AUTOMATIC IMAGE REGISTRATION BY USING MULTI-VARIATE SPLINE FUNCTIONS

by

PENG ZHENG

A THESIS

Submitted in partial fulfillment of the requirements
for the degree of PH.D. in the
Interdisciplinary Applied Mathematics and Mathematical Physics
of Delaware State University

DOVER, DELAWARE
May 2017

This thesis is approved by the following members of the Final Oral Review Committee:

Dr. Xiquan Shi, Committee Chairperson, Department of Mathematical Sciences, Delaware State University
Dr. Guoping Zhang, Committee Member, Department of Mathematical Sciences, Morgan State University
Dr. Zhongyan Lin, Committee Member, Department of Computer and Informational Sciences, Delaware State University
Dr. Sokratis Makrogiannis, Committee Member, Department of Mathematical Sciences, Delaware State University
Dr. Jinjie Liu, Committee Member, Department of Mathematical Sciences, Delaware State University

COPYRIGHT

DEDICATION

I dedicate my dissertation work to my family and my friends. A special feeling of gratitude to my loving parents, Shide Zheng and Shulan Shi whose words of encouragement and push for tenacity ring in my ears. My sisters Keni Zheng have never left my side.

I dedicate this work and give special thanks to my advisor Dr. Xiquan Shi, my friends Ms. Ellen Carr, Penglong Xu and my other friends as well. All of you have been my best cheerleaders.

ACKNOWLEDGEMENTS

I would like to thank every person who helped me throughout my masters work. I gratefully acknowledge advisor Dr. Xiquan Shi for his advice and guidance. He patiently led me back to the correct tracks when I was struggling on the wrong aspects of my research and he also gave me freedom to work on a flexible schedule. Finally, with my advisor's intelligent mathematics framework as the backbone, this article was able to be advent.

I wish to thank Dr. Guoping Zhang, Dr. Zhongyan Lin, Dr. Makrogiannis and Dr. Jinjie Liu for being my committee members, who provided inspirations, support and time commitment. My studies would not be possible without the administration and financial support of the Center For Advanced Algorithms and the Department of Mathematical Sciences. Endless thanks to Dr. Fengsan Liu, head of Center For Advanced Algorithms and Dr. Nicola Edwards, the chair of Department of Mathematical Sciences.

Finally, I would like to thank my family members for their remote care and high expectations. They are the major momentum for me to continue my studies.

ABSTRACT

Image registration is an image processing to transform different sets of data in which one has an overlap with another into one coordinate system. Usually, the data is multiple photographs such as the data from different sensors, times, depths, or viewpoints. Image registration is a very important and popular research field in image processing, not only because the requirement of the medical image processing such as Computational Anatomy(CA) which is applied to study health, disease and epidemiology and uses deformable mappings between images as a basic technique, but also because it is the basis for the application of the meteorological satellite spectral image data[1]. There are various methods for image registration, such as affine, deformable Directly Manipulated Free-Form Deformation(DMFFD), diffeomorphic transformation method with mutual information(MI), mean squared difference(MSQ) or fast cross correlation(CC) as its similarity measure, thin spline method, landmark method and so on. In this paper, an new automatic image registration will be given and discussed. This method has three parts which are edge searching, image registration method by using multivariate spline function and automatic image registration with satellite images. That is, we will introduce a new image registration method by using multivariate spline functions. To facilitate the application with some previous results, we generalized them by using matrix variable. The basic idea of our method is as follows: we had already found the image registration method by using multivariate spline function, now we try to update it as an automatic image registration method, so, firstly, we need find a suitable edge searching method for this method, secondly, we need a method which can make those two parts which are edge searching and image registration method by using multivariate spline function works automatically. The summary of this paper as follows:

Chapter 1 includes the introduction and the preliminaries. The first section includes the background of image registration and the recent progress of Spline Methods. The second section provides definitions and theorems which will be used in this paper.

Chapter 2 describes the boundary selection method for multivariate spline method. There are four sections in this chapter: In section 1, the Pre-boundary Selection Method 1 will be introduce, this method is based on using Otsu's method. In section 2, the Pre-boundary Selection Method 2 will be introduce, this method is based on using sobel operator. In section 3, the Boundary Selection Method will be given by using Pre-boundary Selection

Method 1 and Pre-boundary Selection Method 2. Section 4 introduces Object Recognition Method which is based on the Boundary Selection Method.

Chapter 3 describes the multivariate Spline Method. There are four sections in this chapter: In section 1, the feature point is introduced. In this section, feature points are chosen as the interpolation points for multivariate spline functions. Section 2 is affine transformation. In this section, we will demonstrate the way to calculate the affine transformation by using the feature points. Section 3 introduces the Delaunay triangulation. By using the corresponding interpolation points in output image, we will perform the corresponding Delaunay triangulations in this section. Section 4 shows the application of multivariate spline interpolation. In this section, the barycentric coordinates are used to evaluate the expression of multivariate spline functions. Section 5 describes size determination. In this section, the size of the output image is determined. Section 6 is the valuation scheme. In this section, two valuation schemes can be used for output image.

Chapter 4 describes the Automatic Image Registration by using Multivariate Spline Method. In this chapter, there are 2 sections. Section 1 is Pre-Selection Method for Automatic Image Registration, the basic idea of the Pre-Selection Method is tried to find the first three corresponding feature points automatically, then tried to find more corresponding feature points as much as possible so that we can use the Multivariate Spline Method. Section 2 is Adding point Method for Automatic Image Registration, the basic idea of Adding point Method is that try to match the number of points for the pair of corresponding feature points, we add enough points so that the number of the points is equal to the least common multiple(LCM) of the number of the points in corresponding feature points, then tried to find more corresponding feature points as much as possible so that we can use the Multivariate Spline Method.

Chapter 5 includes results and experimental procedures. We will show some results by using both satellite image and MRI.

Chapter 6 includes the conclusion and future work. Some conclusion and future work will be given in this chapter.

Key words: Automatic Image Registration, Multivariate, Spline, Delaunay Triangulation, Affine Transformation, Edge Searching.

TABLE OF CONTENTS

LIST OF FIGURES OR ILLUSTRATION

Chapter 1: INTRODUCTION AND PRELIMINARIES

In this chapter, the first section includes the background of image registration and the recent progress of spline methods. The second section provides definitions and theorems which will be used in this dissertation.

1.1 Introduction

Image registration is an image processing to transform different sets of data in which one has an overlap with another into one coordinate system. Usually, the data is multiple photographs such as the data from different sensors, times, depths, or viewpoints.

For image processing, there is no general agreement among authors regarding where image processing stops and other related areas. The birth of what we call digital image processing today can be traced back to 1964 when pictures of the moon transmitted by Ranger 7, those pictures were processed by a computer to correct various types of image distortion inherent in the on-board television camera at Jet Propulsion Laboratory(Pasadena, California), see figure 1.1.

Since 1979, The tomograph was invented by Godfrey N. Hounsfield and Allan M. Cormack

Figure 1.1: First Picture of the Moon Transmitted by Ranger 7[3]

and in 1895 X-rays were discovered by Wilhelm Conrad Roentgen[3]. These two inventions still led to some of the most active application areas of image processing today, such as image registration.

In the early days, image registration originated from the optical flow. Nowadays, most researchers treat them as the same problem, since both of them focus on the relationship between the given two images which may have been obtained at different times or different views and so on for the same or different objects. In my opinion, there is still small difference. Optical flow can give us important information about the spatial arrangement of the objects viewed and the rate of change of this arrangement[4]. Discontinuities in the optical flow can help in segmenting images into regions that correspond to different objects[5]. Here are some early papers about how to deal with this kind of problem [6,7,8]. All of the above works dedicate to predict the relationships and future changing of the interested objects, by image registration, based on the existing images obtained from different time, different visual angle and so on. It is very useful for weather forecasting and television signals.

Nowadays, many researchers focus on image registration. For image registration, studying the relationship between the given images is still very important. After we know the relationship, we want to put the given images in one coordinate system or one image(output image). In recent years, there are a host of clinical applications requiring this kind of image processing. For example, one would like to compare two Magnetic Resonance Imaging (MRI) scans of a patient, taken say six months ago and yesterday. Or someone needs to identify differences between the two, e.g., the growth of a tumor during the intervening six months. Someone may want to align Positron Emission Tomography (PET) data to an MR image, so that to help identify the anatomic location of certain mental activation [9]. More over, one may also want to register lung surfaces in chest Computed Tomography (CT) scans for lung cancer screening [10]. While all of these identifications can be done in the radiologists head, the possibility always exists that small, but critical, features could be missed. So the

image registration is required to deal with those kind of problems.

More and more researchers are working on these kinds of problems, and some of them use spline functions as the image registration method. Here are some published papers on this research field: In 1994, Richard Szeliski and James Couglan use spline function to do image registration[11]. In this paper, this algorithm is used to solve the problem including the computation of optic flow (general pixel-based motion), stereo correspondence, structure from motion, feature tracking and the recovery of 3D projective scene geometry. This method is based on solving the image for CCD camera(visible light) so they use the first-degree Bezier curve form as the spline interpolation function. The result looks petty good for CCD camera image like trees, buildings and so on. In 1997, Seungyong Lee, George Wolberg and Sung Yong Shin describe a fast algorithm for scattered data interpolation and approximation. Multilevel B-splines are introduced to compute a C^2 -continuous surface through a set of irregularly spaced points. They got a good result for image registration on the human face(need a non-linear transform). This method we consider the similarity and transformation together, which is much faster. In recent years, as the medical requirement, a lot of people focus on CT image and MRI for human organs, this is essentially different with CCD camera images and much more difficult to do image registration.

In 1999, D. Rueckert,* L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes used the Multilevel B-splines Method for breast MRI image registration, named by Free-Form Deformation(FFD)[12], in which normalized mutual information is used as a voxel-based similarity. In 2006, Nicholas J. Tustison and James C. Gee used the parameterized reconstruction method for data fitting. The parameterized reconstruction method employs n-D C^k B-splines with typically utilized least-squares methodology for data fitting[13]. In 2009, Nicholas J. Tustison, Brian B. Avants, and James C. Gee improved FFD method for image registration, by directly using manipulated free-form deformation without similarity, named DMFFD[14]. In 2013, Feiyu Chen, Peng Zheng, Penglong Xu, Andrew D.

A. Maidment, Fengshan Liu and Xiquan Shi presented non-linear local affine transformation method for image registration[15]. In [15], Sherped interpolation method is used to calculate the affine transformation of each pixel.

The theory of multivariate spline functions has relatively matured. Actually, in 1977, M. J. D. Powell and M. A. Sabin already proved that, under the Powell-Sabin Scheme, piecewise quadratic spline function exist a unique solution[16] for the given function and partial derivative values on the vertices of a triangulation. In 1988, Renhong Wang , Wubao Wang, Shoming Wang and Xiquan Shi found a method to calculate the piecewise quadratic spline function for the Powell-Sabin Scheme on the same assumption[17]. In this paper, we will generalize those results to matrix variable for image registration.

1.2 Preliminaries

In this section, some definitions and theorems are introduced which will be used in this paper.

1.2.1 Transformations

In this section, I will introduce some transformations.

Definition1.1: In Euclidean geometry, a translation is a function that moves every point a constant distance in a specified direction.

Matrix representation:

$$T_v p = \begin{pmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix} = \begin{pmatrix} p_x + v_x \\ p_y + v_y \\ p_z + v_z \\ 1 \end{pmatrix} = p + v$$

.

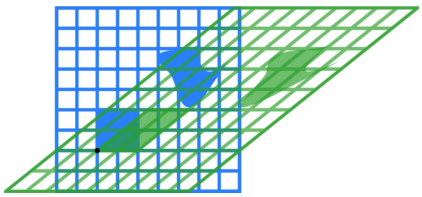

Figure 1.2: Shear Mapping

Definition1.2: In Euclidean geometry, uniform scaling is a linear transformation that enlarges (increases) or shrinks (diminishes) objects by a scale factor that is the same in all directions.

Matrix representation:

$$S_v p = \begin{pmatrix} v_x & 0 & 0 \\ 0 & v_y & 0 \\ 0 & 0 & v_z \end{pmatrix} \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} = \begin{pmatrix} p_x v_x \\ p_y v_y \\ p_z v_z \end{pmatrix}$$

. **Definition1.3**: A rotation is a circular movement of an object around a center (or point) of rotation. A three-dimensional object rotates always around an imaginary line called a rotation axis.

2D Matrix representation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

, Where Θ is the rotation angle. **Definition1.4**: In plane geometry, a shear mapping is a linear map that displaces each point in fixed direction, by an amount proportional to its signed distance from a line that is parallel to that direction, see Figure 1.2. Matrix

representation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x + my \\ y \end{pmatrix} = \begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

,

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ mx + y \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ m & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

. **Definition1.5**: In geometry, an affine transformation is a function between affine spaces which preserves points, straight lines and planes include shear, translation, scaling and rotation. Also, sets of parallel lines remain parallel after an affine transformation.

Augmented matrix:

$$\begin{pmatrix} \overrightarrow{y} \\ 1 \end{pmatrix} = \begin{pmatrix} A & \overrightarrow{b} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \overrightarrow{x} \\ 1 \end{pmatrix}$$

. This is equivalent to $\overrightarrow{y} = A\overrightarrow{x} + \overrightarrow{b}$. The above mentioned augmented matrix is called affine transformation matrix, or projective transformation matrix.

1.2.2 Spline Function

In this section, spline function and some theorem are introduced that will used in this work.

Definition1.6[2]: Multivariate spline functions is defined by

$$S_k^{\mu}(\triangle) \ = \{s \in C^{\mu}(D) | s|_{D_i} \in P_k, i = 1, 2, ..., N\}.$$

Where $\triangle$ is the partition of domain D, denote by D_1, D_2, ..., D_N, and

$$P_k \ := \{p = \sum_{i=0}^{k} \sum_{j=0}^{k-i} c_{ij} x^i y^j \mid c_{ij} \text{ is a real value}\}.$$

. **Definition1.7**[18]: A d-degree Bezier hyper surface is given by

$$C(u_1, u_2, ..., u_n) = \sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} ... \sum_{i_n=0}^{N_n} P_{i_1,i_2,...,i_n} \prod_{j=1}^{n} B_{i_j,d}(u_j) \tag{1.1}$$

where the basis for the Bezier hyper surface is given by

$$B_{i,d}(u) = \binom{d}{i} (1-u)^{d-i} u^i, \qquad 0 \ \leq \ u \ \leq \ 1. \tag{1.2}$$

Definition1.8[18]: Let U = $\{u_0, u_1, ..., u_m\}$ be a nondecreasing sequence of real numbers, i.e., $u_i \leq u_{i+1}$, i = 0, ..., m-1. The u_i are called knots, and U is the knot vector. The ith B-spline basis function of p-degree(order $p+1$), denoted by $N_{i,p}$(u), is defined as

$$N_{i,0}(u) = \begin{cases} 1, & u_i \leq u < u_{i+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{1.3}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \tag{1.4}$$

Definition1.9[18]: A p-degree n-D B-spline hyper surface is defined as

$$S(u_1, u_2, ..., u_n) = \sum_{i_1=0}^{N_1} \sum_{i_2=0}^{N_2} ... \sum_{i_n=0}^{N_n} P_{i_1,i_2,...,i_n} \prod_{j=1}^{n} N_{i_j,d}(u_j) \tag{1.5}$$

Lemma1.1[2]: Let $p(x, y) \in P_k$. If certain n zeros (x_i, y_i)(i=1, ..., n), $n \geq$ k+1 of a linear polynomial

$$\ell(x, y) = ax + by + c, qquada^2 + b^2 \neq 0$$

are also the zeros of $p(x, y)$. Then $p(x, y)$ is exactly dividable by $\ell(x, y)$, that is, there is a polynomial $q(x, y) \in P_{k-1}$ such that

$$p(x, y) = \ell(x, y)q(x, y) \tag{1.6}$$

. **Lemma1.2**[2]: Let $p(x, y) \in P_k$, and $q(x, y)$ be an irreducible algebraic polynomial. If $p(x, y)$ and $q(x, y)$ have more than km common zeros, then $p(x, y)$ is divisible by $q(x, y)$. Namely, there is a $r(x, y) \in P_{k-m}$ such that $p(x, y) = q(x, y)r(x, y)$.

Theorem1.3[2]: Let the representation of $z = s(x, y)$ on the two arbitrary adjacent cells D_i and D_j be

$$z = p_i(x, y) \quad and \quad z = p_j(x, y)$$

where $p_i(x, y)$ and $p_j(x, y) \in P_k$. In order to let $s(x, y) \in C^{\mu}(\overline{D_i \bigcup D_j})$, if and only if there is a polynomial q_{ij}(x,y)$\in P_{k-(\mu+1)d}$ such that

$$p_i(x, y) - p_j(x, y) = [\ell_{ij}(x, y)]^{\mu+1} q_{ij}(x, y), \tag{1.7}$$

where $\overline{D_i}$ and $\overline{D_j}$ have common interior edge

$$\Gamma_{ij} : \ell_{ij}(x, y) = 0,$$

and the irreducible algebraic polynomial $\ell_{ij} \in P_d$.

Theorem1.4[2]: Let $\triangle$ be a straight line partition of D. The multivariate spline functions $s(x, y) \in S_k^{\mu}(\triangle)$ exists, if and only if for every interior mesh segment, there exists a smooth cofactor of the s(x,y), and satisfies the global conformality condition

$$\sum_{A_v} [\ell_{ij}(x, y)]^{\mu+1} q_{ij}(x, y) \equiv 0, \tag{1.8}$$

where A_v goes through all the interior mesh points, $\ell_{ij} \equiv a_i x + b_i y + c_i = 0$ is an interior mesh segment around A_v, and $q_i(x, y) \in P_{k-\mu-1}$ is the smooth cofactor on $\ell_i(x, y)$.

1.3 Main idea of this paper

In this section, I will show you the main idea of this paper by a picture. In this paper, we build 2 different boundary selection methods, by combining those two methods, we find the boundary image. We build a new simple method to extract the closed boundary from the boundary image. Then we automatically matching the corresponding closed boundaries between two given image, produce feature points and affine transformation for the corresponding feature points. By using the feature points, we construct the triangulation. As finding the mapping is the key for image registration. We obtain the mapping by using the Multivariate Spline Functions which is affine transformation valued, not a traditional Spline Function. This Multivariate Spline function is the mapping which defined on this triangulation. We given two method for constructing this mapping. One is called Pre-Selection Automatic Image Registration by using Multivariate Spline Method, The other is called Adding point Automatic Image Registration by using Multivariate Spline function Method. By the way, we did a lot of experiments for satellite images, at present, the automatic image registration method given in this paper works well for all current experiments, see chapter 5 for more details.

For the detail of each part of this automatic image registration method will be shown in chapter 2, chapter 3 and chapter 4.

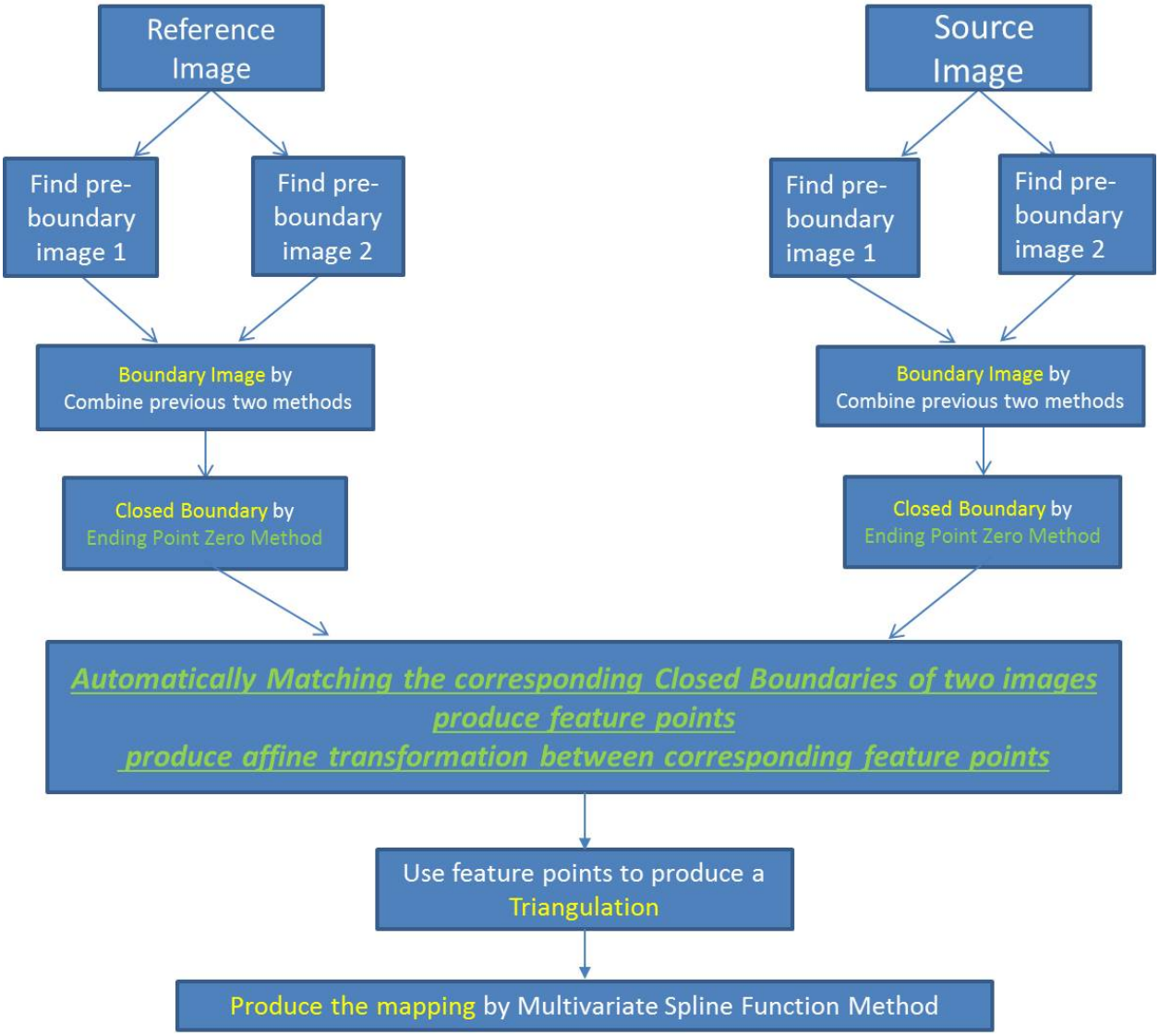

Figure 1.3: main idea of this paper

Chapter 2: AN EDGE OF OBJECT RECOGNITION METHOD FOR MULTIVARIATE SPLINE METHOD

In this chapter, a boundary selecting method will be given for satellite image. In order to satisfy the multivariate spline Method, our boundary selection method is used for finding the object in the image which the objects' boundary has a closed curve. We will talk about our boundary selecting method step by step in this chapter, further more, this method will works for finding the closed curve. This Method include 2 boundary selection method. We combine these two boundary selection method, so that the new boundary selection will works well for the Multivariate Spline Method.

2.1 Pre-boundary Selection Method 1

In this section, we will introduce the first boundary selection method which we called pre-boundary selection method 1. There are two steps for this method. Each step will be given in the following sections.

2.1.1 Otsu's method

As we know, most of satellite images are RGB images. There already have many methods to convert RGB image to gray image, such as using the three values of each pixel in RGB image to convert it into one value. So we don't talk about it in our paper.

Firstly, in order to find the boundary, we will try to convert gray image to binary image. To the best of my knowledge, the Otsu's Method is one of the best simple and feasible method to find the threshold for a given gray image. The other method is fisher linear discriminant.

Here is the Otsu's method[**20**]:

1.Searching for the threshold that minimizes the intra-class variance (the variance within the class), defined as a weighted sum of variances of the two classes ω_0 and ω_1:

$$\sigma_\omega^2 = \omega_0(t)\sigma_0^2 + \omega_1(t)\sigma_1^2 \tag{2.1}$$

where $\omega_0(t) = \sum_{i=1}^{t} p(i)$ and $\omega_0(t) = \sum_{i=t+1}^{n} p(i)$, t is the threshold, $p = n(i)/N$ is the probability.

2.Otsu shows that minimizing the intra-class variance is the same as maximizing inter-class variance:

$$\sigma_b^2(t) = \omega^2 - \sigma_\omega^2 = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2 \tag{2.2}$$

which is expressed in terms of class probabilities ω and class means μ.

where $\mu_0(t) = \sum_{i=0}^{t-1} \frac{ip(i)}{\omega_0}$, $\mu_1(t) = \sum_{i=t}^{L} \frac{ip(i)}{\omega_1}$,$\mu_T = \sum_{i=0}^{L} ip(i)$.

