n_x$]                       ...... Eqn 3.4(b)
        For j= $n_x$ . . . . ($n_x$-s+1)       // s is the layer with the min. final cost

            $r^*_{m_x,i} = \frac{\sum_{c_1=r-s+1}^{r}\sum_{c_2=i}^{d^*} x_{c_1,c_2}}{(d^*)*(s)}$    ....... Eqn 3.4 (c)

            r_img[r][i]= $r^*_{m_x,i}$ * max_val            ...... Eqn 3.4 (d)
    index=$m_x$ - s

For r= $m_x$-1. . . . . . 2  (decrementing loop)  // all middle rows
    s=v[r]                                                  ....... Eqn 3.5(a)
    For i = $m_y$ . . . . . 1
        $d^*_{m_x,1}$ = d[r][s][i][index]                   ...... Eqn 3.5 (b)
        For j = index . . . . . (index-s+1)

            $r^*_{r,i} = \frac{\sum_{c_1=r-s+1}^{r}\sum_{c_2=i}^{d^*} x_{c_1,c_2}}{(d^*)*(s)}$    ....... Eqn 3.5 (c)

            r_img[r][i]= $r^*_{r,i}$ * max_val              ....... Eqn 3.5 (d)
    index=index-s                     // decrementing the rows for "j"

For r = 1
    s=v[r]                                                  ...... Eqn 3.6(a)
    For i = $m_y$ . . . . 1
        $d^*_{m_x,1}$ = d[r][s][i][index]                   ...... Eqn 3.6 (b)
        For j = index . . . . index-s+1  // decrementing loop

            $r^*_{m_x,i} = \frac{\sum_{c_1=r-s+1}^{r}\sum_{c_2=i}^{d^*} x_{c_1,c_2}}{(d^*)*(s)}$    ....... Eqn 3.6(c)

            r_img[r][i]= $r^*_{m_x,i}$ * max_val            ...... Eqn 3.6 (d)
```

Figure 3.10. Image reconstruction and mathematical formulation

reconstruct the pixel corresponding to pixel (r,i) in the reference image; "s" is the winning layer for the final row as well as all the other rows, this variable gives information on which of the rows need to be combined. The value of "s" is obtained from the vertical direction matrix given by v[]. The maximum value from the comparator image that was used in normalizing the image before processing is given by "max_val". The reconstructed registered comparator image is given by r_img[][].

Equations 3.4 (a,), 3.5 (a) and 3.6(a) give the values from the vertical direction matrix. This value gives which of the row combinations are required to be combined from the comparator image for every row in the reference image. Equations 3.4 (b), 3.5 (b) and 3.6(b) give values from the direction matrix. This value gives the combination of the pixels from the comparator image that is combined horizontally during the row combination and reconstruction process. Both the values from the above equations give the exact pixel-group combinations that were used during cost calculation. This pixel-group represents one base pixel from the reference image that is summed, averaged and replaced to obtain the reconstructed image. Equations 3.4 (c), 3.5 (c) and 3.6(c) are used to calculate and average the combined pixel-groups from the comparator image. Equations 3.4 (d), 3.5 (d) and 3.6(d) are the final calculated and rescaled reconstructed image values that were written back to the registered image matrix.

3.5. SIMPLE REGISTRATION EXAMPLE

A simple registration example is presented to walk through the entire algorithm and have a better understanding of the complete cost function and image registration process as developed and described in the preceding sections.

0.870968	0.862903	0.822581
0.963710	1.000000	0.88306
0.955645	0.911290	0.810484

(a) Normalized reference image

0.620968	0.479839	0.423387	0.350806
0.451613	0.439516	0.350806	0.358871
0.407258	0.358871	0.262097	0.129032
0.354839	0.221774	0.205645	0.080645

(b) Normalized comparator image

Figure 3.11. Normalized reference and comparator images.

Figure 3.11(a) gives the normalized image values of the reference image (3x3 matrix). Figure3.11(b) gives the normalized values of the comparator image (4x4 matrix). Figure 3.12 demonstrates a sample calculation for computing the cost value obtained in comparing the first pixel (1st row and 1st column) from the reference image with all possible legal pixel-groups from the comparator image for both layer-1 and layer-2.

1st layer first row cost $= [\,|0.870968\ - 0.620968| + (1-1)]^2$

$= 0.062500$ (Figure3.9 (a), first row, first column)

2nd layer first row cost $= [((|0.870968-(0.620968+0.451613)/2))|+(2-1)]^2$

$= 1.781364$ (Figure3.9(c), first row, first column)

Figure 3.12. Sample calculation for first row cost computation

Figure 3.13(a) and (b) show the cost and direction matrix values for the first row computation of the image comparison for layer-1. Figure3.13(c) and (d) presents the cost and direction matrices for layer-2 of the first row.

0.0625	1.7438	0.00	0.00
0.00	.02092	1.9370	0.00
0.00	0.00	0.00	2.1596

(a) First row layer-1 cost matrix

1	2	0	0
0	1	1	0
0	0	0	1

(b) First row layer-1 direction matrix

1.7813	11.377	0.00	0.00
0.00	3.7504	13.555	0.00
0.00	0.00	0.00	15.664

(c) First row layer-2 cost matrix

1	2	0	0
0	1	1	0
0	0	0	2

(d) First row layer-2 direction matrix

Figure 3.13. First row layer-1 and 2 cost and direction matrices for the image example

As discussed earlier the dimensions of the cost and direction matrices are defined by the width of the reference and the comparator images. The height of the cost and direction matrix equals the width of the reference image while width of the two matrices equals the width of the comparator image. All the pixel-group combinations for every pixel in the reference image are stored in a separate row of the cost and direction matrix starting from the column corresponding to the column from where the first pixel from the comparator image was compared.

Hence, the entries in the cost and direction matrices run along the diagonal of the matrices. The direction matrix stores the number of pixels combined from the comparator image for every corresponding cost value computed in the cost matrix. All other elements in the cost and direction matrix are made zeros, because those positions of the matrices imply deleted elements of the reference image, which is not done in this dynamic programming implementation.

The cost and direction matrices for the middle rows are similarly computed and are shown in Figure 3.14. However, the cost from the associated previous row are added to find the combined cost incrementally. The cost matrix for the final row was computed by adding the cost from the previous layers and from the previous rows.

0.3247	4.0486	0.00	0.00
0.00	0.7856	4.6632	0.00
0.00	0.00	0.00	5.0714

(a) Second row layer-1 cost matrix

1	2	0	0
0	1	1	0
0	0	0	2

(b) Second row layer-1 direction matrix

2.0910	13.875	0.00	0.00
0.00	4.4710	16.59	0.00
0.00	0.00	0.00	19.23

(c) Second row layer-2 cost matrix

1	2	0	0
0	1	1	0
0	0	0	2

(d) Second row layer-2 direction matrix

Figure 3.14. Second row layer-1 and 2 cost and direction matrices for the image example

Figure 3.15(a) and (b) presents the cost and direction matrix values for the final row first layer, and Figure 3.15(c) and (d) shows the cost matrix value for the final row second layer. The layer with the minimum cost in the final row is considered the winner and this cost is taken to be the final minimum cost for comparing the reference image to the comparator. For this example, layer-1 has the minimum cost of 22.84.

In the cost computation of middle rows, layer costs are computed in different combinations of rows within a layer. The row combination giving the minimum cost was selected to represent that layer. This information regarding which row combinations represent a layer cost for middle rows is given in the vertical direction matrix. Vertical direction matrix for this sample example is shown in Figure 3.16. This matrix along with the direction matrix can precisely backtrack all the pixel-group combinations for all the rows in the comparator image to reconstruct the registered image. The winning layer gives the number of rows combined for the last row cost computation. This value is entered in the rightmost column of the last row of the vertical direction matrix. The number of row combinations for every layer is inserted in each column entry of the vertical direction matrix. Combinations for layer-1 are always made up of one row hence the entries in the first column of the vertical direction matrix are always one.

Layer-2 can have either one or two row combinations for this example and the appropriate number entry is made in the second column representing the row combinations for that row. Now to register this image to the reference size, the rows corresponding to the winning layer are taken. This value is taken from the last column from the last row of the vertical direction matrix. The direction matrix corresponding to this layer is used for the reconstruction (Figure 3.15 (b)).

2.4519	16.655	0.00	0.00
0.00	5.3074	19.840	0.00
0.00	0.00	0.00	22.849*

(a) Final row layer-1 cost matrix

1	2	0	0
0	1	2	0
0	0	0	2

(b) Final row layer-1 direction matrix

2.8040	17.152	0.00	0.00
0.00	5.8925	20.459	0.00
0.00	0.00	0.00	23.4362

(c) Final row layer-2 cost matrix

1	2	0	0
0	1	2	0
0	0	0	2

(d) Final row layer-2 direction matrix

Figure 3.15. Final row layer-1 and 2 cost and direction matrices for the image example

A sample calculation of the reconstruction of the final row of the registered image is presented in Figure 3.17. The direction matrix from the winning layer-1 is selected from Figure 3.15(b).

1	2
1	1
0	1

Figure 3.16. Vertical direction matrix

Since the last value in the direction matrix is 2, the last two elements from the comparator image in Figure 3.11 (b) are combined, averaged and scaled by the maximum comparator image value to be used as the pixel at the last column, last row position in the reconstructed image.

$$R\_img[3][3] = \frac{0.080645 + 0.205645}{2} = 0.143145$$

$$R\_img[3][2] = \frac{0.221774}{1} = 0.221774$$

$$R\_img[3][1] = \frac{0.354839}{1} = 0.354839$$

Figure 3.17. Sample calculation

Next, the direction matrix was backtracked one row up and one column left of the direction matrix. Since the value at this position is one the pixel corresponding to third last column in the last row of the comparator image (Figure 3.11(b)) was used as the pixel to be scaled and replaced in the second column of the last row of the reconstructed image.

Similarly, the direction matrix was backtracked for every row, until the complete image was reconstructed and written back as the registered image. Normalized values of the reconstructed image matrix for the sample example in this section are shown in Figure 3.18.

0.620968	0.479839	0.387096
0.429435	0.399193	0.275201
0.354839	0.221774	0.143145

Figure 3.18 Reconstructed registered image

In the next Section, the image registration technique is evaluated with radioprobe image pairs acquired from the University of Missouri database. Original and registered images are shown and interpretation of the image quality is given based on the cost function values as well as the visual registered images.

4. EXPERIMENTS AND EVALUATION OF THE ALGORITHM

4.1. INTRODUCTION

The image registration technique presented in Section 3 provides two important results for comparing image pairs: 1) registration or alignment between the image pairs and 2) a quantitative evaluation of the similarity between the image pairs. The final cost from the dynamic programming-based registration provides a quantitative value highlighting the similarity between the reference image and the comparator. This final cost value is used as a quality measure for evaluating the image pairs.

The comparator image was registered based on the direction matrix corresponding to the determined cost matrix. In this section, the results are presented by running the image registration technique on 18 normal and tumor image pairs obtained from the University of Missouri Ellis Fischel Cancer Center image database. Image pair examples are also shown to illustrate visually the comparison of the results.

4.2. EXPERIMENTS PERFORMED WITH BASE IMAGE REGISTRATION TECHNIQUE

The image registration technique presented in detail in Section 3 was run over the available 17 image pairs. For the normal and tumor images for each image pair, a region of interest (ROI) was extracted. The ROI refers to the portion of a microarray image where the spots are contained and is outlined in the four corners with sets of marker spots. Figure 4.1 shows an example of a normal and tumor image pair with the black box shown on each image highlighting the ROI. The region of interest (ROI) in the original images was bordered with an unwanted area that does not have any information regarding the spots. For these experiments, the ROI was cropped out using Paint Shop Pro^{TM} and written to a new sub-set image that was used for processing. Figure 4.1 shows an example of a normal and tumor image pair with the black box shown on each image highlighting the ROI.

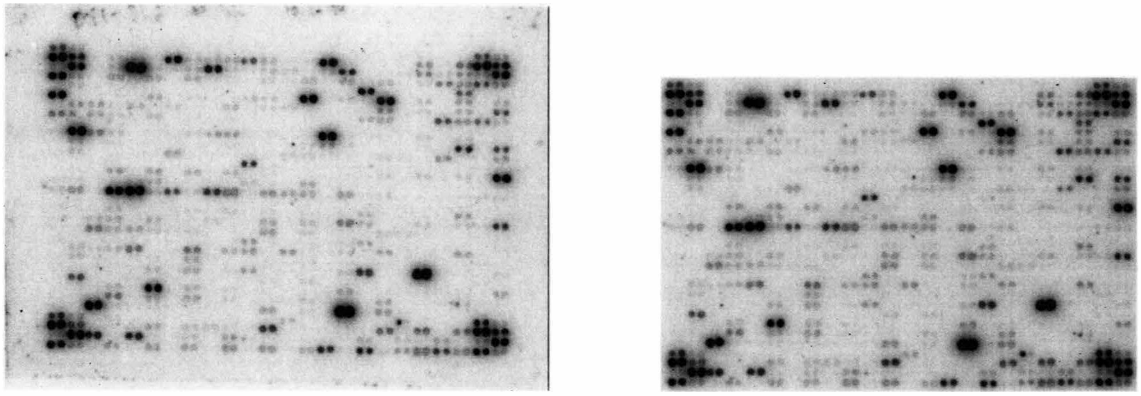
(a) Normal image showing the ROI (b) Tumor image showing the ROI

Figure 4.1. Normal and Tumor images highlighting the Region of Interest (ROI)

To accommodate for tilting that resulted from manual image acquisition using the phosphoimager, the original images were rotated between 0.5-1 degree. An example of such an image before and after the cutout is shown in Figure 4.2.