As inter-class variance is first order, and intra-class variance is second order, so using the maximizing inter-class variance.

Note: intra-class variance: $\sigma_\omega^2 = \omega_0\sigma_0^2 + \omega_1\sigma_1^2$, inter-class variance: $\sigma_b^2 = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2$

code:

$function[threshold_otsu] = ThresholdingOtsu(Image)$

$nbins = 256;$

$counts = imhist(Image, nbins);$

$p = counts/sum(counts);$

$for t = 1 : nbins$

$q_L = sum(p(1 : t));$

$q_H = sum(p(t + 1 : end));$

$miu_L = sum(p(1:t).*(1:t)')/q_L;$

$miu_H = sum(p(t+1:end).*(t+1:nbins)')/q_H;$

$sigma_b(t) = q_L * q_H * (miu_L - miu_H)^2;$

end

As Otsu's method is roughly a one-dimensional, discrete analog of Fisher's Discriminant Analysis, we can use Fisher's Discriminant Analysis as well.

Here is the Finisher's Discriminant Analysis[**18**]:

Let

$$J(\omega) = \frac{\omega^T * S_B * \omega}{\omega^T * S_w * \omega} \tag{2.3}$$

where $S_B = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$ is called Between-class scatter, $S_w = S_1 + S_2$ is called Between-class scatter, $S_i = (x - \mu_i)(x - \mu_i)^T$, $\mu_i = \frac{1}{N}\sum_{x \epsilon \omega_i} x$

Finding the ω which can maximin the $J(\omega)$ is the key. By solving the following equation:

$$S_\omega^{-1} S_B \omega = \lambda\omega \tag{2.4}$$

we can find this ω. Actually, without solving the equation,

$$\omega = S_\omega^{-1} * (\mu_1 - \mu_2) \tag{2.5}$$

In the picture, the first picture show a direction of line which works bad for separating the

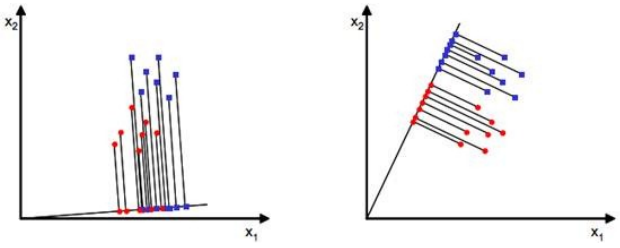

Figure 2.1: Finisher's Discriminant

given points, the second picture show the right direction ω which separate the given points well. We can use this method to convert RGB image to gray image too.

In this paper, we will use first method as example, which is called Otsu's method, as the first step of finding the boundary.

Here is the input images(we take 2 satellite images as input):

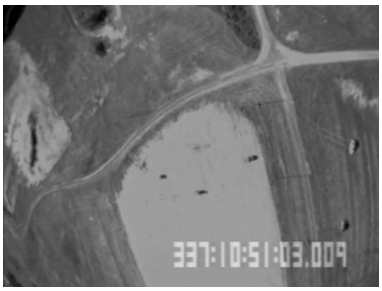

Figure 2.2: Satellite017

Figure 2.3: Satellite018

Here is output images by using the Otsu's method:

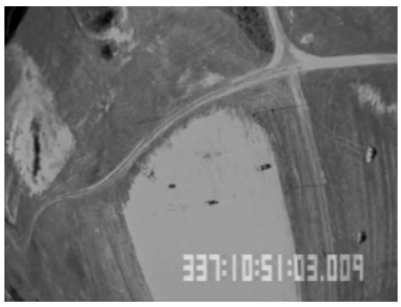

Figure 2.4: Satellite0171

Figure 2.5: Satellite0181

2.1.2 Pre-boundary Determining

In this section, by using the result of gray images in the last section, we will find the boundary. we call this method as pre-boundary determining Method 1.

As the image is gray image (i.e. the value of each pixel is 0 or 1), it will easy for us to find the boundary, here is the method: First, in order to make the zero and non-zero points

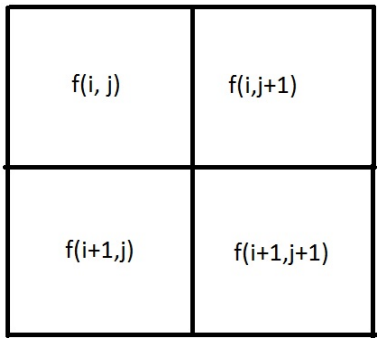

Figure 2.6: 4 pixels

separately, we multiply each pixel by n, such as $n = 2$:

$$f(i,j) = f(i,j) * n \tag{2.6}$$

Second, for each pixel $f(i,j)$, find the threshold R by multiply a scale on the different between maximum and minimun of its $n^2 - 1$ neighbourhoods (take $n = 2$ as example see Figure 2.6). we can use this scale to control the boundary even if the given image is not a 0 or 1 image. If the different between $f(i,j)$ and its three neighbourhoods is greater than the threshold R, we will treat this point to be a boundary point. For $\forall i, j \in Z^+$

$$R(i,j) = \alpha(max_{i,j=1,2}(f(i,j)) - min_{k=0,1}(f(i+k, j+k))) \tag{2.7}$$

where α is the scale, $\alpha \in (0,1)$.

Note: The scale $\alpha \in (0, 1)$ is using for control the number of the boundary points. For any input images, a bigger scale will give us less boundary points.

Here is the code:

$function image1 = edge3(image1, d)$

$[m, n] = size(image1);$

$image1(m, :) = 0;$

$image1(:, n) = 0;$

$for i = 1 : (m - 1)$

$for j = 1 : (n - 1)$

$R = (max(max(image1(i : i + 1, j : j + 1))) - min(min(image1(i : i + 1, j : j + 1))))/d;$

$if abs(image1(i, j) - image1(i + 1, j)) > R || abs(image1(i, j) - image1(i + 1, j + 1)) > R || abs(image1(i, j) - image1(i, j + 1)) > R$

$image1(i, j) = 100;$

$else$

$image1(i, j) = 0;$

end

end

end

Here are the output images(see Figure 2.7 and Figure 2.8):

Figure 2.7: Satellite0172

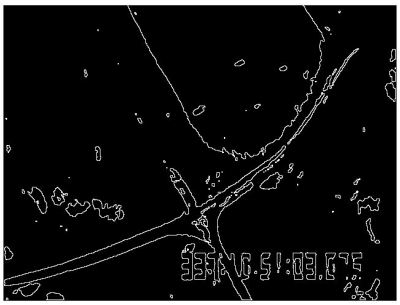

Figure 2.8: Satellite0182

Form the out put image, some of close curves are show clearly. But there are a lot of non-useful close curves too, we called it noisy curves. We want to select the exact object's boundary as a close curve for us to use in the further, so we need try to remove those noisy curves.

2.2 Pre-boundary Selection Method 2

In this section, we will talk about the second method for finding image's boundary, called pre-boundary selection method 2. The following subsection will show the method step by step.

2.2.1 Find image gradient by using sobel operator

We are trying to use the derivative for boundary selection. As this is one of the most general mathematic idea for finding the boundary of image. After the researching, we got the second method in hand.

As the 2-D image has 2 dimensions, in order to have both 2 dimensions direction derivative include in our method for boundary selection, of cause, we will try to use the image gradient, such as gradient's magnitude and gradient's direction. The image gradient is a directional change in the intensity or color in an image, its may be used to extract information from images. To find the image gradient, there are some operators for us to use. For example, Sobel, Prewitt, Central Difference and so on.

In this paper, we used sobel operator to find the image gradient. Here is the Multidimensional discrete convolution formula:

$$f(n_1, n_2, ..., n_k) = \sum_{i_1=-m_1}^{m_1} \sum_{i_2=-m_2}^{m_2} ... \sum_{i_k=-m_k}^{m_k} h(i_1, i_2, ..., i_k) x(n_1 - i_1, n_2 - i_2, ..., n_k - i_k) \quad (2.8)$$

First, we need calculate the derivative for x-axis and y-axis.

For generally 2-D case,

$$G_x(i, j) = f(i + 1, j) - f(i, j) \quad (2.9)$$

$$G_y(i, j) = f(i, j + 1) - f(i, j) \quad (2.10)$$

where $f(i,j)$ is the value of (i,j)'s pixel in image matrix A.

For 2-D sobel operator,

$$G_x(i,j) = \alpha * \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} * A \tag{2.11}$$

$$G_y(i,j) = \alpha * \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} * A \tag{2.12}$$

where $*$ is discrete convolution, A is a given image matrix, α is a parameter constant, usually $\alpha = 1$.

The gradient magnitude:

$$G(i,j) = \sqrt{G_x(i,j)^2 + G_y(i,j)^2} \tag{2.13}$$

The gradient direction:

$$\Theta(i,j) = \arctan(\frac{G_y(i,j)}{G_x(i,j)}) \tag{2.14}$$

Here are the output images(see Figure 2.9 and Figure 2.10):

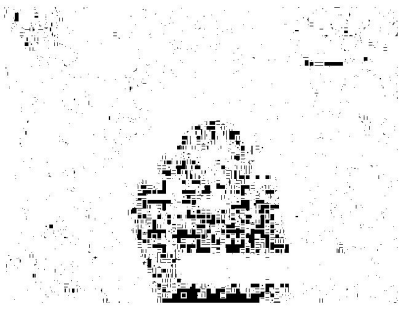

Figure 2.9: 017 magnitude

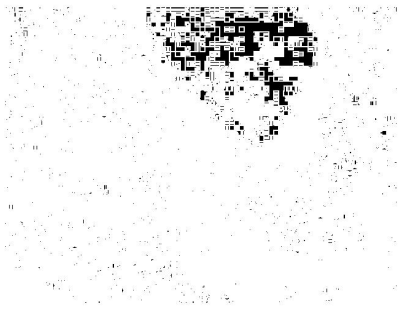

Figure 2.10: 018 magnitude

According to the Figure 2.9 and 2.10, the results are not good enough for us to use it as the

boundary of the given image, of cause, we can't find any object on the out put images ether. So we need further steps to make it works.

Note: The gradient magnitude image can give us some information about the value of the directional derivative, so we should us it for boundary searching. But the gradient direction can't, it show some information about the direction, we didn't use it for boundary selection.

2.2.2 Find boundary by using Image Gradient

In last section, we obtain the image gradient. In this section, we will try to use the gradient's magnitude to find the boundary of the given image. The basic idea is similar as the Pre-boundary selection method 1, such as find the boundary point, set boundary point value nonzero and other points value zero. So our boundary image should be a binary image too. But the fuzzy mathematics will be used in this method, the reason is that we can't determine which pixel is the boundary or not(see Figure 2.9 and 2.10), even using human eye directing, it's still difficult to find the boundary points directly. As we try to use the fuzzy to do the boundary selection, how to find the degree of membership for each pixels in the given gradient's magnitude image is the key for Pre-boundary selection method 2. The range of degree of membership is $[0, 1]$.

By the way, because those two method - Pre-boundary selection method 1 and Pre-boundary selection method 2, are the advance preparation for the boundary selection, may not be the exact boundary, we can use fuzzy mathematical method to help us for Pre-boundary selection, even the boundary image may be include some of non-boundary pixels. But disadvantage and advantage always come together. Because of this, in this method, we can also use the degree of membership to control the number of boundary points, so that we can

control the sharpness level of the boundary image. This is really helpful for us to find the image's finial edge.

The simplest method for calculate the degree of membership for each pixel is as follow: In order to separate the value of each pixel well, firstly we square each pixels. Secondly, in order to evaluate the degree of membership, as you know, the range of the degree of membership is between 0 and 1, first step we find the maximum value and minimum value for the given image matrix A_{m*n}.

$$M_f = max_{(i,j)}(f(i,j)), \qquad m_f = min_{(i,j)}(f(i,j)). \tag{2.15}$$

By using the equation 2.16, we can calculate the degree of membership for each pixel by expression 2.16:

$$f(i,j) = (f(i,j) - m_f)/M_f \qquad \forall i,j \in Z, \quad 0 \le i \le m, \quad 0 \le j \le n. \tag{2.16}$$

The results are as follow(see Figure 2.11 and 2.12): When we obtain the degree of membership

Figure 2.11: 017 degree of membership

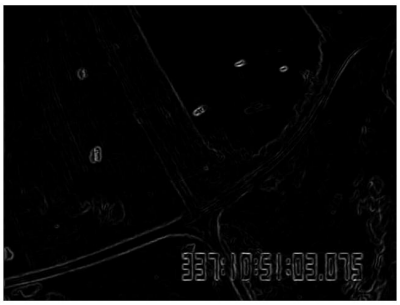

Figure 2.12: 018 degree of membership

for each pixel, we can choose a threshold ε(see equation 2.17) for Figure 2.11 and 2.12 so

that I can reduce the non-boundary pixels.

$$\varepsilon = 0.1^n \qquad n > 0 \tag{2.17}$$

So, we can use n to control the threshold ε. The boundary points will be all pixels which the degree of membership is greater than ε, the Figure 2.13 and 2.14 are the pro-boundary images. From the output image Figure 2.13 and 2.14, it is clearly that the boundary of the

Figure 2.13: 017 n=1.25

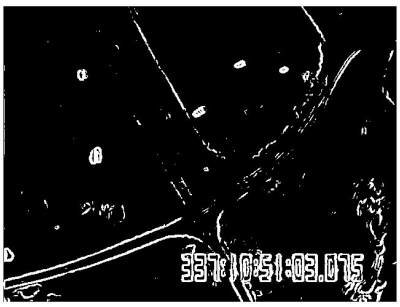

Figure 2.14: 018 n=1.25

objects such as cars in the image may have some breakpoints, and this is means that there are some boundary points missing in this pro-boundary images. So, Dilation ([**3**] 9.1) for this pro-boundary image A is need to make sure that we already include the boundary points as many as we can. There are some other methods such as opening or closing Digital Image Processing 9.3.

Dilation for Matrix:

Let A and B are matrix. The dilation of A and B, denoted as $A \bigoplus B$, is defined as

$$A \bigoplus B = \{z \mid (\hat{B})_z \bigcap A \neq \phi\} \tag{2.18}$$

Formula 2.18 is equivalent to Formula 2.19.

$$A \bigoplus B = \{z \mid (\hat{B})_z \bigcap A \subset A\} \tag{2.19}$$

where $\hat{B} = \{z \mid z = -x, \forall x \in B\}$ is called the reflection of set B, $B_z = \{x \mid x = a - z, \forall a \in B\}$ is called the translation of set B by point $z = (z_1, z_2)$.

Take

$$B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{2.20}$$

This matrix B will be simplest matrix for us to do the dilation with the pro-boundary image A. we pick up four-neighborhoods of each boundary points as the boundary points, of cause, if this point is not a boundary point. We should mention that $A \bigoplus B$ is a set including all non-zero points when we do the discrete convolution of matrix A and B. Such as expression 2.21 and 2.22, expression 2.22 may not be zero if $A(1,1) \neq 0$, under this hypothesis, the point (0,1) will be a new boundary point, as image matrix should count begin with (1,1) and (0,1) can't be a point in the pro-boundary image A.

$$f(0,0) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} B(i,j) * A(0-i, 0-j) = B(-1.-1) * A(1,1) = 0 * A(1,1) = 0. \tag{2.21}$$

$$f(0,1) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} B(i,j) * A(0-i, 1-j) = B(-1.0) * A(1,1) = A(1,1) \tag{2.22}$$

Note: Because the convolution formula require matrix B from a negative number to a positive number, the size of matrix B should be odd $*$ odd, and 3 is the minimum odd number to be the size of a matrix. This is the Pre-boundary selection method 2. From output image (the figure 2.15 and figure 2.16), we find that the special objects in the original image(see figure

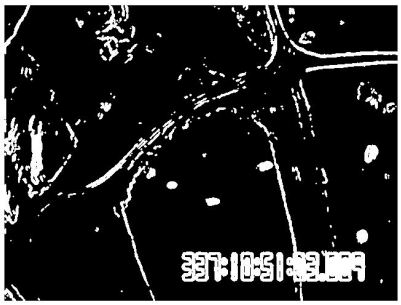

Figure 2.15: 017 delation

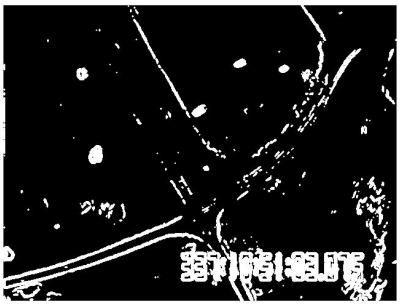

Figure 2.16: 018 delation

2.2 and 2.3) already include in the new pro-boundary image, and most of the noise edge in the figure 2.7 and 2.8 are vanished. This is really good for us to find the boundary of the given image.

2.3 Boundary Selection Method by using Method 1 and Method 2

The last step of our Boundary Selection Method for Multivariate Spline Method is find the intersection part of the output images from Pre-boundary selection method 1 and Pre-boundary selection method 2. Let Image Matrix A and B be the output image matrix form Pre-boundary selection method 1 and Pre-boundary selection method 2 separately. As addition and subtraction matrix is faster than For-loop in MATLAB, we find the intersection by this way:

1. Let $C = B - A$.

2. We set the value of pixels which's value is negative equals to 0 in image matrix C.

3. The boundary image matrix will equal to $B - C$. So, the boundary points which only belongs to Method 1 will vanished.

So, the figure 2.17 and 2.18 are the boundary image by using our Boundary Selection Method

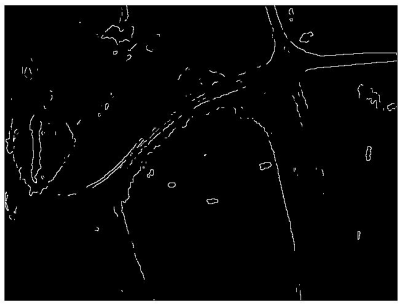

Figure 2.17: 017 boundary image

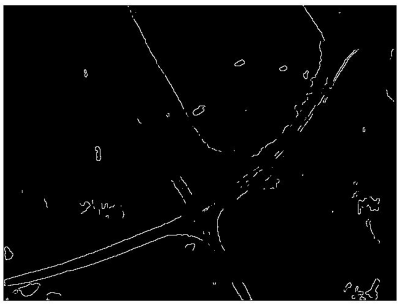

Figure 2.18: 018 boundary image

for Multivariate Spline Method.

2.4 Edge of Object Recognition Method by using Boundary Selection Method

In this section, we will talk about an Edge of Object Recognition Method by using the Boundary Selection Method which we have already talked before, we called this Object Recognition Method as **Ending Point Zero Method**.

As we have already obtained the boundary images(Figure 2.17 and Figure 2.18) for the input images(Figure 2.2 and Figure 2.3). We will use it as example to show the Edge of Object Recognition Method. As the edge of objects should be a closed circle, our goal is to find all the closed circle in the boundary image.

First Step: The **Ending Point Zero Method** is inspected the boundary image column by column to find the nonzero pixels. Take a 480 $\times$ 640 image as example, this is a 480 rows and 640 columns' image, the order of the inspection is form [1,1] to [1,480] then move to next column form [2,1] to [2,480] and so on, until we find an nonzero pixel in boundary image.

Second Step: The **Ending Point Zero Method** need a searching direction so that we can find the next point after we obtain an nonzero pixel. There are 4 directions such as up-right, up-left, down-right, and down-left. Each Direction has a direction matrix. The size of the

direction matrix is changeably. Usually, bigger size of direction matrix give us more object in the result image. Each pixel of a direction matrix should be ordered. The order is flexible depend on what we needed. In this paper, we use 3 by 3 matrix as our direction matrix, look at Table 2.1 and 2.2 below, '1, 2, 3, 4, 5, 6, 7, 8' represent the order of each pixel which is the picking up priority for each pixel in the direction matrix.

3	4	1
5	2	6
origin pixel	7	8

(a) up-right diection

1	4	3
6	2	5
8	7	origin pixel

(b) up-left diection

Table 2.1: up direction

origin pixel	7	8
5	2	6
3	4	1

(a) down-right diection

8	7	origin pixel
6	2	5
1	4	3

(b) down-left diection

Table 2.2: down direction

If every pixel is zero, we will change the direction to next direction. As we have 4 directions the order of the direction is up-right, down-right, then down-left and up-left, and then up-right. So that we can find a close curve from the boundary image. The direction of next pixel will be set as the current direction. Note:

- The direction of initial pixel is up-left.
- If the direction didn't change, the direction of next pixel will be set as the current pixel's direction, otherwise, the direction of next pixel will be set as the current direction.

Third Step: In **Ending Point Zero Method**,

- case 1 is that the direction is changed 4 times, we still didn't find the non-zero pixel, we will set the current pixel as zero, then move to step one.

- Case 2 is the current pixel is the initial pixel or belong to a common 3 by 3 matrix. We will take those pixel as a edge of an object and set them as zero, then move to step one to find new close curves.

The Ending Point Zero Method result is as follow, see Figure 2.19 and 2.20, the blue curve show all the pixel we pick up in the boundary image by using Ending Point Zero Method which is the edge of object we find in the given image.

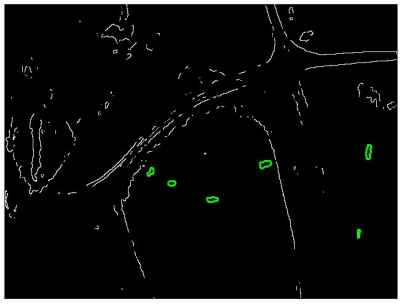

Figure 2.19: 017 edge of objects

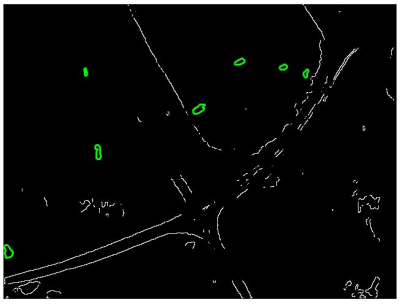

Figure 2.20: 018 edge of objects

Chapter 3: THE DESCRIPTION OF MULTIVARIATE SPLINE METHOD

3.1 Introduction

In this chapter, multivariate spline method for image registration is introduced step by step. This method is based on the two dimensional spline function. As we know, a lot of researchers working on the image registration by using spline and many papers have been published, most methods using the multilevel B-splines method for image registration, our method is based on the multivariate spline functions for image registration, whereas our method is more reasonable, flexible and easy to control. To our knowledge, multivariate spline function is the first on the application of the actual meaning of multi-dimensional spline functions for image registration. Take "Directly Manipulated Free-Form Deformation image registration"[14] for example, which is a paper published by Nicholas J. Tustison in 2009 on IEEE Transactions on Image Processing. The main idea of [14] is using B-spline functions to approximate the demand function. Actually, Bezier curve, a special case of the B-spline curve, is used to approximate the demand function.

As we know, the two dimensional Bezier curve function combines two one-dimensional Bezier curve bases directly. It is easy for us to get the general form, and it is easy for us to do the calculation, since the bases are as follows:

$$B_{i,d}(u) = \binom{d}{i} (1-u)^{d-i} u^i \qquad \text{for } 0 \le u \le 1 \quad and \quad 0 \le i \le d$$

What we need to do is to use the given conditions to calculate the control point or weight for each Bezier curve base. The success or failure of the affair is all due to the general form, because the domain of the 1-D Bezier curve bases is $0 \le u \le 1$, and two-dimensional Bezier curve function combined two of one dimensional Bezier curve bases directly, then

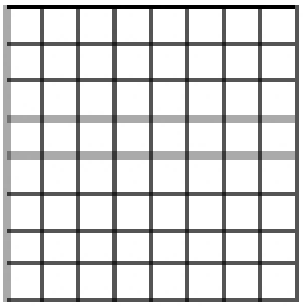

Figure 3.1: Control Lattice for B-spline Function

the control lattice should be composed by unit squares, and overlaid on the domain of the interpolation area where all of the control points are nodes of control lattice(like Figure 3.1). In this way, the relationship between the interpolation points and the partition of the domain is independent. This problem is fatal, since they multiply the two one-dimension Bezier bases directly to get the two-dimension interpolation function. Even though they have extended to three dimensions or n dimensions, this problem is still fatal. Even if we use the general B-spline bases, we still have this kind of problem. We can use any other shape of polygon as the domain for any given piecewise polynomial, except the square or rectangle. Clearly, this is because this kind of B-spline interpolation is not a real two-dimensional spline interpolation, but instead a one-dimensional spline interpolation. we will use the multivariate spline functions which are the real 2-D spline function fir the core technology, and I will give the general form of the two-dimensional spline function and use it to deal with 2D image registration. In order to use this core technology, we find a good algorithm to make it become true. In this chapter, I will introduce it step by step in the following section, and use the breast MRI as an example to explain our method, since breast MRI registration is a very hot research topic in recent years.