All the 17 image pairs were cutout as described above to have only the ROI. These image pairs were cropped using three row and column difference combinations between the reference and comparator images to produce three image sets:

1) the reference and comparator image ROIs are equal size (set-1)
2) the reference image ROI is 2 rows and 4 columns smaller than the comparator image ROI (set-2) and
3) the reference image ROI is 6 rows and 6 columns smaller than the comparator image ROI (set-3).



(a) Original image with the unwanted bordered area around ROI

(b) Final image after cutting out ROI

Figure 4.2. Image example showing region of interest (ROI)

Reference and comparator images were randomly selected from the dataset. However, the reference and the comparator images were kept consistent in all the three datasets. All the three sets of 17 image pairs each were run on the dynamic-programming based algorithm. The resulting final cost-per-pixel with respect to the reference image is shown in Table 4.1. From the final cost values in Table 4.1, the goal is to translate the final cost value to a quality measure for the image pairs. As explained earlier, images of the same row and column differences for each set were taken for these experiments so that a fair evaluation of the cost could be carried out. A high cost was attributed to image pairs that have either some or all of the characteristics mentioned below in varying magnitudes:

(a) Grainy background of either both or any one of the images,

(b) No correlation between the spots from one image to the other,

(c) A relatively large shift both vertically and horizontally between the spots,

(d) Relatively high disparity between the spots intensities and

(e) A lot of artifacts in any of the image under consideration

A low cost of matching pairs was attributed either due to the absence of the attributes listed above or due to a lesser degree in magnitude of these attributes expressed in the images.

The cost value largely depends on the relative difference between the image pairs processed. These results clearly show that the cost value linearly determines the similarity between the processed image pairs. In set-1 from Table 1.1, image pairs are taken with equal sizes. Images in this set are arranged in decreasing order of their cost value. These cost values determine the dissimilarity between the two images and hence their quality in terms of the factors discussed earlier. Cost-per-pixel value with respect to the reference image gives a good measure of the similarity or dissimilarity of the images compared. The final cost value is divided by the number of pixels from the reference image. Identical images of the same size, that are similar will give a cost per pixel value of 0.0 while the image pairs that are largely dissimilar will give a cost per pixel value closer to 1.0. Images with the highest row and column difference give a very high cost while those with less row and column difference give a smaller cost value. This can be observed in the results from set-2 and set-3 from Table 4.1. From Table 4.1, it is seen that image pairs processed from set-3 having 6 rows and 6 columns difference have on an average high cost as compared with set-2 with 2 rows and 4 columns difference. Set-1 having no row or column difference has the least cost. With respect to the dataset used in this research, the amount of shift in the spots taking place between the images was directly proportional to the high cost value.

Table 4.1. Listing of the cost values for the dataset (set-1, set-2 and set-3)

Sr.No	Image pairs	Set-1	Set-2	Set-3
1	19n-65.pgm, 19t-65.pgm	0.0287	0.0395	0.0521
2	10n-65.pgm, 10t-65.pgm	0.0192	0.0314	0.0487
3	11n-65.pgm,11t-65.pgm	0.0162	0.0270	0.0416
4	22t(65c).pgm, 22n(65c).pgm	0.0159	0.0288	0.0438
5	55t-(65c).pgm,55n(65c).pgm	0.0158	0.0262	0.0439
6	23t-65.pgm,23n-65.pgm	0.0111	0.0247	0.0424
7	34n-65.pgm,34t-65.pgm	0.0104	0.0221	0.0386
8	12t(65c).pgm,12n(65c).pgm	0.0101	0.0218	0.0375
9	39t-65.pgm, 39n-65.pgm	0.0098	0.0193	0.0383
10	58t-65c.pgm,58n-65c.pgm	0.0087	0.0180	0.0343
11	norm2a.pgm, tum2a.pgm	0.0078	0.02128	0.0380
12	20n-65.pgm,20t-65.pgm	0.00767	0.01896	0.0346
13	37t-65c.pgm, 37n-65c.pgm	0.00763	0.02064	0.0372
14	2t-65.pgm,2n-65.pgm	0.00704	0.01728	0.0321
15	26t(65c).pgm,26n(65c).pgm	0.00656	0.01932	0.0355
16	59t-65c.pgm,59n-65c.pgm	0.00523	0.01834	0.0335
17	15n(65c).pgm,15t(65c).pgm	0.00445	0.02189	0.0292

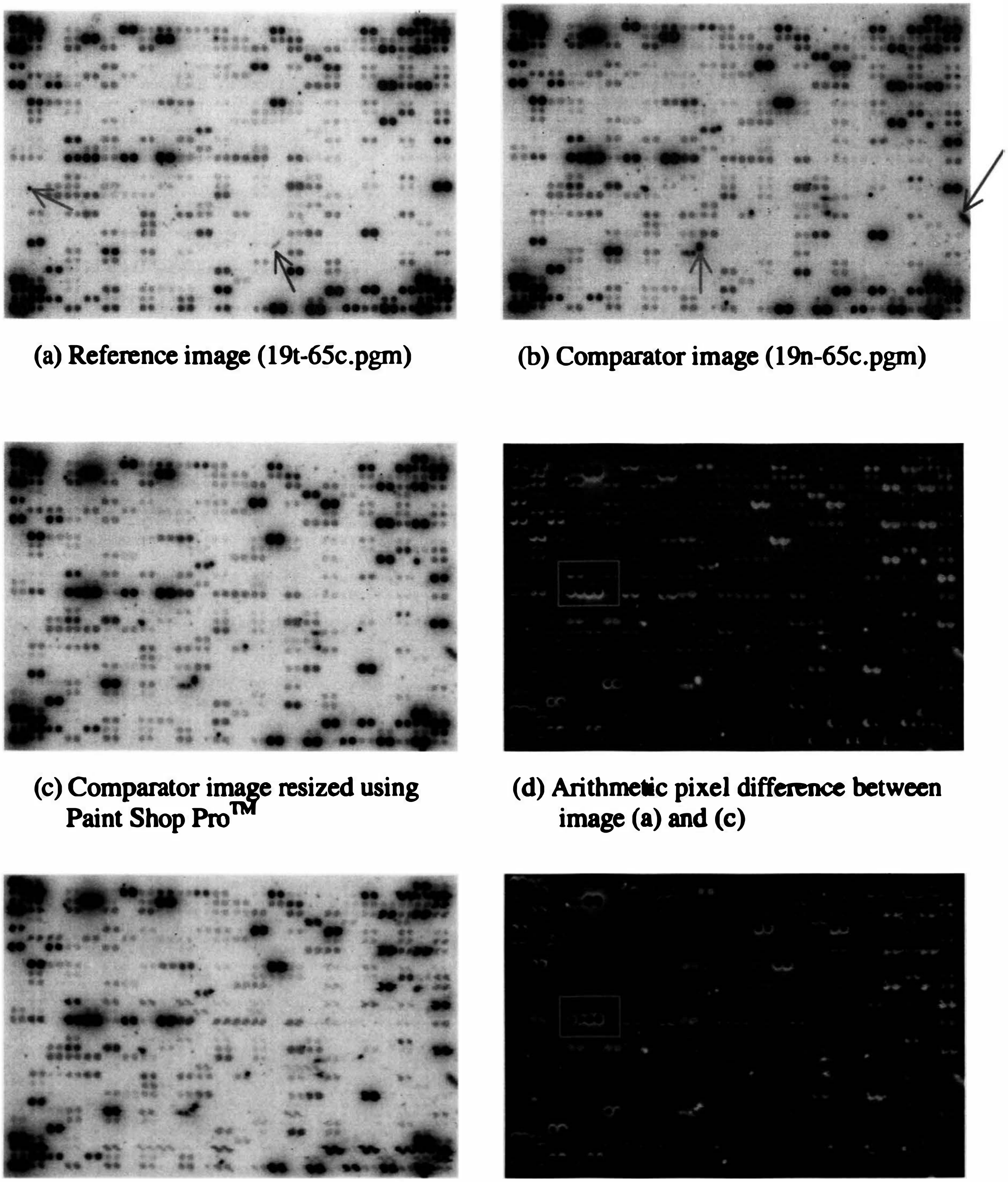
In set-2 and set-3 it is observed that the image pairs with high cost have the greatest shift associated with them during registration. This indicates that the images had their spots misaligned to a greater extent as compared with the image pairs having a lower cost. Table 4.1 shows that the image pair 1 has high cost and hence the misalignment between the spots in these images during registration is greater. Similarly, image pair 17 gives a low cost implying that there is comparatively a lesser degree of misalignment between the spots in this image pair.

Three specific examples of the image pairs are presented and discussed in the next section. These image pairs represent cases of high cost, medium cost and low cost as obtained from the dataset under consideration and are experimental results from set-1 in Table 4.1

4.3 SPECIFIC EXAMPLE OF AN IMAGE PAIR

In order to have a better understanding of the cost value interpretation, specific examples are given in this section. The images seen visually help in validating the interpretation of their cost value obtained by comparing them. The pixel-wise difference was computed between the reference image and the comparator image in two different cases using Paint Shop Pro^{TM}. In the first case, the comparator image is resized to the reference image size using Paint Shop Pro^{TM}. In the second case, the comparator image is registered to the size of the reference image using dynamic programming-based image registration technique.

4.3.1 Worst Case Image Example. As seen in Table 4.1 the poorest quality image pair was the one with the highest cost. The image pair 19t-65.pgm and 19n-65.pgm yielded the highest cost of 10639.74. Figure 4.3 shows the images for this example.



(a) Reference image (19t-65c.pgm)

(b) Comparator image (19n-65c.pgm)

(c) Comparator image resized using Paint Shop Pro™

(d) Arithmetic pixel difference between image (a) and (c)

(e) Registered comparator image using image registration algorithm

(f) Arithmetic pixel difference between the image (a) and (e)

Figure 4.3. Example of worst-case quality and image registration from the dataset

The original images are shown in Figure 4.3(a) and 4.3(b) as reference and comparator images, respectively. The comparator image was resized to the size of the reference image using Paint Shop Pro^{TM} as shown in Figure 4.3(c).

The arithmetic difference taken between this resized image and the reference image shows how closely the two images are aligned. This image is shown in Figure 4.3(d). The registered image is shown in Figure 4.3(e), and the arithmetic difference of the registered image and the reference image is shown in Figure 4.3(f).

For a preliminary evaluation of the image pairs, the average gray level of each image was used as a general indicator of the similarity between the two images. For the image pairs in this example, the average gray level found for the tumor (19t-65.pgm) and normal (19n-65.pgm) images is 166 and 173, respectively. The difference in the average values can be attributed mainly to the contrasting backgrounds of the two images. Visually, the images have different grainy backgrounds. Besides seeing the arithmetic difference before registration in Figure 4.3(d), it was noticed that the spots between the two images are greatly misaligned. Furthermore, the image was somewhat noisy and other artifacts contribute to the higher overall cost. Some artifacts seen in the original images are highlighted using a blue arrow while random noise is highlighted using a red arrow in Figure 4.3(a) and Figure 4.3(b).

Figure 4.3(e) shows the registered comparator image after being processed using the dynamic programming-based image registration algorithm. Legal groups of pixels were optimally selected, averaged and replaced by their average during the registration process based on the minimum cost obtained from their different combinations. Depending on which groups of pixels along the layered row from the comparator were combined, the entire row from the comparator image shifts left, right or towards the center of the registered image. The type of shift that takes place depends on various factors like artifacts along that row, graininess of the image background or a shift between the corresponding spots. This makes the shifts in the rows and in the columns nonlinear. Some of the pixel-groupings within a row or group of rows that form the entire spots shift to the right and some shift to the left, while others remain locally stationary, thus giving the spots a distorted look. A different approach is required for image registration to eliminate distortion of the registered image. A cascaded approach of dynamic programming and registration is presented in Section 5 where the distortion discussed above is eliminated.

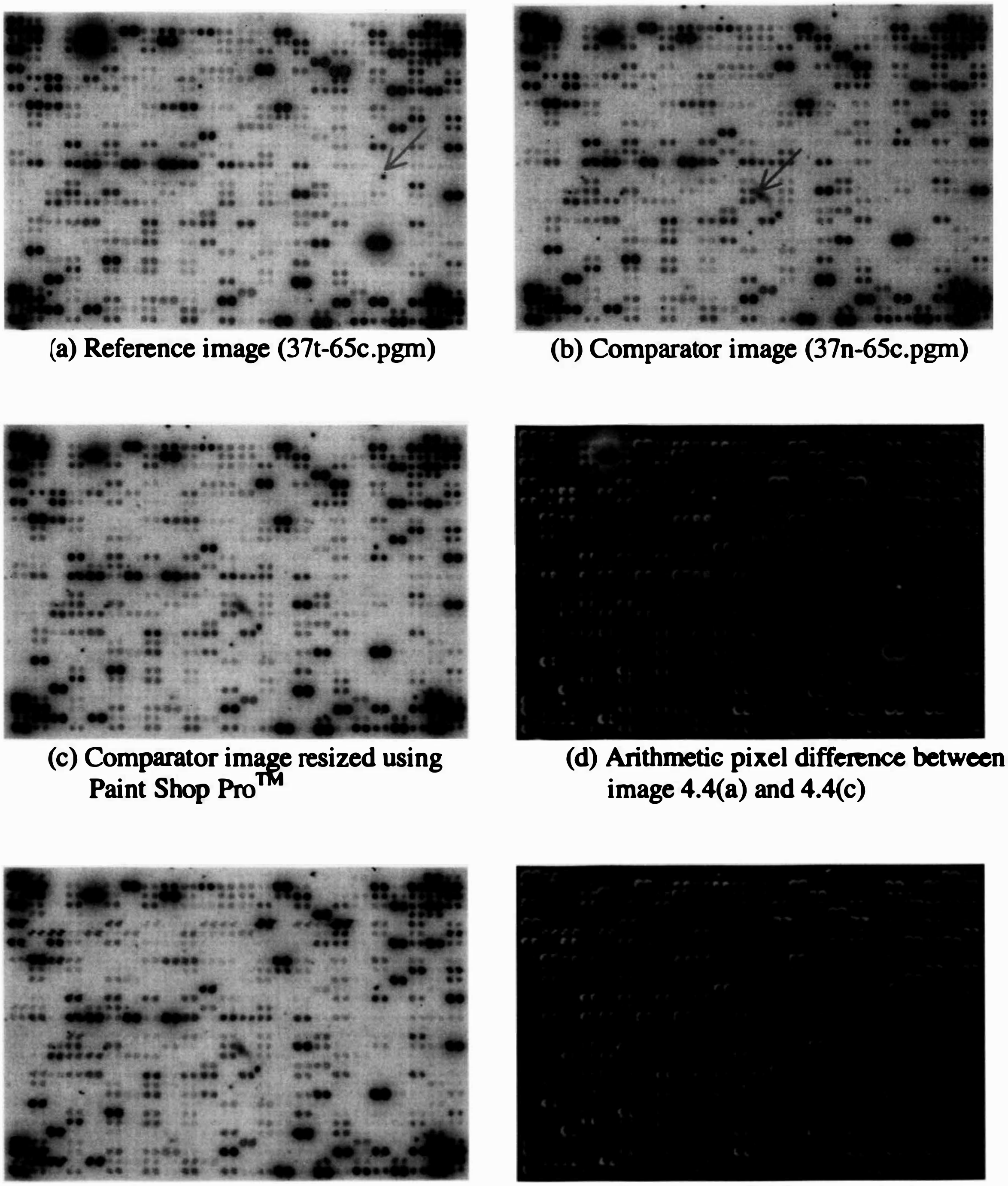
When the arithmetic pixel difference of the image pairs is compared before and after registration, it is noticed that the registered image has significantly shifted the misaligned spots. Figure 4.3(d) shows the pixel arithmetic difference before registration. Most of the spots are seen not aligned with each other. A group of such spots is shown highlighted in a red box in figure 4.3(d). The misalignment is seen here before registration. The same group of spots is highlighted again with a red box in the arithmetic pixel difference after registration in Figure 4.3(f). This image clearly shows most of the spots better aligned as compared to the previous image pixel-difference before registration.