3.2 Feature Points

In this section, we will introduce the definition of feature points where finding the corresponding feature points is first step of our method.

Definition3.1: Feature points is a set of two groups of corresponding points which are chosen from the multiple images, say $\{X_1, X_2, ..., X_n\}$ belong to one image and $\{Y_1, Y_2, ..., Y_n\}$ belong to other image, where

$$X_i \to Y_i, \qquad \text{for i=1,2,...,n.}$$

Each X_i and Y_i is a group of the points which are chosen from each image. Here are the corresponding rules for the feature points of X_i and Y_i.

(1) X_i and Y_i belongs to the similar part of each image for i=1,2,...,n.

(2) The sharpness of X_i and Y_i looks like the letter 'X' or 'Y', simultaneously, for i=1,2,...,n.

(3) Choose the intersection point as the first point of X_i and Y_i, and choose the corresponding point around the intersection point on the corresponding side as other points of the X_i and Y_i for i=1,2,...,n.

In this way, we get the feature points X_i and Y_i. Take figures 3.2-3 as an example:

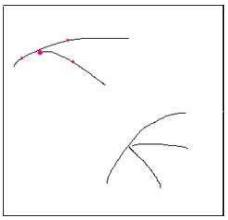

Figure 3.2: Feature Point Example X

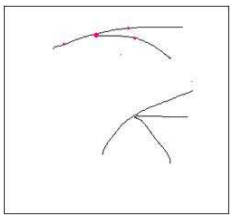

Figure 3.3: Feature Point Example Y

Clearly, the red point in the Figure 3.1 my feature point, say X_1, and the corresponding red point in Figure 3.2 is the corresponding feature point Y_1. Here, we called intersection the

corresponding point as the center point of the feature point or interpolation point.

Are the feature points useful? The answer is affirmative. In the real world, there is a lot of intersecting curves such as the crossroads, vertices of a building and so on. For the image registration of kind of images are easy, and our method works very well on them. These types of images are not our concern. The point is the CT and MR images. Those kind of images are much more complex. Could we find the feature points on these kinds of images? The answer is yes. Here is an example:

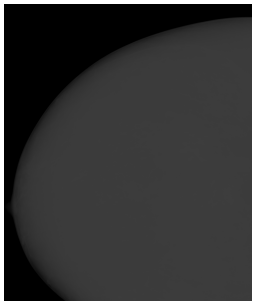

Figure 3.4: Input Example I

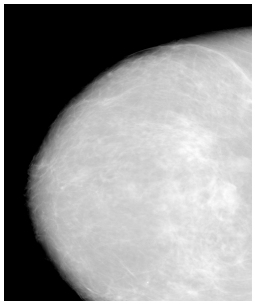

Figure 3.5: Input Example I'

These two images(Figure 3.4 and Figure 3.5) are breast MRIs. Figure 3.4 is the original breast MRI and difficult for us to find the feature points. We can use software ImageJ to do the brightness and contrast(B&R) with the original breast MRI to get Figure 3.5. Clearly, we can find the as many intersecting curves as we want. That is because we make the image clearer by using software ImageJ. As we know, there are connective tissues and ducts in the breast.

Take the Figure 3.4 and Figure 3.6 as an example, where those two images are the MRI for the same person which is sampled at different times, different angles and different compressions.

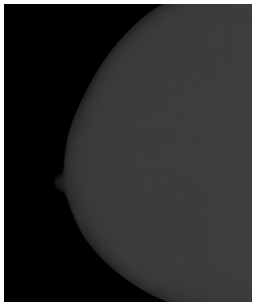

Figure 3.6: Input Example II

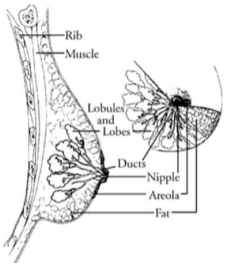

Figure 3.7: Breast Structure

Here are the feature points:

YSETS1=[137.75 138.88 138.5 139.25 139.25; 13.17 14.3 12.42 15.05 11.67];

XSETS1=[119.68 120.81 120.44 121.57 121.19; 49.3 50.06 48.55 50.81 48.55];

YSETS2=[185.45 185.08 184.23 188.75 188.56 189.78; 18.16 18.63 17.41 19.38 20.23 21.08];

XSETS2=[166.07 165.04 164.38 169.55 168.89 169.74; 57.49 57.87 56.64 59.37 59.56 60.12];

YSETS3=[110.93 109.24 106.98 109.52 107.83 112.63 114.6; 21.73 21.45 20.89 24.84 25.97 21.45 20.04];

XSETS3=[91.17 89.48 86.09 90.04 86.66 94 96.25; 57.02 56.45 56.45 58.99 60.69 55.61 54.76];

YSETS4=[202.2 202.11 201.45 202.67 201.45; 31.1 30.77 31.24 31.61 29.73];

XSETS4=[180.94 180.75 180.28 181.41 180.09; 70 69.72 70.38 70.76 68.5];

YSETS5=[202.2 202.58 202.86 202.01 201.82; 103.69 103.97 104.35 104.06 104.53];

XSETS5=[178.49 178.77 179.05 178.21 178.02; 139.25 139.54 139.91 139.63 140.01];

YSETS6=[209.35 208.51 211.33 208.03 205.59 210.76; 127.21 125.61 122.04 129.47 131.73 124.01];

XSETS6=[189.4 188.28 191.38 187.81 184.89 190.72; 160.61 159.3 156.94 163.34 165.51 157.7];

YSETS7=[75.84 75.55 75.27 76.5 76.5 74.61; 91.93 90.99 92.49 91.55 92.77 93.06];

XSETS7=[48.55 47.99 47.61 48.74 49.12 46.47; 135.77 135.02 136.53 135.58 137.09 137];

We will use YSET(i)(: , 1) as interpolation points in following sections, where i = 1, 2, 3, 4, 5, 6, 7. For example, YSET(1)(: , 1) = [137.75; 13.17].

3.3 Best Affine Transformation

In this section, we will introduce Best Affine Transformations which include translation, scaling, similarity transformation, reflection, rotation, shear mapping, and compositions of them in any combination and sequence.

3.3.1 Evaluating Best Affine Transformation By Using Least Squares Algorithm

In the last section, we introduce the feature point. We assume each pair of corresponding feature points are transformed by an affine transformation. It is doable, since the feature point occupies a small local area in each breast MRI, and it can be considered as an affine transformation if the area is small enough, where the transformation in this area is independent with respect to the whole transformation function, although the whole transformation is a non-rigid transformation. According to the data structure of the digital image, the smallest subarea of digital images is pixel. That is, we have to obtain all the affine transformations pixelwisely. Firstly, we find the affine transformations for each interpolation points.

Now, we focus on how to calculate the affine transformation under the feature points X_i

and Y_i. In the last section, we mentioned $X_i = \{X_{i1}, X_{i2}, ..., X_{in}\}$ and $Y_i = \{Y_{i1}, Y_{i2}, ..., Y_{in}\}$ where X_{ij} and Y_{ij} are the corresponding points that belong to the ith feature point. We need find a matrix A and b, such that $AX_{ij} + b = Y_{ij}$ satisfies all j=1,2 ... n, where $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ and $b = \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix}$.
Usually it is impossible to find the A and b to fit with all $AX_{ij} + b = Y_{ij}$ for the ith feature point respectively, since there is at least four corresponding points in each feature points. We can use the Least Squares algorithm(2.3) to find a suitable match.

$$F = \frac{1}{2} \sum_{j} t_j |AX_{ij} + b - Y_{ij}|^2 \tag{3.1}$$

In Equation 3.1, t_j represents the weight of the corresponding points X_{ij} and Y_{ij}.
We will consider the more general case, that is the number of points in each feature point are not equal to each other. We also calculate the Least Squares Algorithm in a more general case.

3.3.2 N-D M-points Least-Squares Algorithm

In last section, Least-Squares Algorithm(2.3) is required for finding the affine transformations for all of the feature points. As you know, the variable of the Least-Squares Algorithm(3.1) is a matrix, and we will do the calculation for the general form of the Least-Squares Algorithm(2.4) which has m corresponding pairs of points and the vector space is n-dimensional with m $\geq$ n+1.

$$F = \frac{1}{2} \sum_{i=0}^{m} t_i |AX_i + b - Y_i|^2 \tag{3.2}$$

In Equation 2.2 $A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$ and $b = \begin{pmatrix} b_{11} \\ b_{21} \\ \vdots \\ b_{n1} \end{pmatrix}$, t_i is the weight of each corresponding points X_i and Y_i,

Now, we need to find an A and b to get the $min_{A,b}$ { F }, and, as we know, the partial derivative should be zero at the extreme point. In this way, we can obtain (3.3)

$$\begin{cases} \frac{\partial F}{\partial a_{pq}} = 0, & 1 \le p, q < n; \quad (1) \\ \frac{\partial F}{\partial b_p} = 0, & 1 \le p < n. \quad (2) \end{cases} \tag{3.3}$$

Assume $F_i = \frac{1}{2} \langle AX_i + b + Y_i, AX_i + b + Y_i \rangle$ for i=1,2,...,m. Where F=$\sum_{i=0}^{m} F_i$.

In this way, by using equation (3.3) (1), we have that:

$$\frac{\partial F_i}{\partial a_{pq}} = \left\langle \frac{\partial A}{\partial a_{pq}} X_i, AX_i + b + Y_i \right\rangle = \langle x_{ip} e_p, AX_i + b + Y_i \rangle$$

$$= x_{ip}(e_p^T AX_i + e_p^T b - e_p^T Y_i) = x_{ip}(\sum_{k=1}^{n} a_{pk} x_{ik} + b_p - y_{ip}),$$

where $e_p = (\ 0 \quad 0 \quad \dots \quad 1_p \quad 0 \quad \dots \quad 0\ )^T$, $X_i = (\ x_{i1} \quad x_{i2} \quad \dots \quad x_{in}\ )^T$ and $Y_i = (\ y_{i1} \quad y_{i2} \quad \dots \quad y_{in}\ )^T$

It is implies that

$$\frac{\partial F}{\partial a_{pq}} = \sum_{r=1}^{m} t_i \frac{\partial F_i}{\partial a_{pq}} = \sum_{i=1}^{m} t_i x_{ip} \sum_{k=1}^{n} (a_{pk} x_{ik} + b_p - y_{ip})$$

$$= \sum_{k=1}^{n} (\sum_{i=1}^{m} t_i x_{ip} x_{ik}) a_{pk} + (\sum_{i=1}^{m} t_i x_{ip}) b_q - \sum_{i=1}^{m} t_i x_{ip} y_{ip} = 0.$$

By using 2.5(2) we have that:

$$\frac{\partial F_i}{\partial b_p} = \langle e_p, AX_i + b + Y_i \rangle = e_p^T AX_i + e_p^T b - e_p^T Y_i$$

$$= \begin{pmatrix} a_{p1} & a_{p2} & \dots & a_{pn} \end{pmatrix} \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{in} \end{pmatrix} + b_p - y_{ip} = \sum_{k=1}^{n} a_{pk} x_{ik} + b_p - y_{ip}.$$

This is implies that:

$$\frac{\partial F}{\partial b_p} = \sum_{i=1}^{m} t_i \frac{\partial F_i}{\partial b_p} = \sum_{i=1}^{m} t_i (\sum_{k=1}^{n} a_{pk} x_{ik} + b_p - y_{ip}) = \sum_{k=1}^{n} (\sum_{i=1}^{m} t_i x_{ik}) a_{pk} + b_p - \sum_{i=1}^{m} t_i y_{ip} = 0.$$

Now we obtain two equations:

$$\begin{cases} \sum_{k=1}^{n} (\sum_{i=1}^{m} t_i x_{iq} x_{ik}) a_{pk} + (\sum_{i=1}^{m} t_i x_{iq}) b_q = \sum_{i=1}^{m} t_i x_{iq} y_{ip} \quad (1) \\ \sum_{k=1}^{n} (\sum_{i=1}^{m} t_i x_{ik}) a_{pk} + b_p = \sum_{i=1}^{m} t_i y_{ip} \quad (2). \end{cases} \tag{3.4}$$

As you know, we need separation of variables: $A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$ and $b =$

$(\ b_{11} \quad b_{21} \quad \dots \quad b_{n1} \ )^T$ out, so that we can do the calculation easily though the computer. So we obtain this form from the equation system 2.4 as follow:

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_p \\ a_{21} & a_{22} & \dots & a_{2n} & b_p \\ \vdots & \vdots & \ddots & \vdots & b_p \\ a_{n1} & a_{n2} & \dots & a_{nn} & b_p \end{pmatrix} \times$$

$$\begin{pmatrix} \sum_{i=1}^{m} t_i x_{i1} x_{i1} & \sum_{i=1}^{m} t_i x_{i2} x_{i1} & \dots & \sum_{i=1}^{m} t_i x_{in} x_{i1} & \sum_{i=1}^{m} t_i x_{i1} \\ \sum_{i=1}^{m} t_i x_{i1} x_{i2} & \sum_{i=1}^{m} t_i x_{i2} x_{i2} & \dots & \sum_{i=1}^{m} t_i x_{in} x_{i2} & \sum_{i=1}^{m} t_i x_{i2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sum_{i=1}^{m} t_i x_{i1} x_{in} & \sum_{i=1}^{m} t_i x_{i2} x_{in} & \dots & \sum_{i=1}^{m} t_i x_{in} x_{in} & \sum_{i=1}^{m} t_i x_{in} \\ \sum_{i=1}^{m} t_i x_{i1} & \sum_{i=1}^{m} t_i x_{i2} & \dots & \sum_{i=1}^{m} t_i x_{in} & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \sum_{i=1}^{m} t_i x_{i1} y_{i1} & \sum_{i=1}^{m} t_i x_{i2} y_{i1} & \dots & \sum_{i=1}^{m} t_i x_{in} y_{i1} & \sum_{i=1}^{m} t_i y_{i1} \\ \sum_{i=1}^{m} t_i x_{i1} y_{i2} & \sum_{i=1}^{m} t_i x_{i2} y_{i2} & \dots & \sum_{i=1}^{m} t_i x_{in} y_{i2} & \sum_{i=1}^{m} t_i y_{i2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sum_{i=1}^{m} t_i x_{i1} y_{in} & \sum_{i=1}^{m} t_i x_{i2} y_{in} & \dots & \sum_{i=1}^{m} t_i x_{in} y_{in} & \sum_{i=1}^{m} t_i y_{in} \end{pmatrix} \tag{3.5}$$

Now we assume that

$$M = \begin{pmatrix} \sum_{i=1}^{m} t_i x_{i1} x_{i1} & \sum_{i=1}^{m} t_i x_{i2} x_{i1} & \dots & \sum_{i=1}^{m} t_i x_{in} x_{i1} & \sum_{i=1}^{m} t_i x_{i1} \\ \sum_{i=1}^{m} t_i x_{i1} x_{i2} & \sum_{i=1}^{m} t_i x_{i2} x_{i2} & \dots & \sum_{i=1}^{m} t_i x_{in} x_{i2} & \sum_{i=1}^{m} t_i x_{i2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sum_{i=1}^{m} t_i x_{i1} x_{in} & \sum_{i=1}^{m} t_i x_{i2} x_{in} & \dots & \sum_{i=1}^{m} t_i x_{in} x_{in} & \sum_{i=1}^{m} t_i x_{in} \\ \sum_{i=1}^{m} t_i x_{i1} & \sum_{i=1}^{m} t_i x_{i2} & \dots & \sum_{i=1}^{m} t_i x_{in} & 1 \end{pmatrix}$$

$$N = \begin{pmatrix} \sum_{i=1}^{m} t_i x_{i1} y_{i1} & \sum_{i=1}^{m} t_i x_{i2} y_{i1} & \dots & \sum_{i=1}^{m} t_i x_{in} y_{i1} & \sum_{i=1}^{m} t_i y_{i1} \\ \sum_{i=1}^{m} t_i x_{i1} y_{i2} & \sum_{i=1}^{m} t_i x_{i2} y_{i2} & \dots & \sum_{i=1}^{m} t_i x_{in} y_{i2} & \sum_{i=1}^{m} t_i y_{i2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sum_{i=1}^{m} t_i x_{i1} y_{in} & \sum_{i=1}^{m} t_i x_{i2} y_{in} & \dots & \sum_{i=1}^{m} t_i x_{in} y_{in} & \sum_{i=1}^{m} t_i y_{in} \end{pmatrix}$$

$$B = \begin{pmatrix} s_1 x_{11} & s_2 x_{21} & \dots & s_m x_{m1} \\ s_1 x_{12} & s_2 x_{22} & \dots & s_m x_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ s_1 x_{1n} & s_2 x_{2n} & \dots & s_m x_{mn} \\ s_1 & s_2 & \dots & s_m \end{pmatrix}$$

$$C = \begin{pmatrix} s_1 y_{11} & s_2 y_{21} & \dots & s_m y_{m1} \\ s_1 y_{12} & s_2 y_{22} & \dots & s_m y_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ s_1 y_{1n} & s_2 y_{2n} & \dots & s_m y_{mn} \end{pmatrix}$$

Above $t_i = s_i^2$ and m $\geq$ n+1, now we get a very good optimization for the equation system 2.6 such as $M = B * B^T$,$N = C * B^T$ and X=[A b] with $X * B * B^T$=C*B^T. Furthermore, as m $\geq$ n+1 we can obtain a unique solution for X. This is because Rank(B)=n+1 as m $\geq$ n+1, then Rank(M)=Rank($B * B^T$)=n+1.

In fact, as we use affine transformations, Rank(A)=2, Rank(b)=2, with

$$X = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \end{pmatrix},$$

$$B = \begin{pmatrix} s_1 x_{11} & s_2 x_{21} & \dots & s_m x_{m1} \\ s_1 x_{12} & s_2 x_{22} & \dots & s_m x_{m2} \\ s_1 & s_2 & \dots & s_m \end{pmatrix}$$

and

$$C = \begin{pmatrix} s_1 y_{11} & s_2 y_{21} & \dots & s_m y_{m1} \\ s_1 y_{12} & s_2 y_{22} & \dots & s_m y_{m2} \end{pmatrix},$$

where m $\geq$ 3. So, we need choose at least 3 corresponding points for each feature point. As a matter of fact, we choose at least 4 corresponding points for each feature point and we set t_1 =0.9 since this is the intersection point of two curves.

3.4 Delaunay Triangulation

In this section, we will talk about the Delaunay triangulation.

In the last section, we are sampling some feature points, include the four corners, then we need obtain a partition that depends on our feature points and four corner points. It is because this method is based on the local transformation, those corresponding points and corner points have to be the interpolation points to control the interpolation function in each local area. This implies that the partition of the given image is dependent on those points. There is a convenient way to do the partition with breast MRI - triangulation. As we know, the triangle is the simplest polygon, so it is easy for us to get the partition with a triangle. There is a proven technique of triangulation method to deal with our triangulation problem, named Delaunay Triangulation.

Definition3.2: In mathematics and computational geometry, a Delaunay triangulation for a set P of points in a plane is a triangulation DT(P) such that no point in P is inside the circumcircle of any triangle in DT(P), see Figure 2.8.

Here are some properties, which is suitable for our method, about Delaunay triangulation:

1. Convex hull property (I will use it in next subsection)
2. Circumcircle property: In the Delaunay Triangulation, the circumcircle of any triangle is

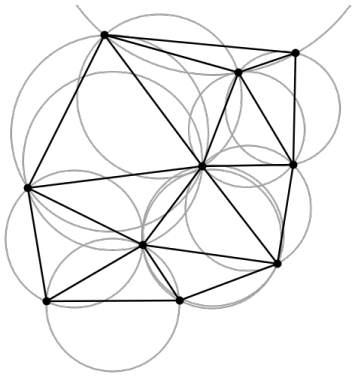

Figure 3.8: Empty Circumcircle of Delaunay Triangulation

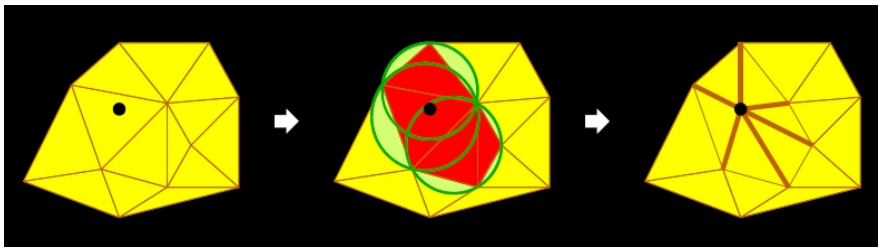

Figure 3.9: Delaunay Triangulation Schematic Diagram

empty (contains no interpolation points).

3. Closest pair property: The closest pair of interpolation points are neighbors in the Delaunay triangulation.

4. Locally: When we add a new interpolation point, the triangulation is only changed in a local area of this point, see Figure 3.9.

5. Unique: Under the same interpolation point set, the Delaunay triangulation is unique.

It is clear that the Delaunay triangulation is a kind of partition determined by the interpolation points and it is unique, so it has a good property for our algorithm.

How to get the Delaunay triangulation under our center points of feature points:

1. Use four corner points to get a big triangle which is content themselves(this ptriangle include all the center points).

2. Adding points: put one point in this triangle. This point should belong to some triangle. Then we do the empty Circumcircle test, remove the triangle which is on longer Delaunay,

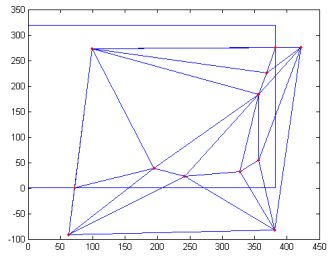

Figure 3.10: Delaunay Triangulation

and retriangulate the cavity with a fan around the new vertex, then we get a new Delaunay Triangulation, see Figure 3.9.

3. Adding the points one by one and do the step 2. See Figure 3.10

3.5 Application Of Multivariate Spline Interpolation

In this section, the multivariate spline function will be discussed by using matrix variable.

In the previous section, I take org168 (Figure 3.4) and org911(Figure 3.6) as input image, I will deform org911 and this source image with the deforming mapping or multivariate spline function:

$$S : Source\ Image(I_2) \rightarrow Output\ Image(R), denote\ by\ S(x,y) = (x_o, y_o), \tag{3.6}$$

$$S(\triangle) = \begin{cases} S_1(x,y) & (x,y) \in D_1 \\ S_2(x,y) & (x,y) \in D_2 \\ \vdots & \\ S_n(x,y) & (x,y) \in D_N \end{cases}$$

$$S_i(x,y) = \begin{pmatrix} a_{11_i}(x,y) & a_{12_i}(x,y) & b_{1_i}(x,y) \\ a_{21_i}(x,y) & a_{22_i}(x,y) & b_{2_i}(x,y) \end{pmatrix} \tag{3.7}$$

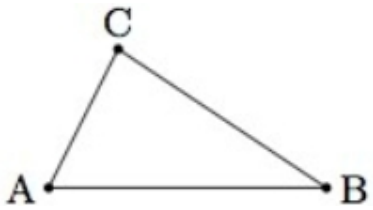

Figure 3.11: Unit Triangle

where $\triangle$ is Delaunay Triangulation of input image, denote by D_1, D_2, ..., D_N, N is the number of triangles in the partition, $(x, y)\in$ source image, $(x_o,y_o) \in$ Output Image and $a_{11_i}(x, y)$, $a_{21_i}(x, y)$, $a_{12_i}(x, y)$, $a_{22_i}(x, y)$, $b_{1_i}(x, y)$, $b_{2_i}(x, y) \in P_2$ for all i=1, 2, ..., N.
On the other hand, there is no change with org168 except translation so I call it reference image with the mapping:

$$T : Reference\ Image(I_1) \rightarrow Output\ Image(R), denote\ by\ T(x, y) = (x_o, y_o), \tag{3.8}$$

where $(x, y)\in$ reference image and $(x_o,y_o) \in$ Output Image.

As we know, multivariate spline function is a piecewise polynomial, and we already have the domain of very piecewise by using Delaunay triangulation in the last section. Here, we need take care that all coefficients of the spline function is 2 by 3 matrix. Now, take any triangle in the Delaunay triangulation for example see Figure 3.11. Now, the restricted expression of the multivariate spline function S under the $\triangle$ABC is very important.

Clearly, A, B and C are the interpolation points, as we used center point of feature point as interpolation point. In section 3.3, we already got the value for all interpolation points. In order to get an appropriate and more accurate transformation, we will also initialize the partial differential value on each interpolation point(see equation 3.11 and 3.12).

$$\frac{\partial S(x, y)}{\partial x} = \begin{pmatrix} \frac{\partial a_{11_i}(x,y)}{\partial x} & \frac{\partial a_{12_i}(x,y)}{\partial x} & \frac{\partial b_{1_i}(x,y)}{\partial x} \\ \frac{\partial a_{21_i}(x,y)}{\partial x} & \frac{\partial a_{22_i}(x,y)}{\partial x} & \frac{\partial b_{2_i}(x,y)}{\partial x} \end{pmatrix} \tag{3.9}$$

$$\frac{\partial S(x,y)}{\partial y} = \begin{pmatrix} \frac{\partial a11_i(x,y)}{\partial y} & \frac{\partial a12_i(x,y)}{\partial y} & \frac{\partial b1_i(x,y)}{\partial y} \\ \frac{\partial a21_i(x,y)}{\partial y} & \frac{\partial a22_i(x,y)}{\partial y} & \frac{\partial b2_i(x,y)}{\partial y} \end{pmatrix} \tag{3.10}$$

In this way, we have three vertices with two partial derivatives and their value as given conditions. Totally, we have nine matrices given conditions in this triangle domain. We can use binary quadratic polynomial as an interpolation function on this triangle domain, and use the Powell-Sabin Scheme to do the refinement on this triangle domain. It is because binary quadratic polynomials have six degrees of freedom for its coefficients and binary cubic polynomials have ten degrees of freedom for its coefficients, so we choose binary quadratic polynomial as an interpolation function with Powell-Sabin Scheme. This is also one reason that we use the Delaunay triangulation as the partition method for our algorithm. As we know, Powell-Sabin Scheme has been proved in function form, now it will be extended to matrix form so that it can be used for our algorithm.

3.5.1 Generalization Of Powell-Sabin Scheme On Matrix Variable

In this section, we will extend the Powell-Sabin Scheme to matrix form. No matter in theory or in practical application, this work is very useful. Here is our problem, we need to find a suitable way to refine the triangle, like Powell-Sabin Scheme. As we know, there are four methods to refine the triangle, see Figure 3.12-3.15 for the triangle refinements. In figure

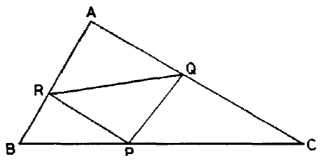

Figure 3.12: Triangle Refinement Method 1

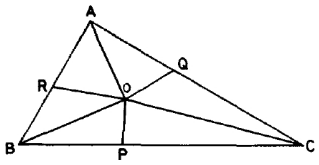

Figure 3.13: Triangle Refinement Method 2

3.12-3.15, we require $\Phi(x,y) = S_i(x,y)$ to be a quadratic polynomial on every small triangle and belong to $C^1[\triangle \text{ABC}]$, hence, $a_{11_i}(x,y)$, $a_{12_i}(x,y)$, $b_{1_i}(x,y)$, $a_{21_i}(x,y)$, $a_{22_i}(x,y)$ and $b_{2_i}(x,y) \in C^1[\triangle \text{ABC}]$. In method 1, 2 and 4, the point P, Q and R are arbitrarily chosen on

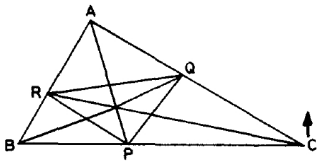

Figure 3.14: Triangle Refinement Method 3

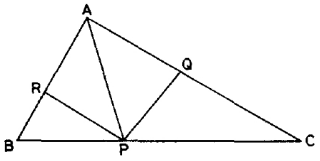

Figure 3.15: Triangle Refinement Method 4

the triangle side. In method 2, point O is also an arbitrary interior point of the triangle. In method 3, AP, BQ and CR are concurrent.

In method 1, we subdivide triangle into four triangle. This subdivision is not suitable since it has only 8*6 degrees of freedom, but we have totally 9*6 given conditions, as each coefficient has 6 degrees of freedom. So this triangle should be divided into more than four parts. Method 2 is suitable which will proved later. Method 4, may not be symmetric when we put all the Delaunay triangle together, is not suitable neither. In Method 3, there are 12 small triangles, and the degree of freedom is 12*6, so usually we can not get a unique solution with 9*6 given conditions, therefore, this method is not suitable.

Theorem 3.5.1: There is a uniqueness binary quadratic polynomial interpolation function by using method 2.

Now, we show that method 2 is suitable. we have six given condition such as $S(A)$, $S(B)$, $S(C)$, $\frac{\partial S(A)}{\partial x}$, $\frac{\partial S(B)}{\partial x}$, $\frac{\partial S(C)}{\partial x}$, $\frac{\partial S(A)}{\partial y}$, $\frac{\partial S(B)}{\partial y}$ and $\frac{\partial S(C)}{\partial y}$ and all of them are 2 by 3 matrix. Assume all of them is zero. Then $a_{11_i}(A)$=0, $a_{11_i}(B)$=0, $a_{11_i}(C)$=0, $\frac{\partial a_{11_i}(A)}{\partial x}$=0, $\frac{\partial a_{11_i}(B)}{\partial x}$=0, $\frac{\partial a_{11_i}(C)}{\partial x}$=0, $\frac{\partial a_{11_i}(A)}{\partial y}$=0, $\frac{\partial a_{11_i}(B)}{\partial y}$=0 and $\frac{\partial a_{11_i}(C)}{\partial y}$=0. Firstly, show $a_{11_i}(x, y)$=0.

1. Show that $a_{11_i}(x, y) \mid_{BC}$=0

Without loss of generality, assume l_{BC} on the X-axes, or translate it if not. Then

$$\begin{cases} a_{11_i}(x, y) \mid_{BP} = ax^2 + bx + c, & x \in BP \\ a_{11_i}(x, y) \mid_{CP} = a_{11_i}(x, y) \mid_{BP} + \lambda(x - x_p)^2, & x \in CP \end{cases}$$

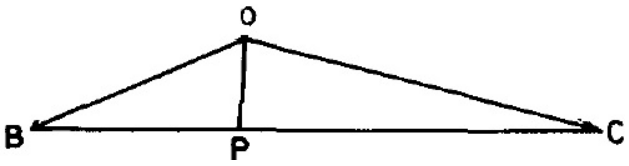

Figure 3.16: Triangle Refinement Method2 Subdivision OBC

Since $a_{11_i}(B)\mid_{BP}$=0, $a'_{11_i}(B)\mid_{BP}$=0, $a_{11_i}(C)\mid_{CP}$=0 and $a'_{11_i}(C)\mid_{CP}$=0, then a=b=c=λ=0.

Therefore $a_{11_i}(x,y)\mid_{BC}$=0.

2. Show that $\nabla a_{11_i}(x,y)\mid_{BC}$=0, see Figure 3.15. Assume $a_{11_i}(x,y)\mid_{OPB}=P_1$ and $a_{11_i}(x,y)\mid_{OPC}=P_2$.

Since $a_{11_i}(x,y)\mid_{BC}$=0, by using Lemma 1.1, we have

$$\begin{cases} P_1 = P_b(x,y)l_{BP}, & \text{(x,y)}\in\triangle\text{OBP and } P_b(x,y)\in P_1 \\ P_2 = P_c(x,y)l_{CP}, & \text{(x,y)}\in\triangle\text{ OCP and } P_c(x,y)\in P_1 \\ P_1 - P_2 = \lambda(l_{OP})^2 \end{cases}$$

We consider the directional derivative along OP, then: $D_{OP}P_1 = D_{OP}P_2 + \lambda D_{OP}(l_{OP})^2$.

Assume l_{OP}:$mx+ny+l=0$, then the direction vector is $(n,-m)$. Now we have that:

$$\lambda D_{OP}(l_{OP})^2 = \lambda(2*l_{OP}*m*n - 2*l_{OP}*m*n) = 0$$

We obtain that $D_{OP}P_1 = D_{OP}P_2$, As $P_1\in\prod_3$, then $D_{OP}P_1 = D_{OP}P_2\in\prod_2$.

Assume $D_{OP}P_1$ = ax+b, as $D_{OP}P_1(B)$=0 and $D_{OP}P_2(C)$=0, then a=b=0.

Therefore $D_{OP}P_1 = D_{OP}P_2$=0, then $\nabla a_{11_i}(x,y)\mid_{BC}$=0.

3. Show that $P_1 = P_2$.

As $\nabla a_{11_i}(x,y)\mid_{BC}$=0, similarly, we can proof $a_{11_i}(x,y)$ and $\nabla a_{11_i}(x,y)$ are zero on the perimeter of triangle ABC. By using lemma 1.1, it means that:

$$\begin{cases} P_1 = \lambda_1 * (l_{BC})^2, & \text{(x,y)}\in\triangle\text{OBP} \\ P_2 = \lambda_2 * (l_{BC})^2, & \text{(x,y)}\in\triangle\text{ OCP} \\ P_1 - P_2 = \lambda (l_{OP})^2 \end{cases}$$

Hence, (λ_1-λ_2) $(l_{BC})^2 = \lambda (l_{OP})^2$. As we know, $((l_{BC})^2,\ (l_{OP})^2) = 1$, otherwise, $P_1 = P_2 = 0$. Since (λ_1-λ_2) $(l_{BC})^2$ - $\lambda (l_{OP})^2 \equiv 0$, then $\lambda_1 = \lambda_2$ and $\lambda = 0$. Therefore $P_1 = P_2$.

4. Show that $a_{11_i}(x,y)$=0.

Since $P_1 = P_2$, $\nabla a_{11_i}(x,y)\mid_{BC}$=0 and $a_{11_i}(x,y)\mid_{BC}$=0, then $a_{11_i}(x,y)\mid_{OBC} = \lambda_1 (l_{BC})^2$. Similarly, we can obtain that:

$$\begin{cases} a_{11_i}(x,y)\mid_{OBC} = \lambda_1 (l_{BC})^2, & (1) \\ a_{11_i}(x,y)\mid_{OAC} = \lambda_2 (l_{AC})^2, & (2) \\ a_{11_i}(x,y)\mid_{OBA} = \lambda_3 (l_{BA})^2, & (3) \end{cases}$$

By using Theorem 1.3, we can obtain that:

$$\begin{cases} a_{11_i}(x,y)\mid_{OBC} - a_{11_i}(x,y)\mid_{OAC} = \lambda_4 (l_{OC})^2, & (4) \\ a_{11_i}(x,y)\mid_{OAC} - a_{11_i}(x,y)\mid_{OBA} = \lambda_5 (l_{OA})^2, & (5) \\ a_{11_i}(x,y)\mid_{OBA} - a_{11_i}(x,y)\mid_{OBC} = \lambda_6 (l_{BO})^2, & (6) \end{cases}$$

Now, our subdivision reduce as Figure 3.16. By using (1), (2) and (4), we have $\lambda_1 (l_{BC})^2$ - $\lambda_2 (l_{AC})^2 = \lambda_4 (l_{OC})^2$. Since l_{BC}, l_{AC} and l_{OC} are in the same plane, then there exists m and n, such that l_{OC} = m l_{BC} + n l_{AC}. Substituted it, we can obtain that:

$$\lambda_1 (l_{BC})^2 - \lambda_2 (l_{AC})^2 = \lambda_4 * (m * l_{BC} + n * l_{AC})^2$$

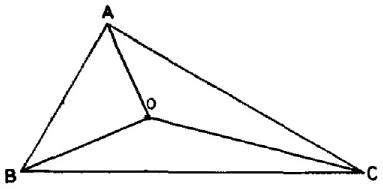

Figure 3.17: Triangle Refinement Method2 Retrograde Triangle

$$\lambda_1(l_{BC})^2 - \lambda_2(l_{AC})^2 - \lambda_4 * (m * l_{BC} + n * l_{AC})^2 \equiv 0$$

$$(\lambda_1 - \lambda_4 m)(l_{BC})^2 - (\lambda_2 + \lambda_4 n)(l_{AC})^2 - 2\lambda_4 mn l_{BC} l_{AC} \equiv 0$$

As we know, (l_{BC}, l_{AC}) = 1, otherwise, BC and AC coincide. For this reason, we obtain that ((l_{BC})^2, (l_{AC})^2, $l_{BC}l_{AC}$) = 1. This is implies $\lambda_1 - \lambda_4 m = \lambda_2 + \lambda_4 n = \lambda_4 mn = 0$.

Now, we consider $\lambda_4 mn = 0$.

Case 1. $mn \neq 0$, then $\lambda_4 = 0$. By using (1), (2) and (4), we can obtain $\lambda_1 = \lambda_2 = 0$. Therefore $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = \lambda_5 = \lambda_6 = 0$. Therefore $a_{11_i}(x, y)$=0.

Case 2. $mn = 0$, assume m = 0. Then l_{OC} = n l_{AC}. This implies that OC and AC coincide, clearly, $a_{11_i}(x, y) \mid_{OAC} = 0$. and we can also obtain that:

$$\begin{cases} a_{11_i}(x, y) \mid_{OBC} = \lambda_1(l_{BC})^2, & a_{11_i}(x, y) \mid_{OBC} = \alpha_1(l_{AC})^2. \\ a_{11_i}(x, y) \mid_{OBA} = \lambda_3(l_{BA})^2, & a_{11_i}(x, y) \mid_{OBA} = \alpha_3(l_{AC})^2. \end{cases} \tag{3.11}$$

As we know, (l_{BC}, l_{AC}) = 1 and (l_{BA}, l_{AC}) = 1, then by using equation 2.13 we can obtain that $\lambda_1 = \lambda_3 = 0 .\!.\!\because a_{11_i}(x, y)$=0.

In this way, we can proof that $a_{11_i}(x, y)$, $a_{12_i}(x, y)$, $b_{1_i}(x, y)$, $a_{21_i}(x, y)$, $a_{22_i}(x, y)$ and $b_{2_i}(x, y)$ $\equiv 0$. So, under this conditions, we can obtain S_i(x,y) $\equiv 0$.

As we know, by using our given conditions, we can obtain a linear equations system, denote by $RX = d$, where R is a (9*6) by (9*6) matrix, X and d are a (9*6) by 1 matrices. What we done here is that: assume $d = 0$, then $X = 0$. So, R is a non-single matrix. By using

this subdivision (method 2), the interpolation function is uniqueness.

3.5.2 Expression Of Multi-Variate Spline Function

In this section, we will show how to calculate the multi-variate spline function.

From last section, we know that method 2 (Figure 3.13) is a suitable subdivision. Since the point O is arbitrarily chosen in method 2, now we choose it as the incenter for each triangle(See Figure 3.18).

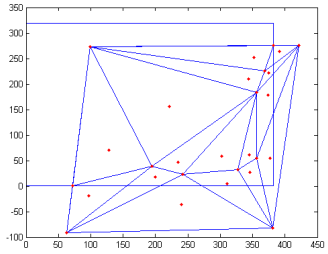

Figure 3.18: Triangle Refinement Method2 Use Incenter

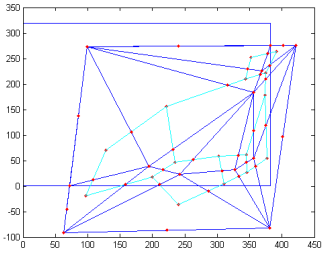

Figure 3.19: Triangle Refinement Method2 Subdivision

We connected the incenter for each two triangles which has common edge and denote the intersection point as the point in that triangle's edge. In this way, we obtain the subdivision for the Delaunay Triangulation, see Figure 3.19(didn't connect incenter with three vertices). Here is the formal to calculate the incenter of triangle:

$$O = \frac{aA + bB + cC}{a + b + c}$$

Where O, A, B and C are coordinate for those four points, $a =\mid BC \mid$, $b =\mid$ AC $\mid$ and $c =\mid$ AB $\mid$. Before we calculate the multi-variate spline function for our algorithm, we need to introduce the barycentric coordinate system which is defined by August Ferdinand Mbius, as we will calculate the multi-variate spline function under the barycentric coordinate system.

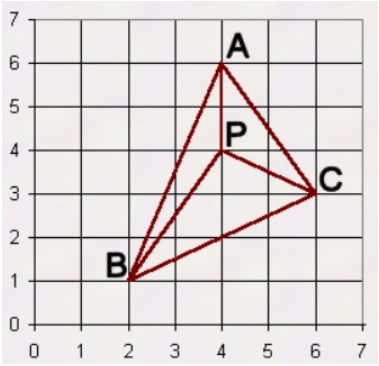

Figure 3.20: Unit Triangle in Euclidean Coordinates

Barycentric coordinates are also known as area coordinates, because the coordinates of P with respect to triangle ABC are proportional to the (signed) areas of PBC, PCA and PAB(see Figure 3.20). Assume the coordinate of point P=(α_1, α_2, α_3), then

$$\alpha_1 = \frac{\begin{vmatrix} 1 & 1 & 1 \\ x_p & x_B & x_C \\ y_p & y_B & y_C \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_C \\ y_A & y_B & y_C \end{vmatrix}} \quad \alpha_2 = \frac{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_p & x_C \\ y_A & y_p & y_C \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_C \\ y_A & y_B & y_C \end{vmatrix}} \quad \alpha_3 = \frac{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_p \\ y_A & y_B & y_p \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_C \\ y_A & y_B & y_C \end{vmatrix}} \tag{3.12}$$

This is implies that:

1. $\alpha_1 + \alpha_2 + \alpha_3 = 1$.

2. The coordinate of three vertices are (1, 0, 0), (0, 1, 0) and (0, 0, 1).

3. For any given point P $\in \triangle$ ABC, there exist u, v, w ≥ 0 and satisfy property (1), s.t. P $= \alpha_1$A + α_2B + α_3C.

As we know, in a Euclidean coordinate system, the coordinates always change, when we worked on different triangle. But we do not need to worry about it in a barycentric coordinate system, since the coordinates of three vertices will never change and the coordinates of all

inside point is a linear combination of the coordinates of three vertices. More over, by using the above properties, we can obtain some relationship between those two systems. Here are two lemmas about the direction differential in a barycentric coordinate system.

Lemma3.5.1[2]:

$$D_{T_j-T_i}\alpha_k = \begin{cases} 1, & \text{if j=k} \\ -1, & \text{if i=k} \\ 0, & \text{otherwise} \end{cases} \tag{3.13}$$

where $(x, y) \in \mathbf{R}^2$ and (α_1, α_2, α_3) $\in$ barycentric coordinate system, T_1, T_2, T_3 are three vertices of the triangle.

Lemma3.5.2[2]: For any given function $f(x)$, let $y = x_1$ - x_2, denote $t^i = (\alpha_1^i, \alpha_2^i, \alpha_3^i)$ as the barycentric coordinate of x_i, and (β_1, β_2, β_3) $=t^1$ - t^2. Then we have:

$$D_y f(x) = D_{x_1-x_2} f(x) = \beta_1 \frac{\partial f(t)}{\partial \alpha_1} + \beta_2 \frac{\partial f(t)}{\partial \alpha_2} + \beta_3 \frac{\partial f(t)}{\partial \alpha_3} \tag{3.14}$$

Here, $f(t)$ is obtained by replacing argument x in $f(x)$ with its area coordinate t.

Now, we calculate the multi-variate spline function: We will use Method 2(see Figure 2.13), The given condition is S(A), S(B), S(C), $\frac{\partial S(A)}{\partial x}$, $\frac{\partial S(B)}{\partial x}$, $\frac{\partial S(C)}{\partial x}$, $\frac{\partial S(A)}{\partial y}$, $\frac{\partial S(B)}{\partial y}$ and $\frac{\partial S(C)}{\partial y}$ By using Lemma 2.5.2, we can obtain that all of the directional derivative of three vertices such as D_{A-R}S(A), D_{A-O}S(A), D_{A-Q}S(A), D_{B-R}S(B), D_{B-O}S(B), D_{B-P}S(B), D_{C-P}S(C), D_{C-O}S(C), D_{C-Q}S(C). Firstly, we calculate the value and directional derivative for points P, Q, R, O. By using Theorem 1.4 we can obtain that:

1. Point O with its coordinate (α_O, β_O, γ_O):

$$\begin{cases} 2S(O) + D_{A-O}S(O) = 2S(A) + D_{O-A}S(A) & (1) \\ 2S(O) + D_{B-O}S(O) = 2S(B) + D_{O-B}S(B) & (2) \\ 2S(O) + D_{C-O}S(O) = 2S(C) + D_{O-C}S(C) & (3) \\ \alpha_O D_{A-O}S(O) + \beta_O D_{B-O}S(O) + \gamma_O D_{C-O}S(O) = 0 & (4) \end{cases}$$

By using (1)$*\alpha_O$ + (2)$*\beta_O$ + (3)$*\gamma_O$ - (4), we obtain:

$$S(O) = \frac{\alpha_O(2S(A) + D_{O-A}S(A)) + \beta_O(2S(B) + D_{O-B}S(B)) + \gamma_O(2S(C) + D_{O-C}S(C))}{2}$$

Hence, we can obtain:

$$D_{A-O}S(O) = 2S(A) + D_{O-A}S(A) - 2S(O)$$
$$D_{B-O}S(O) = 2S(B) + D_{O-B}S(B) - 2S(O)$$
$$D_{C-O}S(O) = 2S(C) + D_{O-C}S(C) - 2S(O)$$

2. Point P with its coordinate (0, β_P, γ_P), here, $\alpha_P = 0$:

$$\begin{cases} 2S(P) + D_{O-P}S(P) = 2S(O) + D_{P-O}S(O) & (1) \\ 2S(P) + D_{B-P}S(P) = 2S(B) + D_{P-B}S(B) & (2) \\ 2S(P) + D_{C-P}S(P) = 2S(C) + D_{P-C}S(C) & (3) \\ \beta_P D_{B-P}S(P) + \gamma_P D_{C-P}S(P) = 0 & (4) \end{cases}$$

By using (1)$*\alpha_P$ + (2)$*\beta_P$ + (3)$*\gamma_P$ - (4), we obtain:

$$S(P) = \frac{\alpha_P(2S(O) + D_{P-O}S(O)) + \beta_P(2S(B) + D_{P-B}S(B)) + \gamma_P(2S(C) + D_{P-C}S(C))}{2}$$

Hence, we can obtain:

$$D_{O-P}S(P) = 2S(O) + D_{P-O}S(O) - 2S(P)$$
$$D_{B-P}S(P) = 2S(B) + D_{P-B}S(B) - 2S(P)$$
$$D_{C-P}S(P) = 2S(C) + D_{P-C}S(C) - 2S(P)$$

3. Point Q with its coordinate (α_Q, 0, γ_Q), here, β_Q=0:

$$\begin{cases} 2S(Q) + D_{A-Q}S(Q) = 2S(A) + D_{Q-A}S(A) & (1) \\ 2S(Q) + D_{O-Q}S(Q) = 2S(O) + D_{Q-O}S(O) & (2) \\ 2S(Q) + D_{C-Q}S(Q) = 2S(C) + D_{Q-C}S(C) & (3) \\ \alpha_Q D_{A-Q}S(Q) + \beta_Q D_{B-Q}S(Q) + \gamma_Q D_{C-Q}S(Q) = 0 & (4) \end{cases}$$

By using (1)$*\alpha_Q$ + (2)$*\beta_Q$ + (3)$*\gamma_Q$ - (4), we obtain:

$$S(Q) = \frac{\alpha_Q(2S(A) + D_{Q-A}S(A)) + \beta_Q(2S(O) + D_{Q-O}S(O)) + \gamma_Q(2S(C) + D_{Q-C}S(C))}{2}$$

Hence, we can obtain:

$$D_{A-Q}S(Q) = 2S(A) + D_{Q-A}S(A) - 2S(Q)$$
$$D_{O-Q}S(Q) = 2S(O) + D_{Q-O}S(O) - 2S(Q)$$
$$D_{C-Q}S(Q) = 2S(C) + D_{Q-C}S(C) - 2S(Q)$$

4. Point R with its coordinate (α_R, β_R, 0), here, γ_R=0:

$$\begin{cases} 2S(R) + D_{A-R}S(R) = 2S(A) + D_{R-A}S(A) & (1) \\ 2S(R) + D_{B-R}S(R) = 2S(B) + D_{R-B}S(B) & (2) \\ 2S(R) + D_{O-R}S(R) = 2S(O) + D_{R-O}S(O) & (3) \\ \alpha_R D_{A-R}S(R) + \beta_R D_{B-R}S(R) + \gamma_R D_{C-R}S(R) = 0 & (4) \end{cases}$$

By using (1)$*\alpha_R$ + (2)$*\beta_R$ + (3)$*\gamma_R$ - (4), we obtain:

$$S(R) = \frac{\alpha_R(2S(A) + D_{R-A}S(A)) + \beta_R(2S(B) + D_{R-B}S(B)) + \gamma_R(2S(O) + D_{R-O}S(O))}{2}$$

Hence, we can obtain:

$$\begin{aligned} D_{A-R}S(R) &= 2S(A) + D_{R-A}S(A) - 2S(R) \\ D_{B-R}S(R) &= 2S(B) + D_{R-B}S(B) - 2S(R) \\ D_{O-R}S(R) &= 2S(O) + D_{R-O}S(O) - 2S(R) \end{aligned}$$

By using the above method and under the barycentric coordinate system, it is easy to find that we obtain a similar form results, it is a pretty form, and it's easy for program implementation.