4.3.2 Average Case Image Example. From Table 4.1 an image example is presented from set-3 that has a medium cost. An average cost is a direct indicator of the relative medium quality of the image pair in the dataset under consideration. The image pair 37t-65.pgm and 37n-65.pgm is chosen for illustration of the medium cost case. The cost obtained for this image pair is 7565.07. The reference and the comparator images representing medium quality are shown in Figure 4.4(a) and Figure 4.3(b).

The comparator image was resized to the size of the reference image using Paint Shop Pro^{TM} as shown in Figure 4.4 (c). The arithmetic pixel difference taken between this resized image and the reference image shows how closely the two images are aligned (Figure 4.4(d)). The registered image comparator image is shown in Figure 4.4(e). The arithmetic pixel difference of the registered image and the reference image is shown in Figure 4.4(f).

The average gray value calculated for the tumor (37t-65.pgm) and normal (37n-65.pgm) images are the same (149). Although the average gray values for these images is identical there is some graininess seen in the backgrounds of these images as well. There are also a significant number of artifacts in these images. In Figure 4.4(a) and 4.4(b) the blue arrow shows one of the prominent artifacts while the red arrow shows some random noise. Another major contributing factor to the relatively high cost is that the two images have contrasting corresponding spots in terms of both intensity and size. Besides seeing the arithmetic difference before registration in Figure 4.4(d), it is noticed that the spots between the two images have some noticeable misalignment that is a major contributor of the cost.

When the arithmetic difference of the images are compared before and after registration, it is noticed that the registered image has significantly shifted the misaligned spots and many of the spots are seen to be centered with each other as seen in Figure 4.4(f) as opposed to Figure 4.4(d).



(a) Reference image (37t-65c.pgm)

(b) Comparator image (37n-65c.pgm)

(c) Comparator image resized using Paint Shop Pro™

(d) Arithmetic pixel difference between image 4.4(a) and 4.4(c)

(e) Registered comparator image using the image registration algorithm

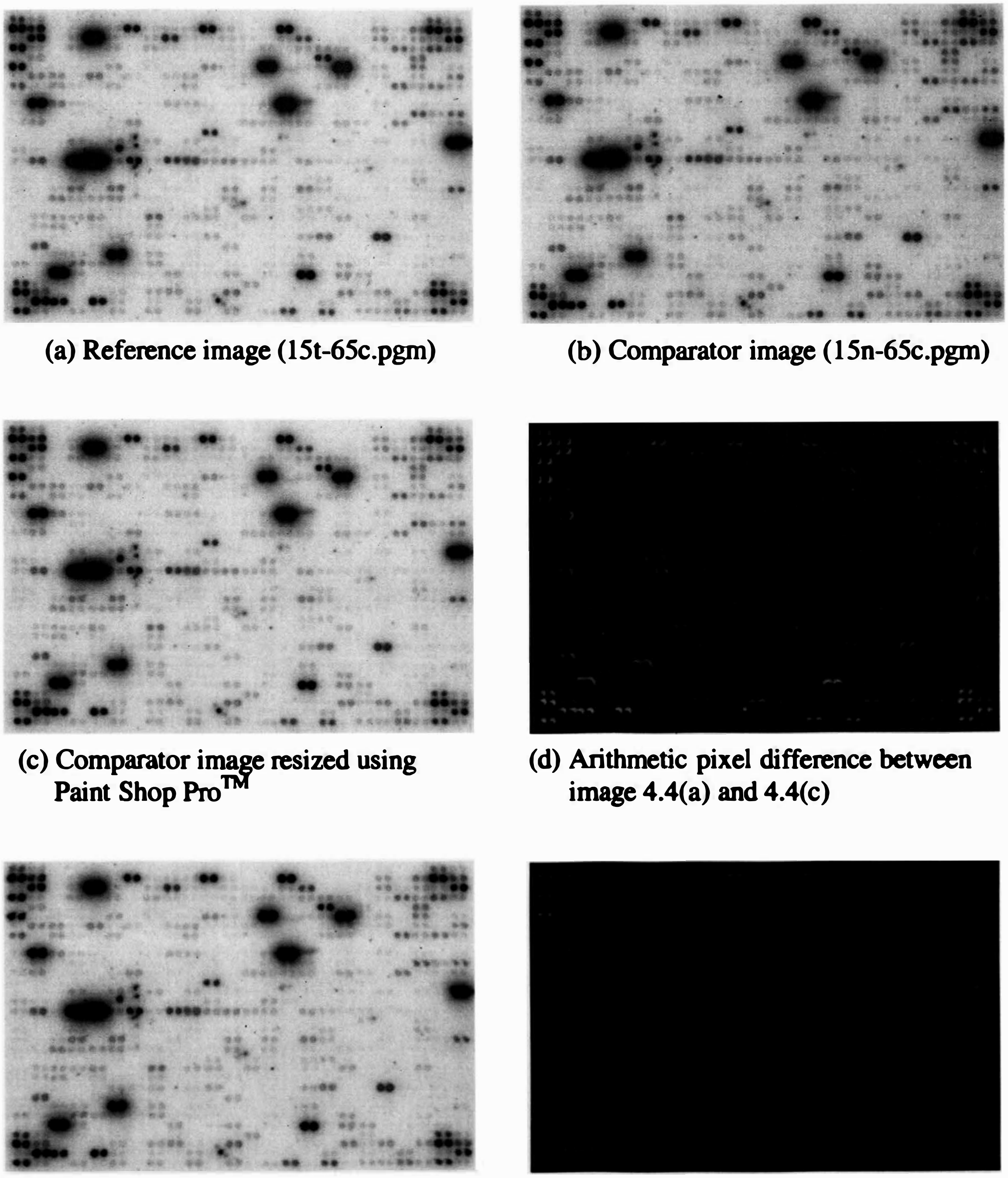
(f) Arithmetic pixel difference between images 4.4(a) and 4.4(f)

Figure 4.4. Example of medium-case quality and image registration from the dataset

It is seen that there is a better image registration taken place as compared to the worst-case image pair. This was because the spots in the average-case image pair are relatively aligned better and the background is relatively smooother and less grainy as compared to the image pair discussed in the previous section for the worst-case image pair.