Second Step: Now, we consider the unit triangle, a general piece after subdivision(see Figure 3.21).

Assume the given condition on this unit are S(u), S(v), S(w), D_{u-v}S(v), D_{v-w}S(w) and D_{w-u}S(u). As a matter of fact, we know all the vertices' value and directional derivative. Let

$$S|_{\triangle uvw} = P(x_1, x_2, x_3) = ax_1^2 + bx_1x_2 + cx_2^2 + dx_1x_2 + ex_2x_3 + fx_3^2 \qquad (3.15)$$

where a, b, c, d, e, f are 2 by 3 matrix and $x_1 + x_2 + x_3 = 1$. Then we can obtain that:

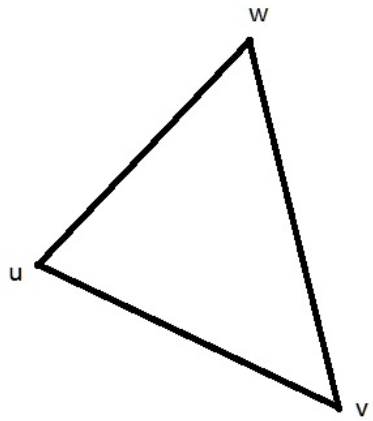

Figure 3.21: Unit Triangle in Barycentric Coordinate

$$\begin{aligned} P(u) = P(1,0,0) = a = S(u) \\ P(v) = P(0,1,0) = c = S(v) \\ P(w) = P(0,0,1) = f = S(w) \end{aligned} \tag{3.16}$$

By using Lemma 3.5.2, we obtain that:

1. u - v = (1, 0, 0) - (0, 1, 0) = (1, -1, 0), hence,

$$D_{u-v}P(v) = D_{u-v}P(0,1,0) = b - 2c = b - 2S(v) \tag{3.17}$$

So, b = $D_{u-v}P(v)$ + 2S(v).

2. v - w = (0, 1, 0) - (0, 0, 1) = (0, 1, -1)

$$D_{v-w}P(w) = D_{v-w}P(0,0,1) = e - 2f = e - 2S(w) \tag{3.18}$$

So, e = $D_{v-w}P(w)$ + 2S(w).

3. w - u = (0, 0, 1) - (1, 0, 0) = (-1, 0, 1)

$$D_{w-u}P(u) = D_{w-u}P(1,0,0) = d - 2a = d - 2S(u) \tag{3.19}$$

So, d = $D_{w-u}P(u)$ + 2S(u). By using the above method and under the barycentric coordinate system, it is easy to find that we still can obtain a similar form results for the final multi-variate spline function.

In this way, every point in the image(I_1) has an affine transformation which is given by the Multi-Variate Spline Function. By using the multi-variate spline functions, we can get our image registration result.

3.6 Size Determination

In this section, we will talk about the size of output image. Since we will use the inverse function of the multi-variate spline function to evaluate the gray value with the point on the output image. As we know, the input coordinate (x,y) $\in \mathbf{Z}\times\mathbf{Z}$, but output coordinate may not belongs to $\mathbf{Z}\times\mathbf{Z}$. So we use the inverse function of Multi-Variate Spline Function to evaluation the output image. When we get the corresponding coordinate in the input image, we can use some methods such as the nearest neighbor method, bilinear method and so on to get the gray value for output image.

Now, the most important thing is that to determine the size of output image, so that we can use the inverse function of Multi-Variate Spline Function to evaluation the gray value on the output image. As we know, the size of output image should be big enough for the Range(T) and Range(S).

Firstly, we assume T is identical mapping, then the Range(T)= Domain(T).

Secondly, we need calculate the Range(S). For the requirement and the convenience of our method, we will use the image of the four corner points in the source image as the corresponding four corner points in the output image.

Next, we need find the affine transformation for the four corner points.

I will use Shepard interpolation to find the affine transformation for the four corner points.

We denote P_i as my center point of feature points and A_i, b_i are the corresponding affine transformation for the center point of feature points, i = 1, 2, ..., n. The Shepard interpolation is given by :

$$\begin{cases} \Phi(Q_{ij}) = \frac{\sum_{i=1}^{n} d_i^{-1}[A_i \ b_i]}{\sum_{i=1}^{n} d_i^{-1}} \\ d_i = \sqrt{(x_{Q_{ij}} - x_{P_i})^2 + (y_{Q_{ij}} - y_{P_i})^2} \end{cases} \tag{3.20}$$

, where i,j = 1, 2 and d_i is the distance between Q_{ij} and P_i.

Now, we can do the calculation to determine the size of the output image:

Firstly, we calculate the corresponding points of each interpolation point in output image.

$$R_i = A_i * Q_i + b_i \tag{3.21}$$

, where Q_i is the coordinate of ith interpolation point.

Secondly, we put the four vertices of reference image into $\{R_i\}$.

As $\{R_i\}$ may have a negative coordinate, so we may translate the whole output image. Here is the size and translation:

$$\begin{aligned} xmin &= \min_{i,j}\{x_{R_{ij}}\} \qquad ymin = \min_{i,j}\{y_{R_{ij}}\} \\ xmax &= \max_{i,j}\{x_{R_{ij}}\} \qquad ymax = \max_{i,j}\{y_{R_{ij}}\} \end{aligned} \tag{3.22}$$

Hence, the size is equal to [xmax-xmin, ymax-ymin].

$$shiftx = |\min\{xmin,\ 0\}| \qquad shifty = |\min\{ymin,\ 0\}| \tag{3.23}$$

3.7 Valuation Scheme

In this section, I will show a method to do the assignment for the point in the output image. For the source image, it is a little difficult. Here are two methods:

Method 1: Do a loop for all the point in the source image, determine the corresponding triangle for the each point. We can use equation 3.12 to find the corresponding triangle:

$$\alpha_1 = \frac{\begin{vmatrix} 1 & 1 & 1 \\ x_p & x_B & x_C \\ y_p & y_B & y_C \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_C \\ y_A & y_B & y_C \end{vmatrix}} \quad \alpha_2 = \frac{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_p & x_C \\ y_A & y_p & y_C \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_C \\ y_A & y_B & y_C \end{vmatrix}} \quad \alpha_3 = \frac{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_p \\ y_A & y_B & y_p \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ x_A & x_B & x_C \\ y_A & y_B & y_C \end{vmatrix}}$$

If all of α_1, α_2 and α_3 are more than or equal to zero, then this is the corresponding triangle.

Algorithm 1:

1. Do the loop for all triangles in the Delaunay triangulation.
2. Use 3.12 to do the test.
3. If $\alpha_1 \geq 0$, $\alpha_2 \geq 0$ and $\alpha_3 \geq 0$
4. Do loop for the subdivision triangles
5. Use 3.12 to do the test.
6. If $\alpha_1 \geq 0$, $\alpha_2 \geq 0$ and $\alpha_3 \geq 0$
7. The point belongs to this triangle.
8. Else the point does not belong to this subdivision triangle.
9. Else the point does not belong to this subdivision triangle.
10. Use multi-variate spline function to do assignment for the point in output image.

Method 2: Do For Loop with each subdivision triangle, in each for loop we do the for loop with the point in this triangle. It is easy to do for loop with all subdivision triangle. Now we show how to do for loop with the point in the triangle:

Look at Figure 3.23, the line equations are as follow:

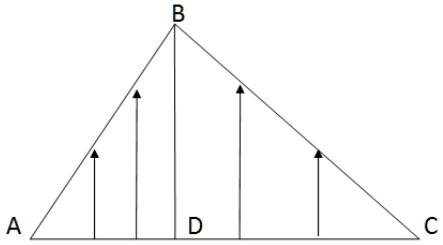

Figure 3.22: Valuation Scheme Case1

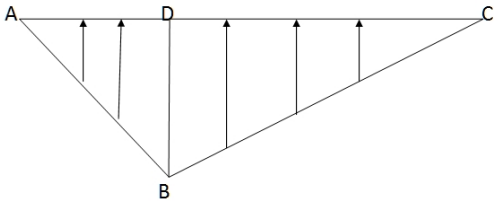

Figure 3.23: Valuation Scheme Case2

$$\begin{aligned} \ell_{AB}: \ f_{AB}(x) &= \tfrac{y_B - y_A}{x_B - x_A}(x - x_A) + y_A \\ \ell_{AC}: \ f_{AC}(x) &= \tfrac{y_C - y_A}{x_C - x_A}(x - x_A) + y_A \\ \ell_{BC}: \ f(x)_{BC} &= \tfrac{y_B - y_C}{x_B - x_C}(x - x_C) + y_C \end{aligned} \tag{3.24}$$

and the coordinate of D is $(x_D,\ f_{AC}(x_D))$.

Algorithms2:

1. Arrange the vertices by $x_A \leq x_B \leq x_C$.
2. If $y_B \leq f_{AC}(x_D)$
3. Use case2, p=-1, Else, use case1, p=1.
5. Do For Loop:
6. for i= $[x_A] : [x_D]$
7. for j= $f_{AC}(i) : \mathrm{p} : f_{AB}(i)$
8. Use multi-variate spline functions to do the assignment for the point in output image.
9. for i= $[x_D] : [x_C]$
10. for j= $f_{AC}(i) : \mathrm{p} : f_{BC}(i)$
11. Use multi-variate spline functions to do the assignment for the point in output image.

Chapter 4: AUTOMATIC IMAGE REGISTRATION BY USING MULTIVARIATE SPLINE METHOD

In this chapter, the Automatic image Registration by using Multivariate Spline Method will be introduced. As we already have an Edge of Object Recognition Method for Multivariate Spline Method and Multivariate Spline Method in hand. Actually, it won't difficult to build an automatic Image Registration Method. Two methods will be given in this chapter, one is called Pre-Selection Automatic Image Registration by using Multivariate Spline Method, write as PAMSM for short. The other is called Adding point Automatic Image Registration by using Multivariate Spline Method, write as AAMSM for short. Both of these two methods are using the edge searching method and Multivariate Spline Method.

4.1 Pre-Selection Automatic Image Registration by using Multivariate Spline Method

In this section, we will talk about the Pre-Selection Method for Automatic Image Registration by using Multivariate Spline Method which is builded by using the Edge of Object Recognition Method and Multivariate Spline Method.

According to the Multivariate Spline Method, firstly, we need find the feature points. According to the section 2.4. we already find the edge of objects for the given satellite image, such as Figure 2.2 and 2.1 as the input image, and the output image are Figure 2.19 and 2.20. As you know, the blue closed curves which are showed on the Figure 2.19 and 2.20 are the objects that we find by using the Ending Point Zero Method. The PAMSM is used the blue closed curves as the feature points for each given input image. Then finding the corresponding feature points for the image registration.

In the PAMSM, as feature point is a set of points, we build the feature point by this way: taking the centre point of each blue close curve as the first point and taking all of points on the blue close curve as the other points, so that we have a points' set as the feature point. Now, each input images has a set of feature points which depends on how many closed curves we can find on the given image.

To find the corresponding feature points, we need use the brute-force search or exhaustive search, in computer science, which is a very general problem-solving technique that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement. So, we pick up one feature point one feature point from one image, and try to find the corresponding feature points from the other image by calculating **the matching error**.

The matching error is calculated by using the L-2 Norm. According to the section 3.3 Best Affine Transformation, we can find the affine transformation for any pair of given feature points. For example, take $X_i = \{x_{1i}, x_{2i}, ..., x_{ni}\}$ as the feature point from one image where $\{x_{ki} | k = 1, 2, ..., n\}$ is a set of points on the one of blue closed curve of this image and its centre point. And $Y_j = \{y_{1j}, y_{2j}, ..., y_{mj}\}$ as the feature point from the other image where $\{y_{pi} | p = 1, 2, ..., m\}$ is a set of points on the one of blue closed curve of this image and its centre point. Then we can obtain the equation 4.1.

$$AX_i + b = Y_j \qquad i, j = 1, 2...n, \tag{4.1}$$

where $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ and $b = \begin{pmatrix} b_{11} \\ b_{21} \end{pmatrix}$.

According to the section 3.3 Best Affine Transformation, before we computed the affine transformation matrix A and column b, firstly, we need make the two given feature points

have equally number of points. Such as X_i include n points and Y_j include m points, n and m may not be equal.

For **adding points** method, in order to keep the position of center point, we are finding the difference between n and m, adding points uniformly to the feature point with fewer points or subtracting points uniformly from the feature points with more points. In fact, usually it is difficult to add or subtract points completely uniformly. So the center points may be changed after adding points.

Now, the number of selected pair of feature point from two given image is equal, say n. And we can obtain the affine transformation matrix A and column b by using the equation 4.1. **The matching error** is calculated by equation 4.2.

$$error = \frac{1}{n}\sum_{k=1}^{n}\sqrt{(Ax_{k1}+b-y_{k1})^2+(Ax_{k2}+b-y_{k2})^2} \tag{4.2}$$

where $\begin{pmatrix} x_{k1} \\ x_{k1} \end{pmatrix} \in X_i$ and $\begin{pmatrix} y_{k1} \\ y_{k1} \end{pmatrix} \in Y_j$, $AX_i + b = Y_j$.

Next, as center point should be the first point, the order of other points can be changed in the feature point, by changing the order of other points, we can keep the minimum error as the matching error of the given pair of feature points.

Assume $p < q$, $X = \{X_1, X_2, ..., X_p\}$ is the set of feature points which obtained from one image, and $Y = \{Y_1, Y_2, ..., Y_q\}$ is the set of feature points which obtained from other image. So we will obtain a p by q matrix of the matching error. Find the first P minimum matching error of pair of feature points without replacement as the Pre - selected corresponding pair of feature point. For each of Pre - selected corresponding pair of feature point X_i and Y_i, say selected from $X = \{X_1, X_2, ..., X_p\}$ and $Y = \{Y_1, Y_2, ..., Y_p\}$, there exist a affine transformation matrix A_i and a column b_i.

NOTE1: Actually, those corresponding pair of feature points may not belong to the overlap part of the two given images. But we assume that there are at least three corresponding pair of feature points which belong to the overlap part so that we can use the Multivariate Spline Method.

Next step is using the Multivariate Spline Method to find the first three corresponding pair of feature points. We need use the brute-force search method, such as picking up three corresponding pair of feature points by selecting three feature points from X and three feature points from Y, then calculate the error by using the Multivariate Spline Method.

The Error is calculated by using L-2 Norm. Firstly, using the three corresponding pair of feature points to do the image registration by using the Multivariate Spline Method, we can obtain the mapping 3.6 as below:

$$S : Source\ Image(I_2) \to Output\ Image(R), denote\ by\ S(x,y) = (x_o, y_o),$$

$$S(\triangle) = \begin{cases} S_1(x,y) & (x,y) \in D_1 \\ S_2(x,y) & (x,y) \in D_2 \\ \vdots & \\ S_n(x,y) & (x,y) \in D_N \end{cases}$$

$$S_i(x,y) = \begin{pmatrix} a_{11_i}(x,y) & a_{12_i}(x,y) & b_{1_i}(x,y) \\ a_{21_i}(x,y) & a_{22_i}(x,y) & b_{2_i}(x,y) \end{pmatrix}$$

Secondly, finding the overlap part on both of the two given images by using the mapping S, assume all the pixels set belong to this overlap is Γ. As each pixel has a value on the original given image, we pick up a pixel on the overlap part from one image, then using the mapping S to find the corresponding pixel value on the other image so that we can calculate

The Error.

$$\textbf{The Error} = \frac{1}{M} \sum_{(x,y)\in\Gamma} \sqrt{(I_2(S(x,y)) - R(x,y))^2} \tag{4.3}$$

Where M is the number of the pixel belong to Γ, I_2 and R are the pixel value mapping of given image.

The source of **The Error** related to several cases, such as the image taken with different photographic equipment, different shooting angle and so on. Those cases may give the same object different value in different picture, we called this error as Initial Error. The Initial Error can be calculate by the mean of the difference of the sampling points. We can use the pair of corresponding feature points which we can find as the sampling points. The error of our Method can be approximate by the difference between **The Error** and the Initial Error. In the chapter 5, **The Error** we showed is the total error, not only for our method error. In fact, the initial error should be a constant $\gamma > 0$ which we can't change by our method, so **The Error** may be not 0.

We pick up the minimum error of **The Error**, and the three corresponding pair of feature points which corresponding to this minimum error is what we need. Basic on those three corresponding pair of feature points, we can add more corresponding feature points by calculate **The Error**. We will build the new feature points by using those three corresponding pair of feature points. The designing scheme is as follow:

- Using the center of testing feature points as the center of new feature points.
- Using the corresponding center point of the nearest three corresponding pair of feature points as the other points of the new feature points respectively.

If we can't find **The Error** which is less than the previous step's error, it means that there is no more corresponding pair of feature points in the given feature points' set, so we can

stop the searching. **The Error** of Pre-Selection Automatic Image Registration by using Multivariate Spline Method is current error.

NOTE2: When we try to add the corresponding points, we will try to test the feature points which near the three corresponding pair of feature points first. The reason is as follow:

- guarantee the testing feature points belongs to the overlap part.
- obtain a reasonable feature points, as we will build the new feature points by using the three corresponding pair of feature points.

NOTE3: As we already have three corresponding feature points, we can find the affine transformation matrix A_i and a column b_i for the other feature point by using the mapping 3.6 which obtained by those three corresponding feature points. We call this matrix A_i and a column b_i as pre-matrix A_i and a pre-column b_i, then we have 3 adding point method as follow:

- Adding Point Method 1: For any given two feature points X_i and Y_j which are from two images, the pre-matrix A_i and a pre-column b_i of feature point X_i is given. So, we find other two sets X_i^1 and Y_j^1, see Equation 4.4. Then the new feature points can be build as $X_i = \{centerponit, X_i, Y_j^1\}$ and $Y_j = \{centerponit, X_i^1, Y_j\}$.

$$X_i^1 = (A_i)^{-1} * (Y_j - b_i) \qquad Y_i^1 = A_i * X_i + b_i \tag{4.4}$$

- Adding Point Method 2: For any given two feature points X_i and Y_j, by using the Equation 4.4, we can build it as $X_i = X_i$ and $Y_j = \{centerponit, Y_i^1\}$.
- Adding Point Method 3: For any given two feature points X_i and Y_j, by using the Equation 4.4, we can build it as $Y_i = Y_i$ and $X_j = \{centerponit, X_i^1\}$.

By using those 3 adding points method, we can obtain a more accuracy affine transformation matrix A_i and a column b_i for the other feature points that may not change the Mapping S too much, and finding the result image much faster. But, by using this method, the convergency result image may be theoretically not the actual result image. Because the affine transformation matrix A_i and a column b_i for the other feature points which we find is depend on the three corresponding feature points which we assumed before. In other world,

$$error \to \varepsilon \qquad as \quad n \to \infty \tag{4.5}$$

where $\varepsilon \neq 0$, n is the number of corresponding pair of feature points which we can find.

NOTE4: As we already have three corresponding feature points, assume the center point of those three corresponding feature points is $\{Xcenter3\}$ and $\{Ycenter3\}$, we can build the next feature points by using $\{Xcenter3\}$ and $\{Ycenter3\}$ directly to find the next corresponding feature point. As we can find the center points of the left feature points, we can build new feature points $X = \{center, Xcenter3\}$ and $Y = \{center, Ycenter3\}$. By calculating the **The matching error** or **The Error**, we can find the corresponding feature point. But this method is more depending on those three corresponding feature points than the **NOTE3** method, and the center of the new feature points couldn't change. So, this isn't a good method, we don't recommend this method.

So, according to the PAMSM, more corresponding pair of feature point we can find, less error of result image we can obtain. We can say that this method is reasonable, controllable and flexible.

4.2 Adding point Automatic Image Registration by using Multivariate Spline Method

In this section, we will talk about the Adding point Automatic Image Registration by using Multivariate Spline Method(AAMSM) which is also builded by using the Edge of Object Recognition Method and Multivariate Spline Method. The most important difference between AAMSM and PAMSM is that the PAMSM requires that there are at least three findable corresponding pair of feature points which belong to the overlap part of two given image, but the AAMSM don't need to satisfy this condition. Now, let's talk about AAMSM step by step.

According to the Multivariate Spline Method, firstly, we need find the feature points. According to the section 2.4. we already find the edge of objects for the given satellite image, such as Figure 2.2 and 2.1 as the input image, and the output image are Figure 2.19 and 2.20. As you know, the blue closed curves which are showed on the Figure 2.19 and 2.20 are the objects that we find by using the Ending Point Zero Method. The AAMSM need to use the blue closed curves as the feature points for each given input image too. Then finding the corresponding feature points for the image registration.

In the AAMSM, as feature point is a set of points, we build the feature point by this way: taking the centre point of each blue close curve as the first point and taking all of points on the blue close curve as the other points, so that we build a points' set as the feature point. In this way, we can find a set of feature points which depends on the number of closed curves for each given images. In the AAMSM, the feature points will always be build by this way. By the way, this is a different between AAMSM and PAMSM.

Similarly, in AAMSM, to find the corresponding feature points, we need use the brute-force search. So, we pick up each feature point from one image, and try to find a corresponding feature point from the other image. So, **The matching error** should be calculated.

In AAMSM, **The matching error** is also calculated by using the L-2 Norm. According to the section 3.3 Best Affine Transformation, we can find the affine transformation for any pair of given feature points. Like we already talked about in last section, similarly, we can obtain the equation 4.2. Before we computed the affine transformation matrix A and column b, firstly, we need make the two given feature points have equally number of points.

For **adding points** method, in the AAMSM, in order to keep the position of center point, we are finding the LCM(Least Common Multiple) between n and m, so that we can adding points complete uniformly for both two given feature points by adding the number of points to $LCM(n, m)$ which is usually greater than 3. This **adding points** method can keep the center point of feature points better after adding points and given us less error, but computation time is much more than the PAMSM's.

Now, the number of selected pair of feature point from two given image is equal, say $LCM(n, m)$. And we can obtain the affine transformation matrix A and column b by using the equation 4.1. In AAMSM, **The matching error** is also calculated by Equation 4.2.

Next, as center point should be the first point, the order of other points can be changed in the feature point, by changing the order of other points, we can keep the minimum error as the matching error of the given pair of feature points.

Assume $p < q$, $X = \{X_1, X_2, ..., X_p\}$ is the set of feature points which obtained from one image, and $Y = \{Y_1, Y_2, ..., Y_q\}$ is the set of feature points which obtained from other image. So we will obtain a p by q matrix of the matching error. Find the pair of feature points without replacement as the Pre - selected corresponding pair of feature point as many as we can. For each of Pre - selected corresponding pair of feature point X_i and Y_i, say selected from $X = \{X_1, X_2, ..., X_p\}$ and $Y = \{Y_1, Y_2, ..., Y_p\}$, there exist a affine transformation matrix A_i and a column b_i.

NOTE1: Actually, some of those feature points may not belong to the overlap part of the two given images. So the number of corresponding pair of feature point is no more than P.

Next step, by using the brute-force search method, we have $p * q$ pair of feature points. For each pair of feature points, we do the image registration Automatically by using the Multivariate Spline Method, as we already found the minimum **matching error** and affine transformation matrix A and column b for them. Then calculate **The Error** by using Equation 4.3.

The third step, taking the pair of feature point which has the minimum **Error** as a pair of corresponding feature point, then try to find the next pair of corresponding feature point. Before to do the for loop,

Case 1: Do the for Loop directly, to find the pair of corresponding feature points as many as we can, until **The Error** couldn't be reduced any more.

Case 2: Refine the affine transformation matrix A and column b for other feature points by using the the mapping 3.6 S. By using the 3 adding point method as follow:

- Adding Point Method 1: For any given two feature points X_i and Y_j which are from two images, the pre-matrix A_i and a pre-column b_i of feature point X_i is given. So, we find other two sets X_i^1 and Y_j^1, see Equation 4.4. Then the new feature points can be build as $X_i = \{centerponit, X_i, Y_j^1\}$ and $Y_j = \{centerponit, X_i^1, Y_j\}$.

$$X_i^1 = (A_i)^{-1} * (Y_j - b_i) \qquad Y_i^1 = A_i * X_i + b_i \tag{4.6}$$

- Adding Point Method 2: For any given two feature points X_i and Y_j, by using the Equation 4.4, we can build it as $X_i = X_i$ and $Y_j = \{centerponit, Y_i^1\}$.

- Adding Point Method 3: For any given two feature points X_i and Y_j, by using the Equation 4.4, we can build it as $Y_i = Y_i$ and $X_j = \{centerponit, X_i^1\}$.

The pre-matrix A_i and a pre-column b_i is calculated by those two case:

- The pre-matrix A_i and a pre-column b_i is calculated by the current image registration mapping S.
- The pre-matrix A_i and a pre-column b_i is calculated by the image registration mapping S which is obtained by using the nearest two or three corresponding feature points of this feature point.

NOTE2: The center point shouldn't change when we use those 2 adding points method. The number of points of both feature points should equal the $LCM(n, m) + 1$.

The affine transformation matrix A and column b for other feature points will be calculated by using the new feature point.

NOTE3: In case 1, abstractly, we can obtain a good result image which **The Error** will go to zero as more and more pair of corresponding feature points can be found. But, as a matter of fact, the method in chapter 2 is not good enough to find infinitely many pair of corresponding feature points, so, most of times, there is not enough corresponding feature points to make **The Error** go to zero. So, if there is only less than 10 pair of feature points, case 1 may be stop the For Loop faster than case 2, because case 1 may not find corresponding feature points as much as case2. In this case, case2 will give us a less error result image. Otherwise, in theory, there should have infinitely many pair of corresponding feature points on the overlap part of two images, **The Error** of case 1 will go to zero, not case 2.

NOTE4: In case 2, because the affine transformation matrix A_i and a column b_i for the other feature points which we find is depend on the three corresponding feature points which we assumed before. In other world,

$$error \to \varepsilon \qquad as \quad n \to \infty \tag{4.7}$$

where $\varepsilon \neq 0$, n is the number of corresponding pair of feature points which we can find.

So, according to this method, first, we should assume that there is at least one pair of corresponding point which can be found by Chapter 2's Method. Second, similarly as PAMSM, in this method, more pair of corresponding feature point can be found, less error of output image we can obtain. This method is also reasonable, controllable and flexible, after improved the boundary selection method, the AAMSM case 1 will works better.

Chapter 5: MAIN RESULTS AND EXPERIMENTS

In this chapter, we will show some results by using breast MRI and do some comparison.

5.1 Results by using semi-automatic image registration by using Multivariate Spline Function with MRI

In this section, we will show our results by using breast MRI.

First experiment:

Here are two input images:

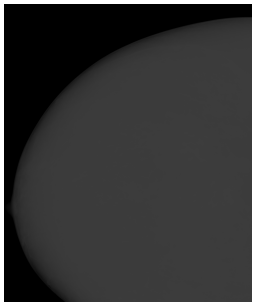

Figure 5.1: MRI168

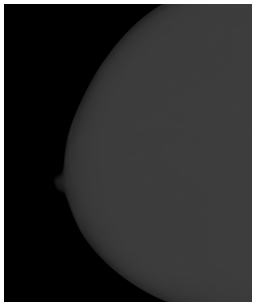

Figure 5.2: MRI911

The first step is to find the feature points. Here are the four corresponding feature points that we used:

YSETS1=[137.75 138.88 138.5 139.25 139.25; 13.17 14.3 12.42 15.05 11.67];

XSETS1=[119.68 120.81 120.44 121.57 121.19; 49.3 50.06 48.55 50.81 48.55];

YSETS2=[185.45 185.08 184.23 188.75 188.56 189.78; 18.16 18.63 17.41 19.38 20.23 21.08];

XSETS2=[166.07 165.04 164.38 169.55 168.89 169.74; 57.49 57.87 56.64 59.37 59.56 60.12];

YSETS3=[110.93 109.24 106.98 109.52 107.83 112.63 114.6; 21.73 21.45 20.89 24.84 25.97 21.45 20.04];

XSETS3=[91.17 89.48 86.09 90.04 86.66 94 96.25; 57.02 56.45 56.45 58.99 60.69 55.61 54.76];

YSETS4=[202.2 202.11 201.45 202.67 201.45; 31.1 30.77 31.24 31.61 29.73];

XSETS4=[180.94 180.75 180.28 181.41 180.09; 70 69.72 70.38 70.76 68.5];

Here are the four corresponding affine transformation results:

$$A_1 = \begin{pmatrix} 1.0868 & 0.0837 \\ 0.0873 & 0.7170 \end{pmatrix} \qquad A_2 = \begin{pmatrix} 1.5109 & -0.9603 \\ 0.2556 & 0.6112 \end{pmatrix}$$

$$A_3 = \begin{pmatrix} 1.3341 & -0.0588 \\ -0.1210 & 0.8487 \end{pmatrix} \qquad A_4 = \begin{pmatrix} 0.9030 & 0.1273 \\ -0.2431 & 1.2962 \end{pmatrix}$$

$$b_1 = \begin{pmatrix} -54.991 \\ 49.452 \end{pmatrix} \qquad b_2 = \begin{pmatrix} -170.38 \\ -1.345 \end{pmatrix} \qquad b_3 = \begin{pmatrix} -97.736 \\ 91.790 \end{pmatrix} \qquad b_4 = \begin{pmatrix} -10.097 \\ 139.257 \end{pmatrix}$$

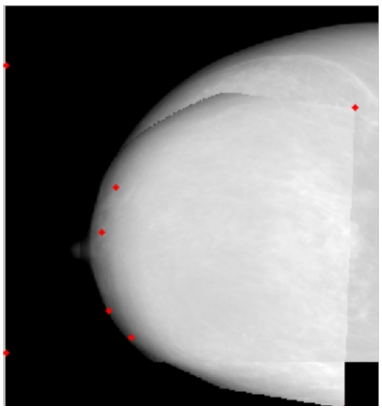

Figure 5.3: 4 Points Output Image

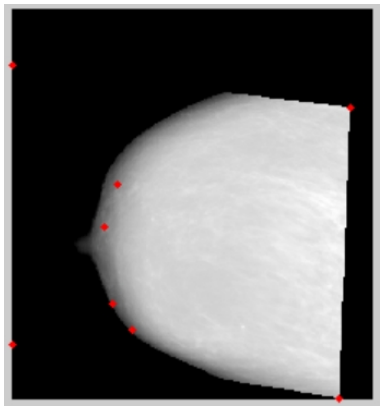

Figure 5.4: 4 Points Source Image

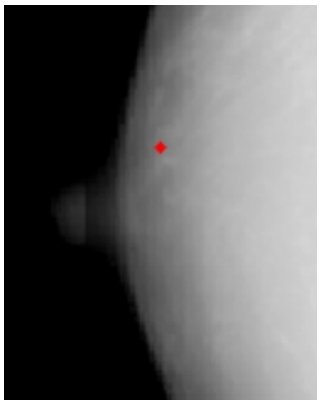

Figure 5.5: 4 Points Nipple

Figure 5.6: 4 Points Inside Image

Figure 5.7: 4 Points Bad Part

From the output image(Figure 5.3-5.6), the nipple matched very well, and we also get a good match for inside of the breast by checking the texture of output image. in order to see the picture clearly, I did the enlargement for some feature parts(Figure 5.5-5.6).

At bottom of the image, the boundary do not match very well, see Figure 6.7. This is because the interpolation point is not enough and can not control it very well. The red point on the picture are interpolation points we used, those points are not far from the nipple, so the nipple matched very well.

Second experiment:

As we know, in the first experiment, it does not match very well on a small part of the

boundary. This time, we use six feature points to do the image registration, and we find two more corresponding feature points on that part.

Here are the feature points:

YSETS1=[137.75 138.88 138.5 139.25 139.25; 13.17 14.3 12.42 15.05 11.67];

XSETS1=[119.68 120.81 120.44 121.57 121.19; 49.3 50.06 48.55 50.81 48.55];

YSETS2=[185.45 185.08 184.23 188.75 188.56 189.78; 18.16 18.63 17.41 19.38 20.23 21.08];

XSETS2=[166.07 165.04 164.38 169.55 168.89 169.74; 57.49 57.87 56.64 59.37 59.56 60.12];

YSETS3=[110.93 109.24 106.98 109.52 107.83 112.63 114.6; 21.73 21.45 20.89 24.84 25.97 21.45 20.04];

XSETS3=[91.17 89.48 86.09 90.04 86.66 94 96.25; 57.02 56.45 56.45 58.99 60.69 55.61 54.76];

YSETS4=[202.2 202.11 201.45 202.67 201.45; 31.1 30.77 31.24 31.61 29.73];

XSETS4=[180.94 180.75 180.28 181.41 180.09; 70 69.72 70.38 70.76 68.5];

YSETS5=[202.2 202.58 202.86 202.01 201.82; 103.69 103.97 104.35 104.06 104.53];

XSETS5=[178.49 178.77 179.05 178.21 178.02; 139.25 139.54 139.91 139.63 140.01];

YSETS6=[209.35 208.51 211.33 208.03 205.59 210.76; 127.21 125.61 122.04 129.47 131.73 124.01];

XSETS6=[189.4 188.28 191.38 187.81 184.89 190.72; 160.61 159.3 156.94 163.34 165.51 157.7];

The last two corresponding feature points are the new input data. Here are the corresponding affine transformation results:

$$A_1 = \begin{pmatrix} 1.0868 & 0.0837 \\ 0.0873 & 0.7170 \end{pmatrix} \quad A_2 = \begin{pmatrix} 1.5109 & -0.9603 \\ 0.2556 & 0.6112 \end{pmatrix} \quad A_3 = \begin{pmatrix} 1.3341 & -0.0588 \\ -0.1210 & 0.8487 \end{pmatrix}$$

$$A_4 = \begin{pmatrix} 0.9030 & 0.1273 \\ -0.2431 & 1.2962 \end{pmatrix} \quad A_5 = \begin{pmatrix} 0.9665 & -0.1457 \\ 0.0495 & 0.9502 \end{pmatrix} \quad A_6 = \begin{pmatrix} 1.2876 & 0.0937 \\ -0.2748 & 0.78797 \end{pmatrix}$$

$$b_1 = \begin{pmatrix} -54.991 \\ 49.452 \end{pmatrix} \quad b_2 = \begin{pmatrix} -170.38 \\ -1.345 \end{pmatrix} \quad b_3 = \begin{pmatrix} -97.736 \\ 91.790 \end{pmatrix} \quad b_4 = \begin{pmatrix} -10.097 \\ 139.257 \end{pmatrix}$$

$$b_5 = \begin{pmatrix} -3.3022 \\ 54.4878 \end{pmatrix} \quad b_6 = \begin{pmatrix} -162.611 \\ 208.445 \end{pmatrix}$$

From the affine transformation matrix we obtained, we find that the first four affine transformation does not change, this means that we add two affine transformation, this is because we used the Delaunay triangulation to do the division, when we add new point, the division is only change at local area(Figure 3.9). That is also the reason that we must use Delaunay triangulation to do the division for the multivariate spline function.

In this way, the adding points is only control the interpolation function locally, so the output image will only change locally.

From our result(Figure 5.8-5.9), we can find that the two input images matched very well on the boundary, as we add feature points around the bottom of the input image, and compare with the Figure 5.3, they look the same on the upper half of the image, the nipple matched as same as each other, by the way, there is some red nodes on the output image, all of them are interpolation points. This is helpful for us to determine the best local area for us to add

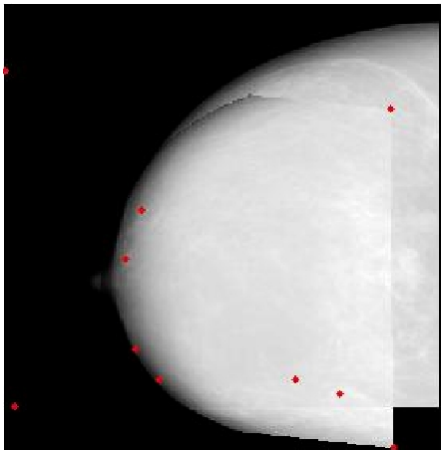

Figure 5.8: 6 Point Output Image

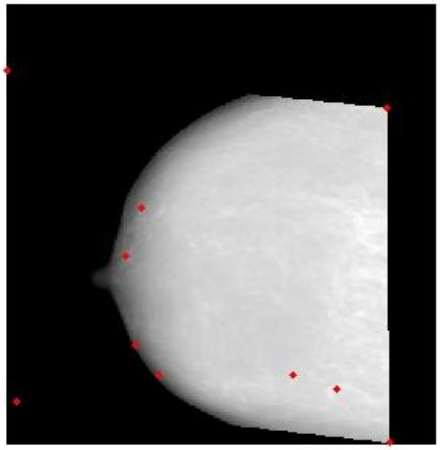

Figure 5.9: 6 Point Deformed Output

the interpolation point if we need.

Here is the corresponding part of the 4 feature points and the 6 feature points of the image.

Figure 5.10: Bad for 4 Feature Point

Figure 5.11: Good for 6 Feature Point

More over, those two Figure 5.12 and 5.13 are very useful for us to check about the Delaunay triangulation and our algorithm. In those two pictures, the red point are the intersection points that obtained by our algorithm, and I display them on this two Figure. It is clearly that all the red the point on the intersection points in Figure 5.8, this mean our intersection points are correct. In Figure 5.9, all of the new intersection points has red point on it, that means our program and algorithm are correct for the refinement of each triangle.

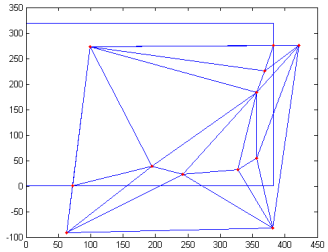

Figure 5.12: 6 Point Delaunay Triangulation

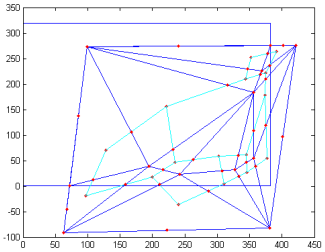

Figure 5.13: 6 Point Subdivision Point

5.2 Results by using automatic image registration by using Multivariate Spline Function with satellite images

First, we use two pair of satellite images as input image, the pair of satellite images is Figure 5.14 and 5.15, the other pair of satellite image is Figure 5.16 and 5.17.

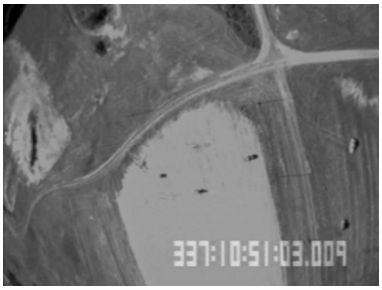

Figure 5.14: firstpair1

Figure 5.15: firstpair2

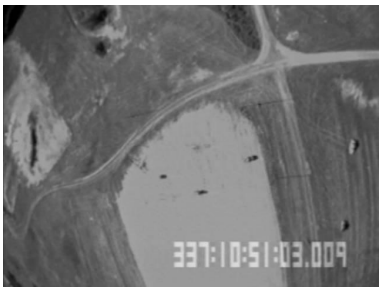

Figure 5.16: secondpair1

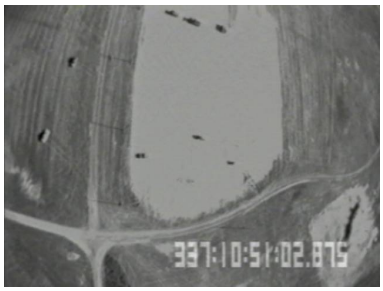

Figure 5.17: secondpair2

By using the boundary finding Method in section 2.2, we obtain the boundary image see Figure 5.18, 5.19, 5.20 and 5.21. In the section 2.2.2, we talked about the Equation 2.17, the threshold ε is dependence on n.

For the first pair of satellite images, we set $n = 1.25$. For For the second pair of satellite images, we set $n = 1.6$. As the first image in those two pair of satellite images is same. Clearly, the greater n return a image with more boundary point or more details.

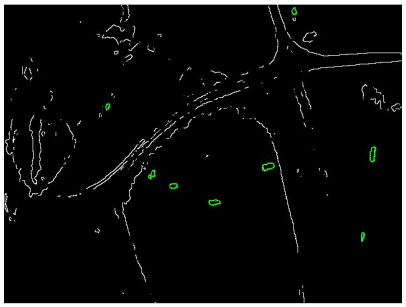

Figure 5.18: boundary of first pair1

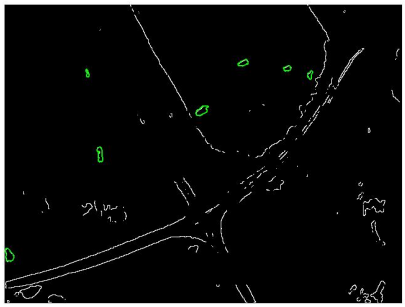

Figure 5.19: boundary of first pair2

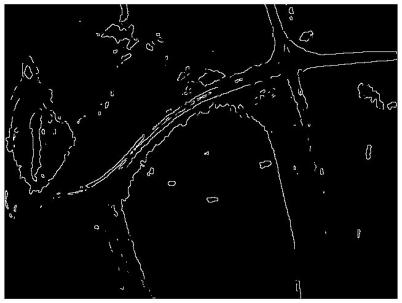

Figure 5.20: boundary of second pair1

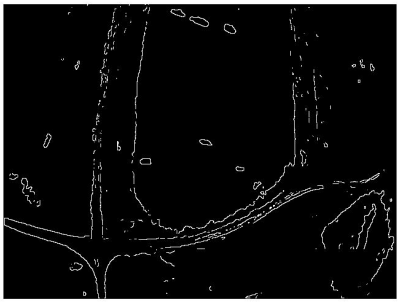

Figure 5.21: boundary of second pair2

By using the Edge of object Recognition Method in section 2.4, we obtain the edge images see Figure 5.22, 5.23, 5.24 and 5.25. In the section 2.4, we talked about the Ending Point Zero Method, the size of direction matrix is 3 by 3 and the number of points should greater than 20 for each objects. If less than 20, we treat it as noisy circle.

The edge of object recognition Method will select some of bad object which is not an object, just like the Figure 5.24. Not good for object recognition. But we will take care about those bad object later by using the method what we talked in Chapter 5. In fact, those

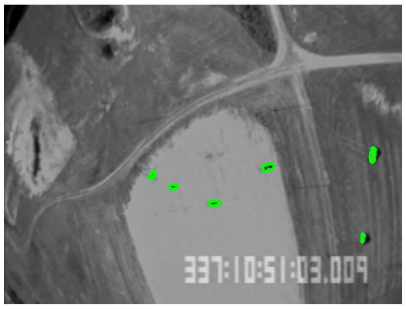

Figure 5.22: recognition of first pair1

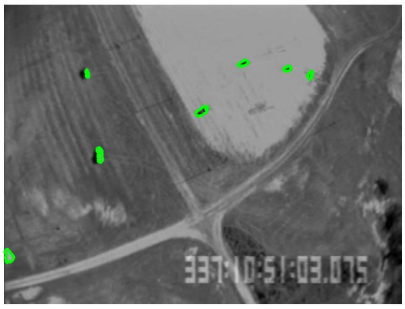

Figure 5.23: recognition of first pair2

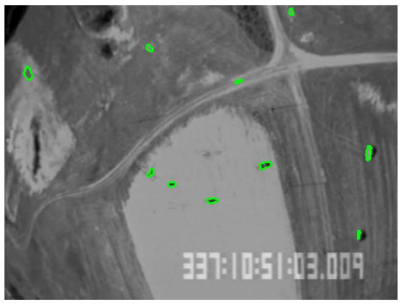

Figure 5.24: recognition of second pair1

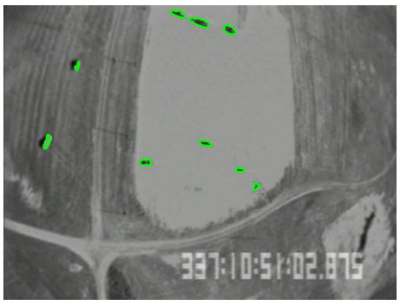

Figure 5.25: recognition of second pair2

bad objects will be removed by our method as our method keep going. By the way, my experiment computer's CPU is Intel(R) Core(TM) i5-2400S @2.50GHZ, RAM is 8GB. The software is MATLAB R2015b.

- For first pair of satellite images, Elapsed time is 13.656386 seconds for completing the previous steps.
- For first pair of satellite images, Elapsed time is 19.056367 seconds for completing the previous steps.

After we obtain the edge of object, we use the center of the edge of objects as the feature points, use the boundary points which on the edge of objects to calculate the affine transformation matrix A and column b.

For the first pair of satellite images, firstly, we use Pre-selection automatic image registration by using multivariate spline method to do the image registration (PAMSM).

The result images with 3 pair of corresponding feature points. As we mentioned before, the assumption for this method is that there are at least 3 pair of corresponding feature points which we can find. Actually, by checking the Figure 5.22 and 5.23, there should have at least 3 pair of corresponding feature points. And we find the 3 pair of corresponding feature points by our method, further more, we obtain the result image.

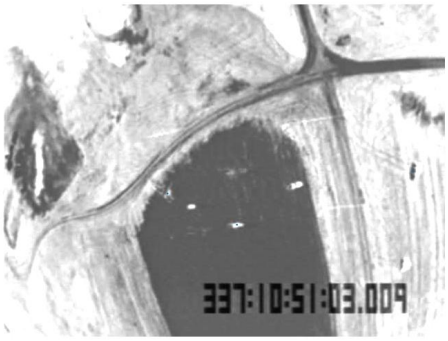

Figure 5.26: the 3 point of first pair1

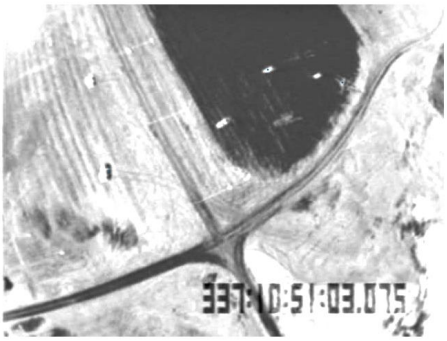

Figure 5.27: the 3 point of first pair2

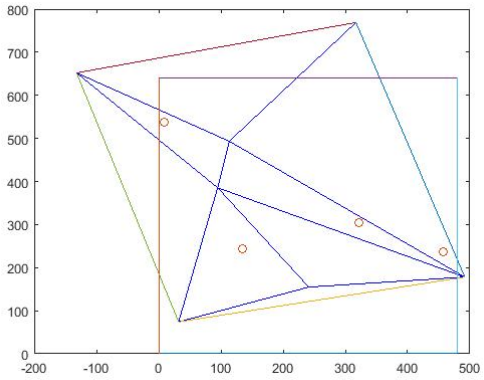

Figure 5.28: Delaunay Triangulation of first pair

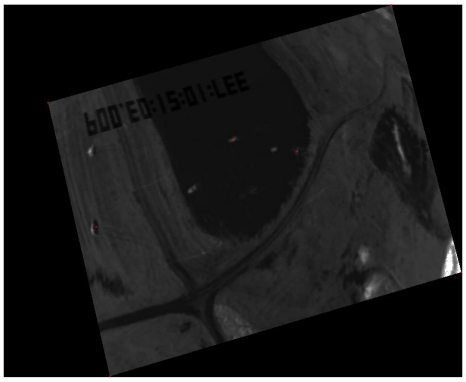

Figure 5.29: transferred image of first pair2

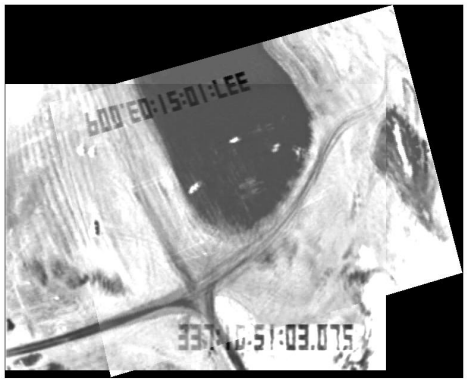

Figure 5.30: 3 point Result image1 of first pair

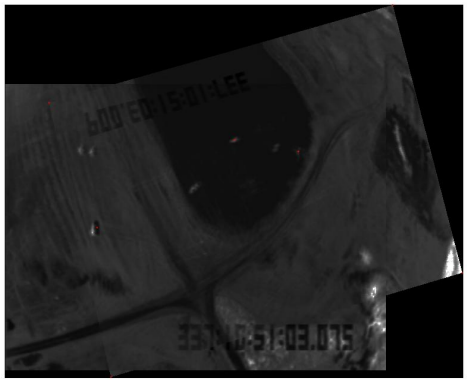

Figure 5.31: 3 point Result image2 of first pair

The error for this output image is round to 12. By the way, the error is calculate by the Equation 4.3 which is the average difference of each points gray value between two input image for their overlap part. The range of gray value for all pixels is 0 - 255. The Figure 5.26 and 5.27 is the input image with the 3 pair of corresponding feature points we used. Figure 5.28 is the Delaunay Triangulation by this 3 pair of corresponding points. The red circles shows the position of the left feature points which we didn't use yet. The Figure 5.29, 5.30 and 5.31 are the result images, a part of Up-Left of the result image doesn't match very well, the non-overlap parts which is on the Up-Middle of the result image looks like not in a correct position. Those give the registration result image an error.