4.3.3. Best Case Image Example. From Table 4.1 the best case is taken as the image pair with the lowest cost in the entire dataset. The tumor (15t(65).pgm) and normal (15n(65).pgm) image pair with the lowest cost for registration yielded a cost of 5874.89. The reference and comparator images for the best-case example are given in Figure 4.5(a) and Figure 4.5(b) respectively. The comparator image was resized to the size of the reference image using Paint Shop Pro^{TM} and is shown in Figure 4.5 (c). The arithmetic pixel difference taken between this resized image and the reference image in Figure 4.5(d) shows how closely the spots in two images are aligned. The registered image is shown in Figure 4.5(e), while the arithmetic pixel difference of the registered image and the reference image is shown in Figure 4.5(f).

The average gray level for the tumor and normal images is 184 for both images under study. The most striking observation seen was in the arithmetic pixel difference before registration as seen in Figure 4.5(d). The spots in the images are closely aligned already. Unlike the worst-case and the average-case examples discussed in the previous sections there is noticeably less misalignment seen in this example. The backgrounds of both the images are similar, smooth and little graininess was observed. There are also no major artifacts in either image, and the spots are closely matched with each other in terms of both their intensity and size.



(a) Reference image (15t-65c.pgm)

(b) Comparator image (15n-65c.pgm)

(c) Comparator image resized using Paint Shop Pro™

(d) Arithmetic pixel difference between image 4.4(a) and 4.4(c)

(e) Registered comparator image using the image registration algorithm

f) Arithmetic pixel difference between image 4.4(a) and 4.4(f)

Figure 4.5. Example of best-case quality and image registration from the dataset

All these favorable features of this image pair best explain the lowest cost obtained in this data set and thus the relatively high quality image pair from the dataset under study. Arithmetic pixel difference between the image pairs is shown in Figure 4.5(f). A near-perfect alignment of the spots is seen. This correctly explains the lowest cost obtained for this image pair.

In general, from the above three examples it was observed that the cost computed using the dynamic programming based approach directly relates to the image quality. However, this cost is relative and specific to a particular dataset. High cost in a given dataset will represent bad quality images while low cost in the same dataset will relate to the good quality images. Main attributes that represent poor quality images in a microarray image pair are average pixel difference between the pair, graininess of the background, and similarity in average pixel values of their backgrounds. Artifacts in any of the images, random noise and spots that are not related to each other are other factors that could contribute to the cost significantly. It was observed that graininess in the image backgrounds and the difference between background intensity level are the factors that contribute most to the higher cost and hence the bad quality of the image. Other factors that contribute to the cost are the relative shift between the corresponding spots in the two images, the intensity level and size of the spots. Relative shift between the spots was rectified using image registration and this was the very purpose in developing this algorithm. Difference in intensity level between the spots can be used to do feature extraction and is important to find the genetic expressions that could represent cancer images [21].

The registered images in all the three examples presented above helps in aligning the spots that occur in pairs in both the normal and the tumor images. The purpose of using the dynamic programming approach for cost evaluation was to simultaneously make it possible to register the two images. The backward solution component of the dynamic programming-based image registration algorithm provided an optimal image registration. The registration process in the comparator image aligned the corresponding spots between the two images. Alignment of spots are seen and highlighted in image examples showing the arithmetic pixel difference between the registered and the reference images in Figure 4.3.

The registered images have a limitation in terms of streaks that are seen. However, it was observed that there is a correlation between the intensity of these streaks and the cost value. Registered images with major distortion generally have a high cost indicative of poor quality image pairs. Registered images where the distortion level was very minor have low cost value indicative of good quality image pairs. Hence, visual distortions in the registered images are also a reflection of the quality level of the image pairs.

4.4. LIMITATION OF THIS METHOD

Good registration in terms of spot alignment was seen to have taken place in all the image pairs. The value of the cost reflected the true quality measure of the images as well. However, a limitation was observed in the registered image using this algorithm. The registered image had some streaks in the reconstructed spots and the spots are not always reconstructed smoothly but some distortion was seen to varying degrees. One of the reasons was that the pixel combinations do not behave consistently all along the row comparisons and hence in some cases the pixels are pulled to the right of the image while in other cases it is pulled to the left for the row combinations.

Section 5 gives an algorithm that successfully reconstructs and eliminates the streaks in the registered images. A cascaded dynamic programming and registration approach was used for the same.

5. CASCADED APPROACH FOR IMAGE REGISTRATION

5.1. INTRODUCTION

The dynamic programming-based image registration technique presented in Section 3 provided the capability to align corresponding spots in the paired images and gave a quantitative measure of the quality of the image pair. In Section 4 the image registration algorithm was applied to 17 image pairs in a local database. The experimental results show that the cost measure yielded a reasonable indicator of image quality. However, the registration of the paired images resulted in streak marks, distorting the reconstructed comparator image. As seen in the registered image examples presented in Section 4, for the worst and average cases, the registered images are distorted and the spots have streak marks that can be easily visible. The main reason for image registration is the corresponding spot-to-spot alignment in the tumor and normal images. Feature extraction from the corresponding spots needs to be done to facilitate comparisons that can be explored in genetic databases. To this point, feature extraction is mainly done using the intensity level between the corresponding spots [21]. The distorted images obtained from the baseline algorithm in Section 3 certainly do not facilitate an accurate intensity measure. In this section a cascaded approach for image registration is presented that builds off the base approach presented in Section 3.

5.2. DESCRIPTION OF THE ALGORITHM

In the base algorithm as described and presented in Section 3, a layered approach was used to compare single rows from the reference image to multiple rows from the comparator image. Every element from the reference image was compared with a defined legal group of pixels from the comparator image. Legal groups of pixels were optimally selected, averaged and replaced by their average during the registration process based on the minimum cost obtained from their different combinations. Depending on which groups of pixels along the layered row from the comparator are combined, the entire row from the comparator image shifts left, right or towards the center of the registered image. The type of shift that takes place depends on various factors like artifacts along that row, graininess of the background or a shift between the corresponding spots. This makes the shifts in the rows and in the columns nonlinear. Because of the above reasons, some of the pixel-groupings within a row or group of rows that form the spots shift to the right, or to the left, while others remain locally stationary, thus giving the spots a distorted look after reconstruction.