Actually, there are several output image, we choose the minimum error output image as the result image. Let's look at a group of output image with a big error.

Look at the Figure 5.32 and 5.33, the red circle shows the wrong pair of corresponding feature points we used for image registration. The Figure 5.34 show the Delaunay Triangulation by using those 3 pair of feature points. There are 3 feature points which isn't located in the overlap part in Figure 5.34. This is one contradiction with Figure 5.22 and 5.23 which shows only one object doesn't belong to the overlap part. Figure 5.35, 5.36 and 5.37 are the output

Figure 5.32: the 3 point of first pair1 b

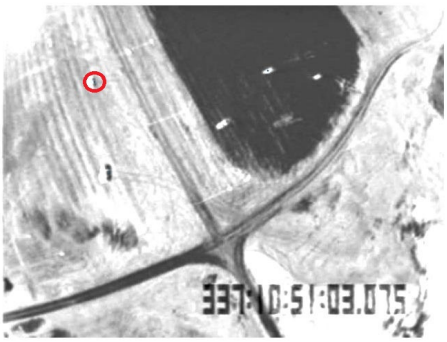

Figure 5.33: the 3 point of first pair2 b

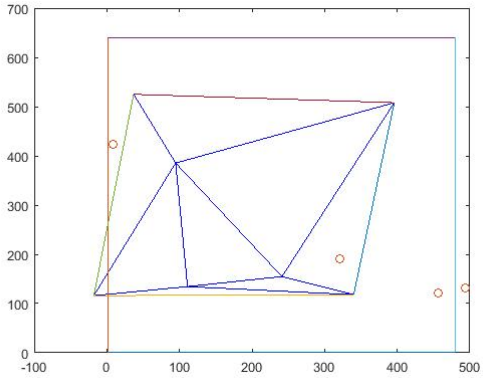

Figure 5.34: 3 point Delaunay Triangulation of first pair b

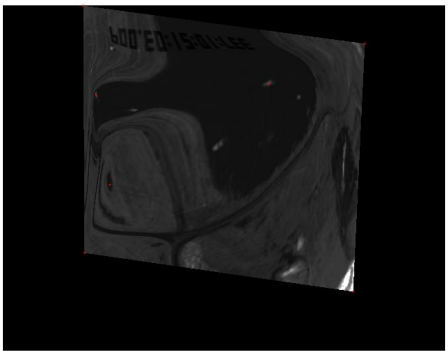

Figure 5.35: transferred image of first pair2 b

images which is really bad solution. This is a bad output image, I am not sure the error it is because this are many output image, but I am sure that this is not the minimum error case. As the minimum error case has already taken.

Next is the result images for Adding Point Method 2 with 4 corresponding feature points. Firstly, we need using the adding point method which we talk about in section 4.1. Here I will use the Adding point Method 2 as example to show the result image. This is because the Adding point Method 2 is better than other two method (see section 5.3). When we add the point for the object, the position of the center point may change. Now we will show three groups of the re-center experiment results.

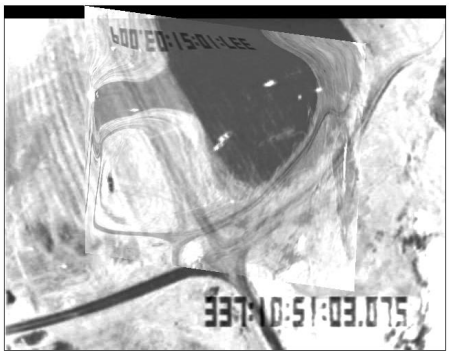

Figure 5.36: Result image1 of first pair b

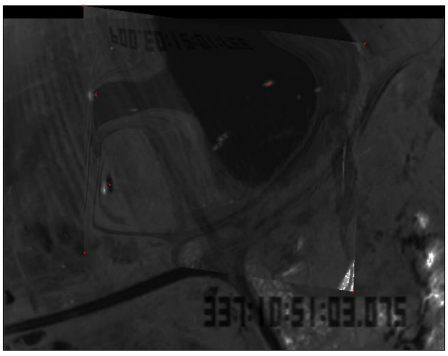

Figure 5.37: Result image2 of first pair b

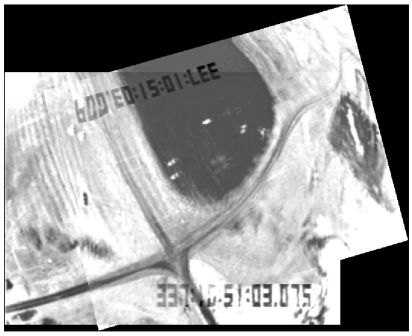

Figure 5.38: 4 point re-center 11

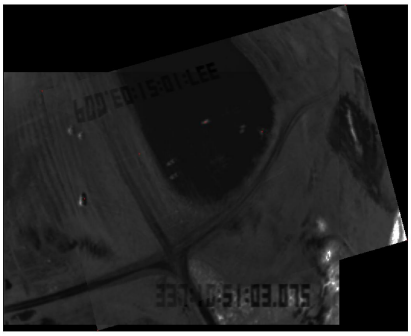

Figure 5.39: 4 point re-center 12

Look at Figure from 5.38 to 5.43. Each two figure corresponding to a different pair of corresponding feature points which we add into the 3 pair of corresponding feature points result image(see Figure 5.30 and 5.31), as we have other 3 feature points left for us to use(see Figure 5.28).

All of those output images are matched, but one part of result images don't look as good as 3 pair of corresponding feature points result image which located at the dark area. In fact, the error of those 3 group's result image is round to 4 which is less than the error of 3 pair of corresponding feature points result image. The reason is that all other part is matched better than before as one more feature points added into our method. but the re-center give

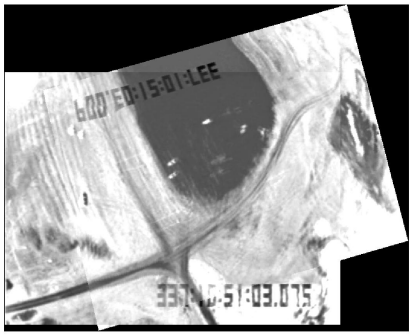

Figure 5.40: 4 point re-center 21

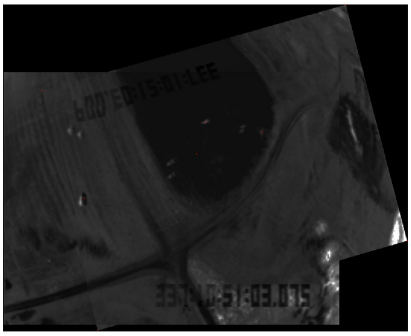

Figure 5.41: 4 point re-center 22

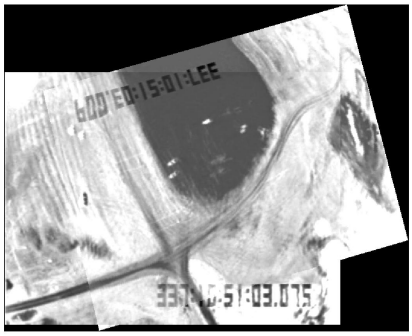

Figure 5.42: 4 point re-center 31

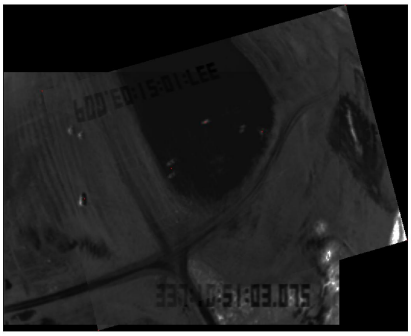

Figure 5.43: 4 point re-center 32

us a very clearly problem, that is one of the object doesn't matched. Because we find the position of object correctly which is clearly showed by Figure 5.22 to 5.25. the re-center method change the position of objects, of cause, our image registration method will match the re-center object but not the original object which's also clearly showed by Figure from 5.38 to 5.43.

Now, Let's check about the result images by Adding Point Method 2 with 4 corresponding feature points without re-center. We are wondering that the re-center results will be removed by the minimum error selection.

Look at Figure from 5.44 to 5.49, this is the result images after we adding a pair of corresponding feature points for the 3 pair of corresponding feature points result image. In Figure

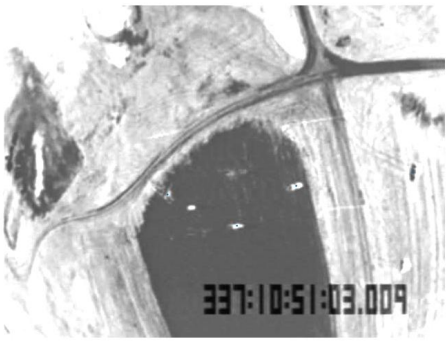

Figure 5.44: the 4 point of first pair 1

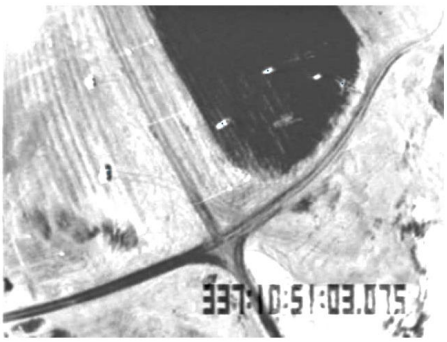

Figure 5.45: the 4 point of first pair 2

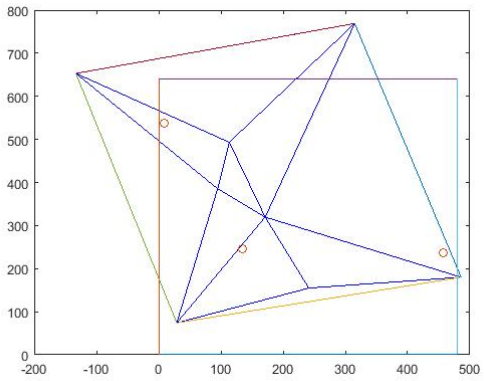

Figure 5.46: 4 point Delaunay Triangulation of first pair

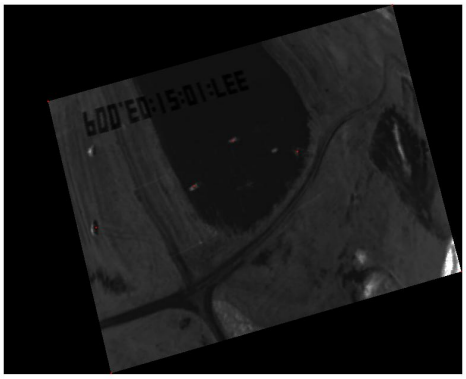

Figure 5.47: transferred image of first pair2

5.46, the number of left points is 3, but not 4. So a pair of corresponding feature points is found and added into the previous 3 pair of corresponding feature points result image. Figure 5.47 showed that the position of this new point locate inside of the dark area at the overlap part. By checking the Figure 5.48 and 5.49, both the dark area and Up-left part are matched better than Figure 5.30. And the result images looks really better than before. In fact, the error of this result image is round to 3 less than all previous result. So all of the re-center method output results will be removed by the the minimum error selection. This proved that the error which calculated by our method is reasonable and show our method works well.

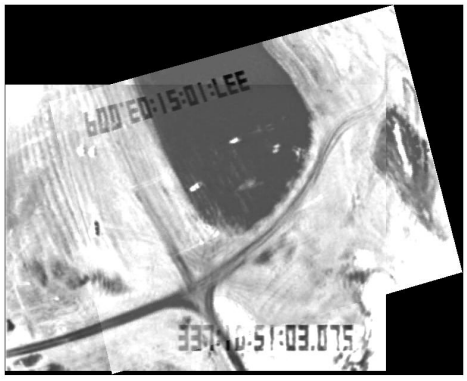

Figure 5.48: 4 point Result image1 of first pair

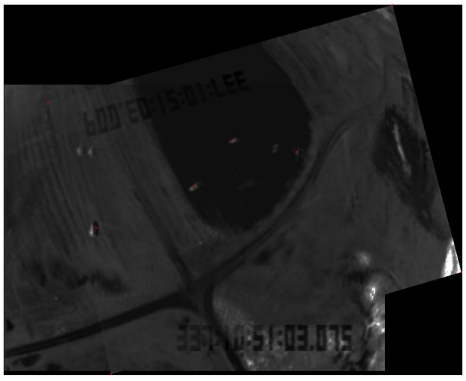

Figure 5.49: 4 point Result image2 of first pair

Now, let me show you our 5 pair of corresponding feature points result images by using PAMSM. As we had already showed the re-center output images for 4 pair of corresponding feature points result images by using PAMSM and its doesn't work well, so we'll only show the result images which is without re-center. See Figure from 5.50 to 5.55.

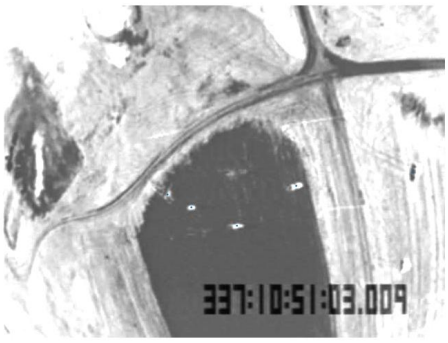

Figure 5.50: the 5 point of first pair 1

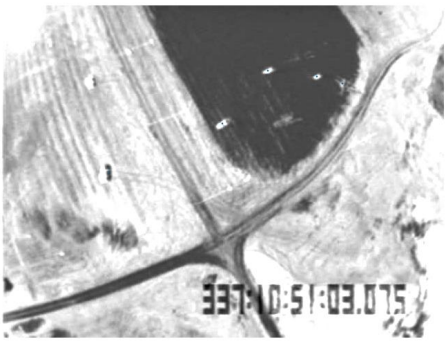

Figure 5.51: the 5 point of first pair 2

As a feature point adding into the method and locating at the dark area of overlap part see Figure 5.53, it is clearly that the dark part is matched very well, see Figure 5.54 and 5.55. And I should say that the result image looks really good. But the error is still round to 3. So, error doesn't change too much. The reason is that the affine transformation matrix

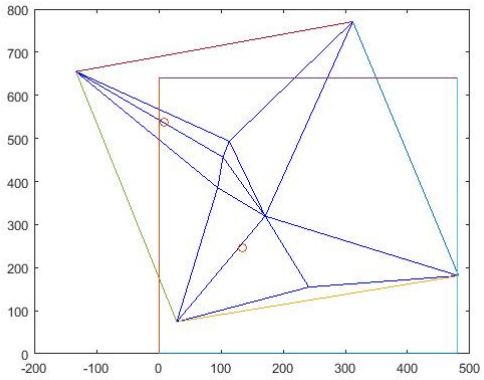

Figure 5.52: 5 point Delaunay Triangulation of first pair

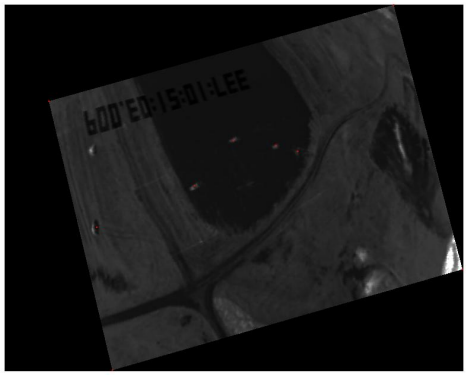

Figure 5.53: transferred image of first pair2

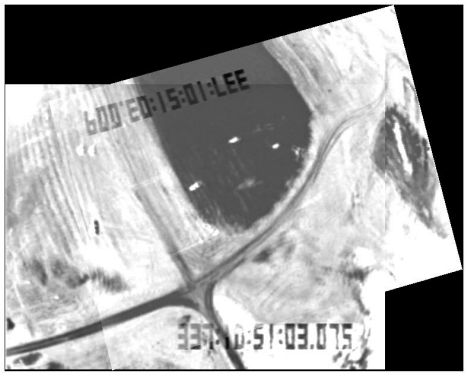

Figure 5.54: 5 point Result image1 of first pair

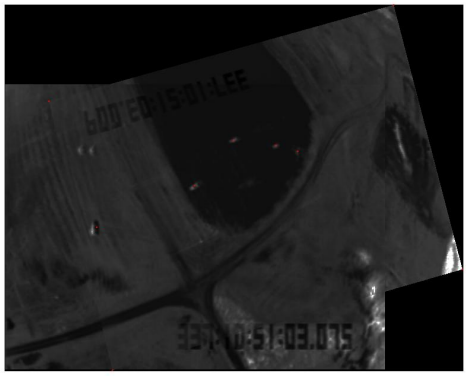

Figure 5.55: 5 point Result image2 of first pair

A and column b of the new feature points we used are depends on the previous 3 pair of corresponding feature points result image, so the error for this method will not goes to 0, but go to a small number ε, even we add more feature points into this method. May be the ε approximate to 3 for those two input images by using PAMSM. We can check by adding one more feature points, as we have other 1 point left there.

Finally, our method stop here, because the method couldn't find one more matching points. The error for all the output images by 6 pair of corresponding feature points result images is greater than 3. So, our method stopped. Actually, there is one more feature points left,

the reason for unable to find this pair of feature point is that the genuine transformation matrix A and column b for this pair of feature points is calculated incorrectly by the Adding Points Method. I guess the reason for Adding Points Method loss efficacy is that pair of feature points is too far away form the previous feature points. This is also one reason that the error for PAMSM may not go to 0.

As I want to check about what happened to 6 pair of corresponding feature points result images, I change the program to registration with the last pair of corresponding feature points, so that I can obtain the error and the result image. The result image is really bad, and the error is 4.0130 which is greater than the 5 pair of corresponding feature points result images. This is the reason that our method stop working at 5 pair of corresponding feature points. Look at the Figure from 5.56 to 5.63, those are the output images for 6 pair of corresponding feature points. It is clearly that the 6th feature points don't match well, and give the problem to the result image. This make the 6 pair of corresponding feature points result images worse than the 5 pair of corresponding feature points result images.

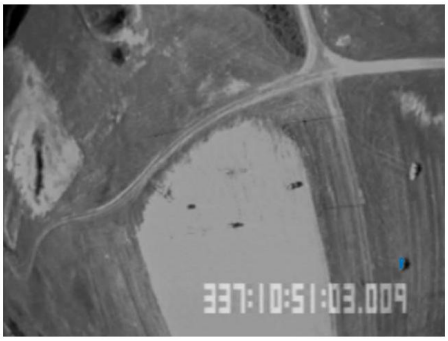

Figure 5.56: 6 point for adding points

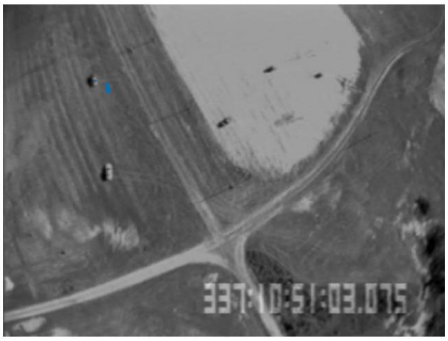

Figure 5.57: 6 point for adding points

Now, we talked about the results by Adding point Automatic Image registration by using Multivariate Spline Method (AAMSM). We still take the first group images as example.

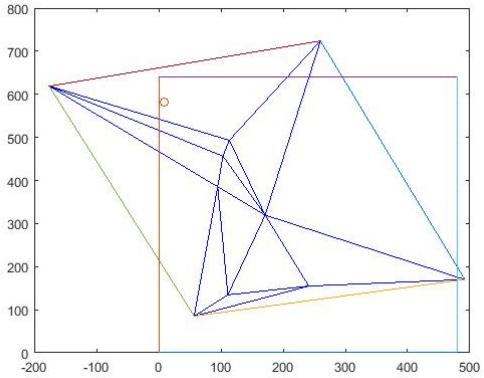

Figure 5.58: 6 point of first pair 1

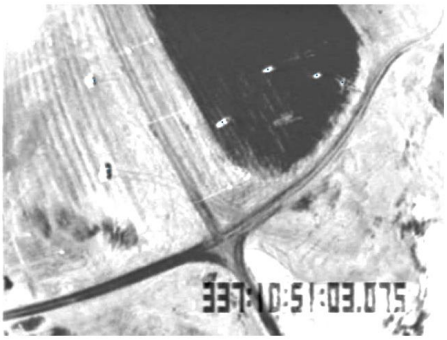

Figure 5.59: 6 point of first pair 2

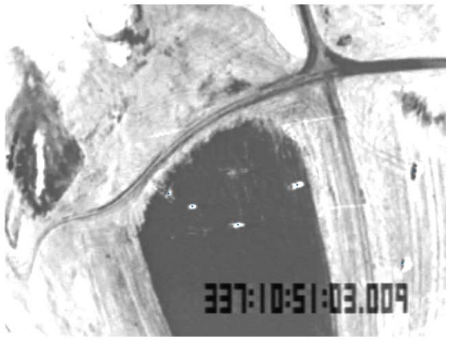

Figure 5.60: 6 point Delaunay Triangulation

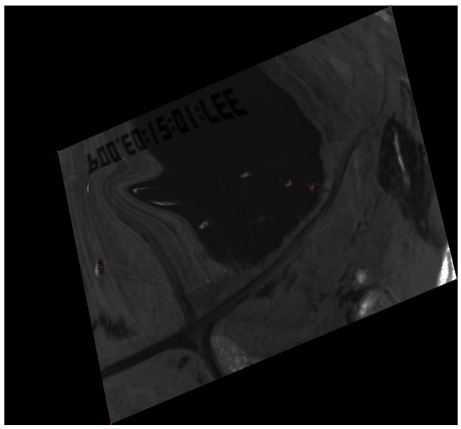

Figure 5.61: transferred image of first pair2

First, look at some output image for one pair of corresponding feature points which do not exist in PAMSM.