The cascaded approach solves this problem in such a way that the distortion is minimized and practically eliminated. The dynamic programming methodology developed in Section 3 remains the same. Pseudo-code from the cascaded dynamic programming approach for image registration is presented in Figure 5.1. Primary difference in the cascaded approach over the base algorithm in Section 3 is that the registration of the reference image is carried out in two stages (hence the name cascaded).

In the first stage, the reference and the comparator images are selected and run through the dynamic programming (base DP)-based algorithm shown in Figure3.2 from Section 3. The cost obtained during the first stage of base DP reflects the quality measure of the two images as discussed in Sections 3. However, the registration of the comparator image was done using a different technique. Registration was done only in the vertical direction along the rows.

```
Read normal and tumor images
Determine reference and comparator images
Initialize cost and direction matrices
Perform base dynamic programming-based approach for image
    registration, registering the image vertically along its height only
Rotate registered image 90 degrees clockwise and make it as the new
    comparator image Rotate the reference image 90 degrees
    clockwise to have orientation same as reference image
Perform base dynamic programming-based approach for image
    registration, registering the image vertically along its height only
Reorient registered image for final registered image
```

Figure 5.1 Pseudo-code for cascaded approach

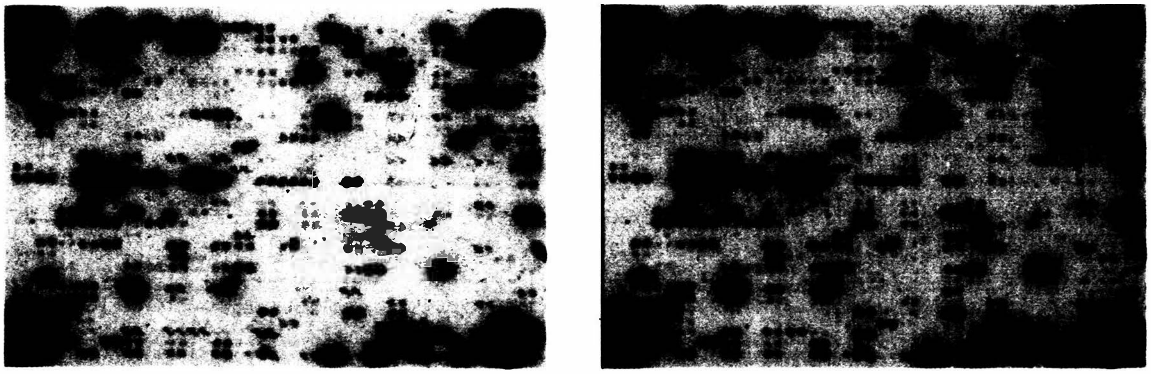
Only the layer combinations of the rows from the comparator image that form the best match with the reference row were combined in the vertical direction and replaced in the registered image during registration. No combinations of pixels were done along the columns (horizontally). It may be noted that entire rows were used during reconstruction at a time. The registered image obtained after the first stage of the algorithm had its height equal to that of the reference image while the width is not changed. The image was hence registered only in the vertical direction. In the second stage of the cascaded algorithm, the registered image obtained from the first stage was taken as the comparator for this stage. This registered image was rotated 90 degrees clockwise. The idea was to have the width of the image registered with the reference image width all along the columns. Hence, there was a need to rotate these images so that the columns are effectively combined just as rows were, during the first stage of the registration. Similarly, the reference image was rotated by 90 degrees clockwise to have this image in its correct orientation in relation with the comparator image. Both images are then processed again using the base DP algorithm. The cost value obtained from the second stage of this algorithm was neglected since this stage does not use the original size of the input images. Again, the registration of the comparator image at this stage was done in the vertical direction only, as explained for the first stage. The registered image obtained was the final registered image and was of the size equal to the reference image. The registered image was then rotated anti-clockwise by 90 degrees to have the image in its correct original orientation.

Registered image examples of the output registered image using the new approach are compared with those obtained in Section 4 in the next section.

5.3. SPECIFIC EXAMPLE OF AN IMAGE PAIR

Image examples of registered images obtained using the cascaded approach are presented in this section. Registered image output results from Section 4 are reproduced here to compare and contrast the improvement obtained after using the cascaded approach. Exact same image pairs used as inputs for baseline algorithm were used in the cascaded approach. Image pair examples from Figure 4.3(a) and Figure 4.3(b) from Section 4 were used as inputs to the cascaded approach algorithm. Figure 5.2(a) shows the worst case registered image obtained using the baseline algorithm; this figure is reproduced from Figure 4.3(e) while Figure 5.2(b) shows the registered image obtained using the cascaded approach.

The registered image obtained using the cascaded approach in Figure 5.2(b) is smooth and clear. No distortion, streaks or artifacts are seen in the image compared to that obtained with the base DP approach shown in Figure 5.2(a). Major improvements in the reconstructed spots in both these images are highlighted in red boxes. Clearly, the cascaded approach produces better results in terms of registered image quality.



(a) Using the base dynamic programming algorithm.

(b) Using the cascaded image registration approach

Figure 5.2. Registered images using base and cascaded approaches

Since the base DP algorithm and registration is performed twice, the processing time taken for this approach nearly doubles. However, since the registered images will be further used for feature extraction from the spots it is essential that the original nature of the spots be maintained. The cascaded approach achieves this goal without any compromise with respect to the alignment of the spots.

An inherent limitation of both the base DP approach as well as the cascaded image registration approach is the relative size of the reference image. The manner in which these algorithms are setup has a primary requirement that reference image be smaller in terms of its both height and width. It is sometimes possible that there are cases where the height or the width of the reference image is greater than the comparator image. Such an image pair is considered hybrid and requires a different approach for image comparison and registration.

6. CONCLUSIONS AND RECOMMENDATION FOR FUTURE WORK

6.1. SUMMARY

The focus of this research work was on the formulation of a technique to evaluate the quality and perform the registration of radio-probe microarray images. The algorithm was developed to evaluate the quality measure of radioprobe microarray image pairs. Furthermore, the registration algorithm was used to align the DNA spots from the images to match with each other, a primary requirement for feature extraction to be done on the spots using their intensity measure [21]. The cost value results obtained from the dynamic programming based image comparison algorithm gives a good measure of the image quality. Secondly, the image registration algorithm aligns corresponding spots in the two images to a fair degree of accuracy. The cost function from the algorithm gives a true measure of determining how different or how similar the two images compared are. If both the images are identical, the cost value of comparing the two images will be zero. If the two images are not identical and not similar to each other, then the cost value will reflect the difference with a high cost value. Difference in image sizes gives a still higher cost since a cost penalty is added to the base cost as per the distance-measure rules used in the algorithm.

However, there are some limitations on how well the algorithm does in terms of registration of the images. Registration much depends on the relative sizes of the image pairs. For better alignment of the spots to take place, there should be a significant difference in terms of rows and columns between the two images. This is required so that the registration algorithm has some rows and columns to perform the required shift in the spots to align them. No registration will be done if both the images are of equal size. A dynamic programming-based approach (DP) was used as the basis for this algorithm development. The algorithm was built as an extension of the one-dimensional DP approach by Stanley et. al. [6]. The research work met the objective of providing a new approach for conducting 2-D image registration of images related to any set of images that can be extended easily to other types of paired images besides radioprobed microarray images. Furthermore, the modified cascaded approach presented in Section 5 yields a visually viable form of image registration

The algorithm for image comparison and registration was developed and experiments were performed to validate the image registration capability and quantitative quality measure evaluation. Experiments were performed with an 17 image pair data set obtained from the University of Missouri, Ellis Fischel Cancer Center image database.

From the base and cascaded versions of the image registration technique presented, there are two obstacles from the implementation perspective. First, from the mathematical formulation for the base DP algorithm shown in Figure 3.2-3.5, the image registration technique is quite computationally intensive. No time complexity measures were taken into consideration while implementing the algorithm. However, optimization techniques can be introduced to address this area and speedup the execution time. Second, maintaining the cost and direction matrices required significant memory. A limitation that is faced in the present implementation is regarding the images sizes. Memory requirements increase significantly with the increase in the row and column differences between the images. Hence, images with large difference between their rows or columns will require high memory requirements. Some space considerations were taken into account; however there could be more improvement in this area as well for further space complexity efficiency and to accommodate images with large row and column differences.

Another limitation in this approach is that the algorithm developed is currently restricted to image pairs where the reference image is smaller in both width and height with respect to the comparator image. The algorithm fails for a hybrid case where either the height or the width of the reference image is greater than the comparator image. The hybrid case needs a different approach as compared to the baseline algorithm. In the baseline algorithm, every single element was taken from the reference image and compared to the legal group of pixels for every row. Since there are less reference elements, an optimization with respect to the comparator elements was possible. However, if the reference image rows or columns are greater than the comparator image, there is the problem of combining a pre-defined group of reference image. Thus, either substitution will need to be utilized in the comparator image or the reference image or have both the reference and the comparator images registered using a cascaded double registration. In any case, the comparison or registration process will not be simple and straightforward.

6.2. RECOMMENDATION FOR FUTURE APPLICATIONS

The dynamic programming based image comparison and registration approach developed in this research can be widely applied to different applications. Medical images like x-rays, CT-scans and MRI images taken from the same patient over the same region of interest can be compared to predict the overall progress of the treatment. Specifically image analysis can be done based on the cost value obtained after comparing such image pairs to see subtle improvements, or deterioration of the case under treatment.

Progress of diseases like osteoporosis cannot be detected by naked eye observation in MRI images taken over a relatively small span of time, especially because these diseases respond to treatment rather slowly. However, the images can be benchmarked using the dynamic programming based comparison approach to give the percentage of change taken place in the progress of the disease.

Other areas where this algorithm can be used are to detect changes in satellite images taken over a region to detect enemy movement or intrusion. So, the image comparison technique can be used in satellite images that study rapid changing weather patterns etc. Space probes could be detected from photographs that are hard to be observed with naked eye observations. Image detection software could be developed from the DP image comparison approach as security or surveillance to areas unmanned by human presence.

BIBLIOGRAPHY

1. Collins FS, et al: New Goals for the U.S. Human Genome Project: 1998-2003.

2. Huang TH, et al: Identification of DNA methylation markers for human breast carcinomas using the methylation-sensitive restriction fingerprinting technique. Cancer Research 57:1030, 1997.

3. Huang TH, et al: Methylation profiling of CpG islands in human breast cancer cells. Human Molecular Genetics, 1999.

4. DNA Methylation Society, http://dnamethsoc.server101.com

5. National Institute of Health official website http://www.nih.gov

6. Stanley Joe et. al: Data-Driven Homologue Matching for Chromosome Identification. IEEE transactions on medical imaging, Vol 17,N0. 3, 1998.

7. *"Time Warps, String Edits and Macromolecules: The theory and practice of sequence comparisons"* - David Sankoff & Joseph B Krushal

8. Muramatsn, Shoji. Kobayashi, Yoshiki. Title : Image pattern search method based on DP matching for detecting accurate pattern positions Systems and Computers in Japan v29 n 4, p22-32, Apr 1998

9. Makoto Hirosawa, Yasushi Totoki, Masaki Hoshinda and Masato Ishikawa Comphrehensive study on iterative algorithms of multiple sequence alignment. 1995.

10. Gusfield,D. Efficient methods for multiple sequence alignment with guaranteed error bounds. Bull. Math. Biol.,55,141-154,1993.

11. Miller,W. Building multiple alignment from pairwise alignments. Comput. Applic. Biosci., 9, 169-176,1993.

12. Chan, S.C., Wong,A.K.C and Chiu,D.K.Y. A survey of multiple sequence comparison methods. Bull math. Biol., 54,563-598,1992.

13. Ching Zang and Andrew K.C Wong [6] Title : A Genetic algorithm for multiple molecular sequence alignment. Department of Systems Design Engineering, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

14. Althof, R.J; Wind , M.G.J; Dobbins , J.T. " Rapid and automatic image registration algorithm with subpixel accuracy" IEEE transactions on Medical Imaging

15. Szeliski, Richard; Coughlan, James " Spline based image registration" International journal of Computer Vision.

16. Fritsch, Danial S.; Pzer, Stephen M.; Morse, Bryan S.; Eberly, David H.; Liu, Alan " Multiscale medial axis and its applications in image registration

17. Leszczynski K. W.; loose S. Boyko S. "An image registration scheme applied to verification of radiation therepy" British Journal of Radiology

18. David K Smith and Pass Maths "Dynamic Programming-an Introduction" http://plus.maths.org/

19. R.A Wagner and M.J Fischer. " The string to string correction problem." J. Acm vol 21. pp 168-173,1974.

20. Michael G Thompson & Erik Granum "Dynamic Programming inference of markov Networks from Finite sets of Sample Strings. IEEE Transactions Vol. PAMI-8 No. 4 July,1986.

21. "A projections based approach to microarray image spot segmentation" Masters thesis by Maheshwar Gattupalli

22. "*Fundamentals of Digital image Processing*" - Anil K. Jain

23. "*Digital Image Processing*" - Rafael C. Gonsalez

24. "*Dynamic Optimization*" - Arthur E. Bryson, Jr.

25. "*Dynamic Programming and Markov Processes*"- Ronald A. Howard

26. "*Dynamic Programming and Partial Differential Equations*" - Edward Angel and Richard Bellman

VITA

Sunil Bosco Rodrigues was born on the 9th of February 1975 in Panjim, Goa-India. He attended Goa University's College of Engineering where he earned his Bachelor of Engineering degree in Electrical Engineering in June 1997 with highest honors. He joined the Master of Science program in Computer Engineering at the University of Missouri Rolla in January 2000. He graduated from UMR in the Summer of 2001. He is a member of IEEE and Sigma Xi honor society.