The Figure from 5.64 to 5.72 shows the result image by using only one pair of feature points in the first group satellite images. Take 3 figures as a group, each group of result image is obtained by different pair of feature points. The selected feature point is showed in the Figure 5.64, 5.67 and 5.70 which is the first image in the group. The last two images are the result images which matched by the selected feature point. It is clearly that all the result

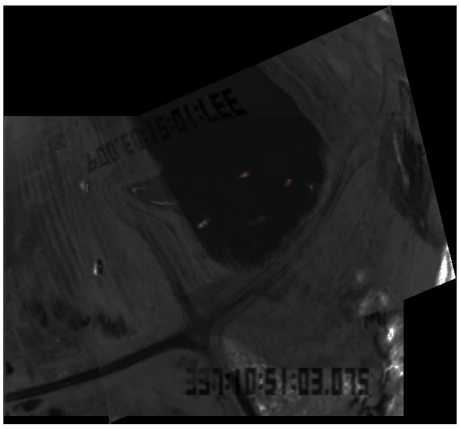

Figure 5.62: 6 point Result image1 of first pair

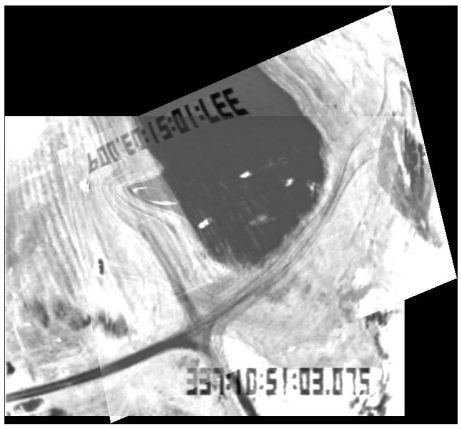

Figure 5.63: 6 point Result image2 of first pair

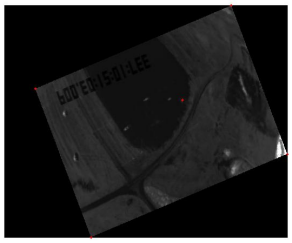

Figure 5.64: transferred image of first pair2

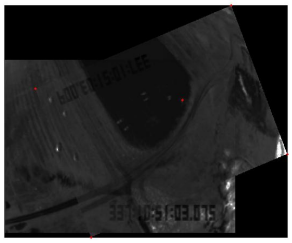

Figure 5.65: 1 point Result image1 of first pair

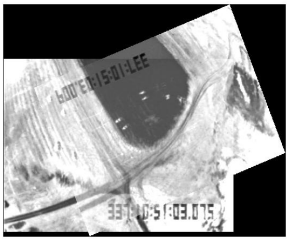

Figure 5.66: 1 point Result image2 of first pair

images from Figure 5.64 to 5.72 are matched but not a good registration. On other hand, of cause, there are some other output images which isn't matched, take some for example, see Figure form 5.73 to 5.78. Those wrong matching output image will be removed by the minimum error selection. In fact, all the one points output images isn't good enough to be a final result image.

The error for Figure from 5.64 to 5.72 are 15.1492, 15.728 and 16.6131. As the error are lager than the PAMSM's final result, even larger than 3 pair feature points result by PAMSM, I will not talk too much about it. The reason is that one pair of feature point is not enough for our Method. So, similarly, in order to obtain a good output images, we will add more

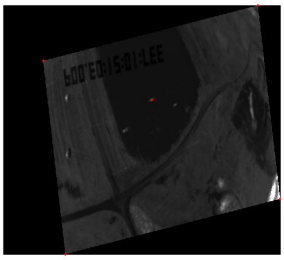

Figure 5.67: transferred image of first pair2

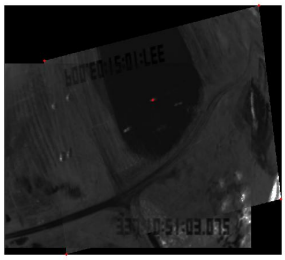

Figure 5.68: 1 point Result image1 of first pair

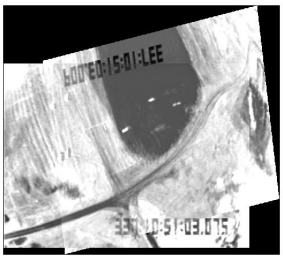

Figure 5.69: 1 point Result image2 of first pair

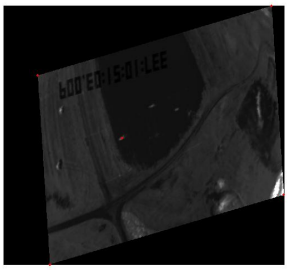

Figure 5.70: transferred image of first pair2

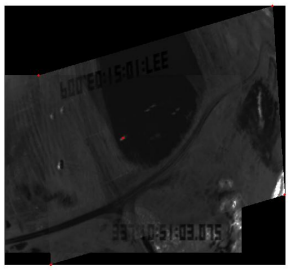

Figure 5.71: 1 point Result image1 of first pair

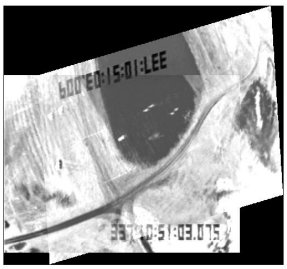

Figure 5.72: 1 point Result image2 of first pair

feature points to get a better output image.

We will show the 3 pair of corresponding feature points result image by AAMSM directly for comparison, ignore the 2 pairs of corresponding feature points result images. As there are $6 * 5 * 4$ choice to select 3 pair of corresponding feature points out the given 6 pair of corresponding feature points. I will show some output images and the result images.

Look at Figure from 5.79 to 5.96, those figure show the 3 pair of corresponding feature points result image by AAMSM, I selected 3 sets of result images from $6*5*4$ sets of result images, all those 3 sets of result image include the pair of corresponding feature points which can't find by PAMSM. By using the AAMSM, look at the red rectangle in Figure from 5.79 to 5.96, those images show the 6th feature point can be matched by AAMSM. The error of those 3 sets of result images is round to 4.2, 4.3 and 4.3 which is greater than the PAMSM.

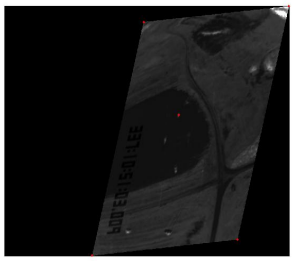

Figure 5.73: transferred image of first pair2

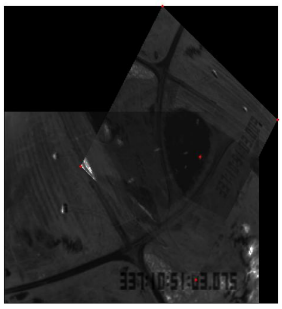

Figure 5.74: 1 point Result image1 of first pair

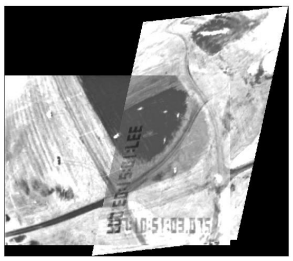

Figure 5.75: 1 point Result image2 of first pair

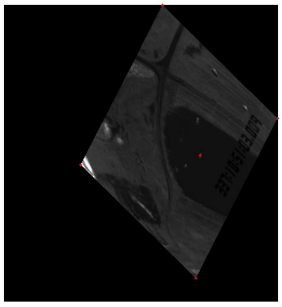

Figure 5.76: transferred image of first pair2

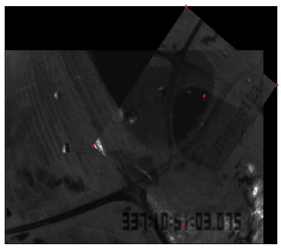

Figure 5.77: 1 point Result image1 of first pair

Figure 5.78: 1 point Result image2 of first pair

But those results are good to show that AAMSM can match for all kinds of feature points, this is the reason that the error of AAMSM will go to 0 as we can find enough pair of feature points.

Look at the Figure from 5.97 to 5.102, we selected the set of result images which have same 3 pair of corresponding feature points with the PAMSM's. The result image by AAMSM look a little better than PAMSM's. In fact, the error of those result image by AAMSM is round to 3.9 which is also a little better than PAMSM's. I am sure the reason is because the Adding points Method by LCM is worked better for our method if there have the 3 pair of corresponding feature points which we can find by our method. As we only find 6 objects in the given first pair satellite images, this case may not good for finding the result image

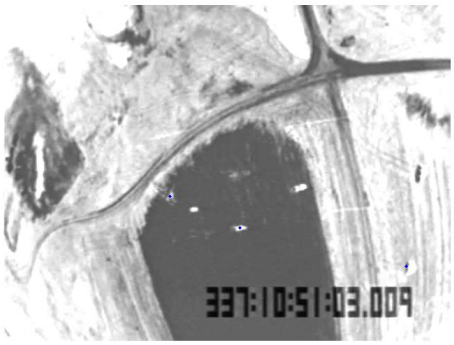

Figure 5.79: 3 point of first pair 1 by AAMSM

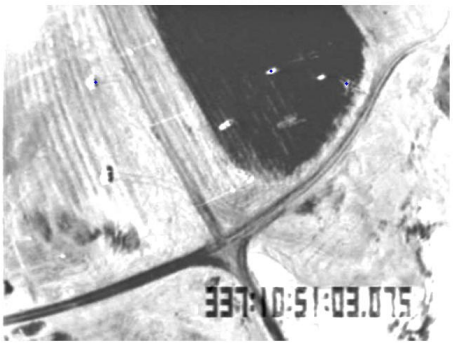

Figure 5.80: 3 point of first pair 2 by AAMSM

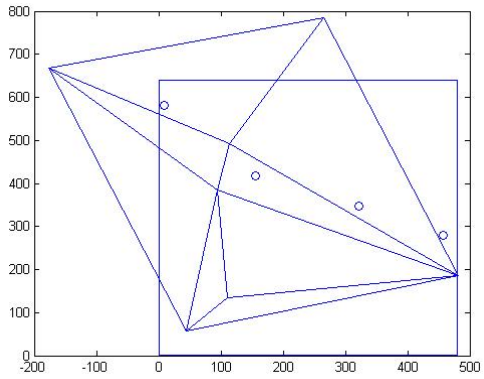

Figure 5.81: 3 point Delaunay Triangulation by AAMSM

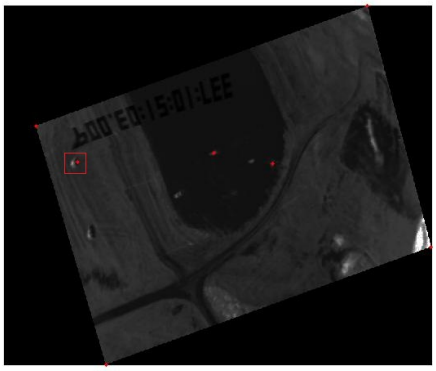

Figure 5.82: transferred image of first pair2 by AAMSM

by AAMSM, as there is not enough pair of corresponding feature points to use. So I won't show the 4 pair of corresponding feature points with the AAMSM's at here, I will show it in the next section as a comparison case.

What's next, we will show the result image for the second group satellite images,see Figure 5.16 and 5.17. I just select 2 sets of output images from $10 * 9$ sets of output images as example, see Figure from 5.103 to 5.108. The red dot point which shows on the Figure 5.103, 5.105, 5.106 and 5.108 is the feature point that selected for registration. The selected feature points and the area which near those feature points are matched very well. The

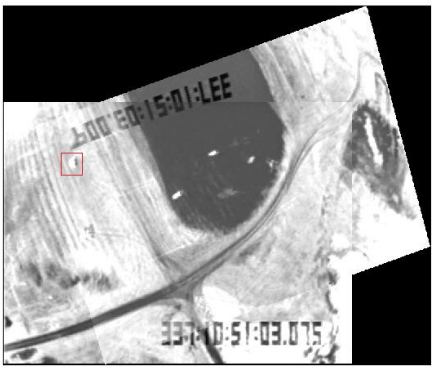

Figure 5.83: 3 point Result image1 of first pair by AAMSM

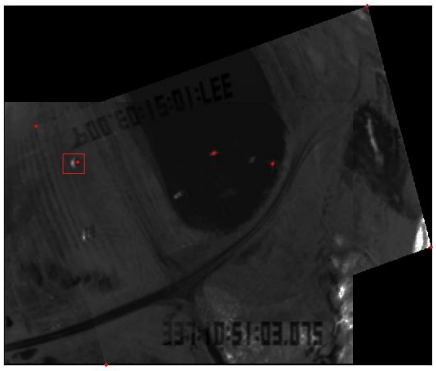

Figure 5.84: 3 point Result image2 of first pair by AAMSM

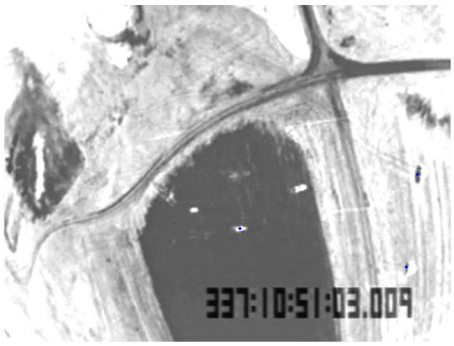

Figure 5.85: 3 point of first pair 1 by AAMSM

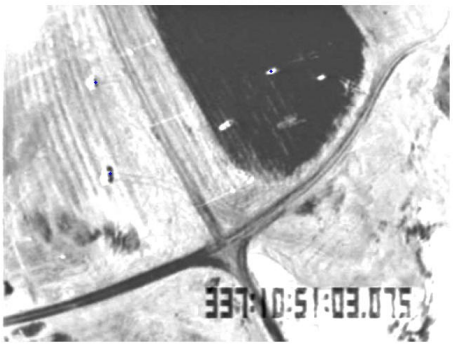

Figure 5.86: 3 point of first pair 2 by AAMSM

first set of output images looks better than the second set of output images. The reason is that the image registration mapping S is closed to the mapping obtained by the correspond feature point in Figure 5.103. So, when we try to calculate the transfer matrix for each points, we can set the weight by the ratio of the error of their one point image registration result image for each feature point when the number of feature points is less than .00001 times the number of image pixel, this may help to obtain a better result image.

Now, Let me show the result image of 3 pair of corresponding feature points for the second group satellite images by using the AAMSM. Look at Figure from 5.109 to 5.114, Figure

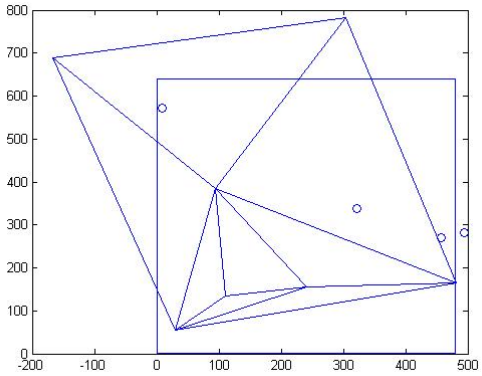

Figure 5.87: 3 point Delaunay Triangulation by AAMSM

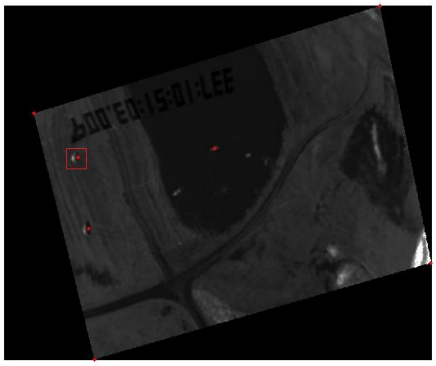

Figure 5.88: transferred image of first pair2 by AAMSM

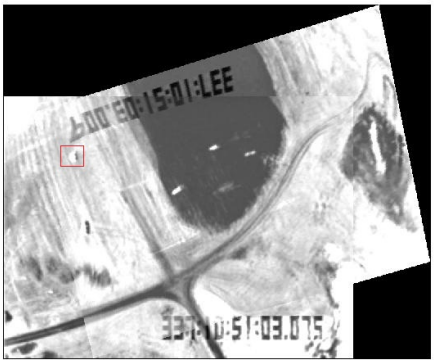

Figure 5.89: 3 point Result image1 of first pair by AAMSM

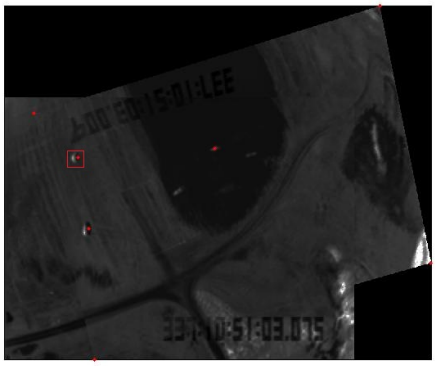

Figure 5.90: 3 point Result image2 of first pair by AAMSM

5.109 and 5.110 show the 3 pair of feature points that we used which show as the red dot points in the figure. Figure 5.111 is the Delaunay Triangulation and the left feature points which didn't use. Figure 5.113 and 5.114 are the result images, as all the feature points which selected belongs to the left hand side of image, so the right-bottom part of result image doesn't match very well. For all other part of result image is matched by our method. As both one feature points case and three feature points case are working well, and the error is reducing as more feature points adding into the method for registration, this mean the AAMSM will still work for other new group of satellite image. As AAMSM need enough

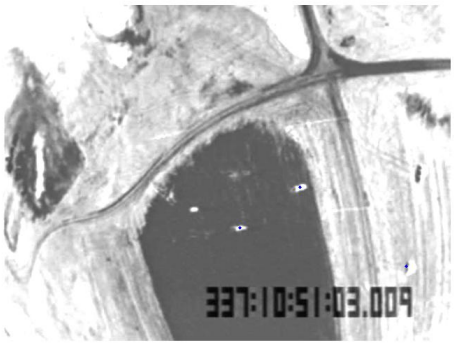

Figure 5.91: 3 point of first pair 1 by AAMSM

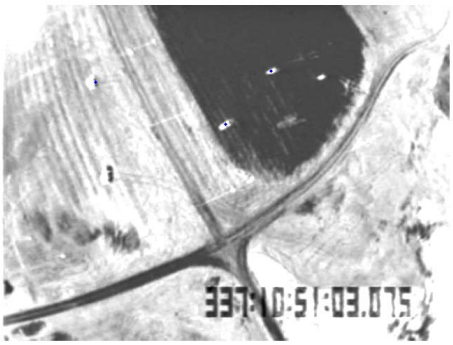

Figure 5.92: 3 point of first pair 2 by AAMSM

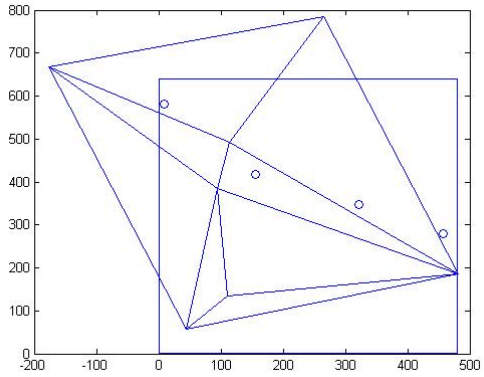

Figure 5.93: 3 point Delaunay Triangulation by AAMSM

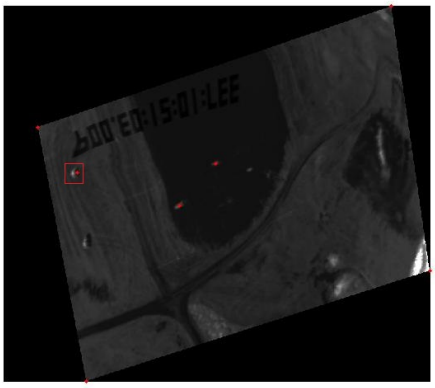

Figure 5.94: transferred image of first pair2 by AAMSM

feature points to show the final result image, so we stop at here without show the 4 or more pair of corresponding feature points result images.

Finally, in order to show the PAMSM will work for all satellite image, let's show some result image for the 2nd group satellite images by using PAMSM. Look at Figure from 5.109 to 5.114 are the result image for 2nd group satellite images by using 3 pair of corresponding feature points, Figure from 5.115 to 5.120 are the result image for 2nd group satellite images by using 4 pair of corresponding feature points. It is very clearly that 1 more feature points adding into our method makes the result image much better, especially the right-bottom

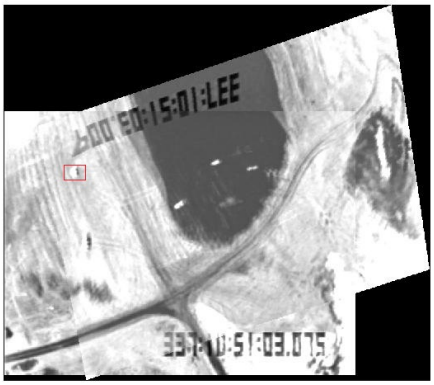

Figure 5.95: 3 point Result image1 of first pair by AAMSM

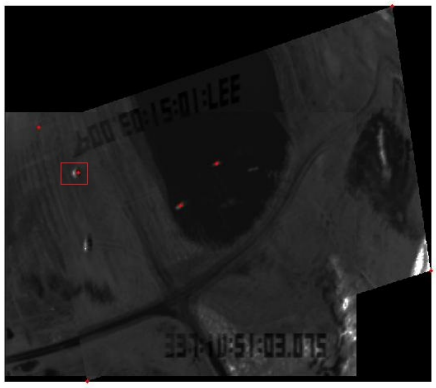

Figure 5.96: 3 point Result image2 of first pair by AAMSM

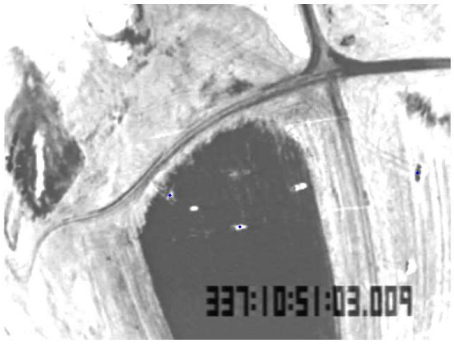

Figure 5.97: 3 point of first pair 1 by AAMSM

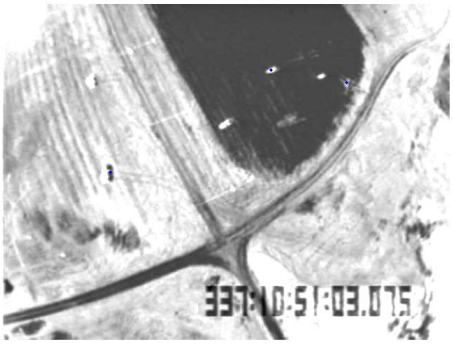

Figure 5.98: 3 point of first pair 2 by AAMSM

part becomes better than 3 pair of corresponding feature point's result image. The error of 4 pair of corresponding feature points is 9.8942, the error of 3 pair of corresponding feature points is 10.0023 which is greater than 9.8942. This show that the PAMSM works well, as more feature points add into our method, the error will go smaller and smaller.

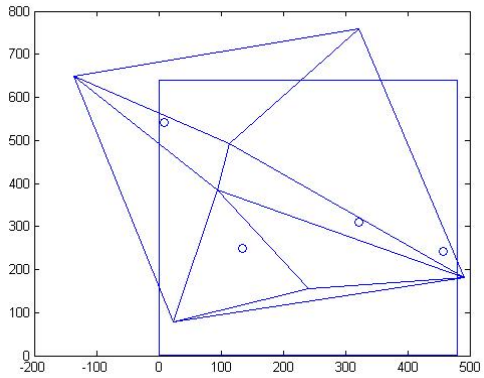

Figure 5.99: 3 point Delaunay Triangulation by AAMSM

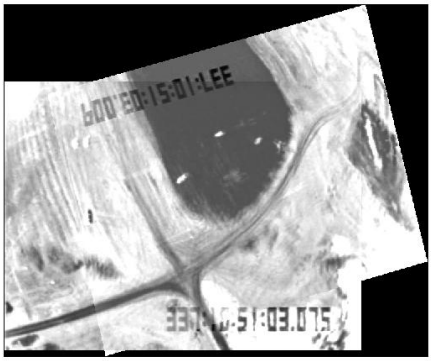

Figure 5.100: transferred image of first pair2 by AAMSM

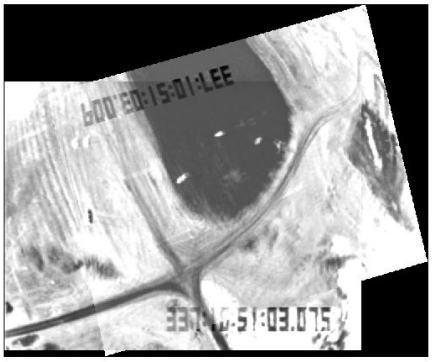

Figure 5.101: 3 point Result image1 of first pair by AAMSM

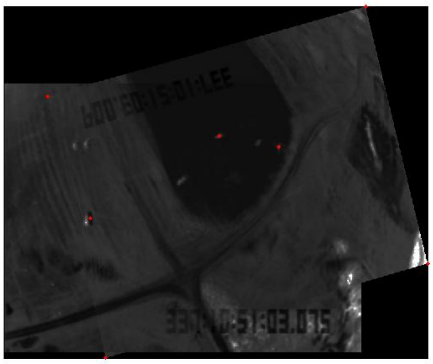

Figure 5.102: 3 point Result image2 of first pair by AAMSM

5.3 The Result Comparison

5.3.1 Comparison for MRI

In this section, I will compare the result between this paper and last paper.

In the last paper, we do image registration by using non-linear local affine transformations.

The main difference between those two methods is as follows:

1. In the last paper, we used the Shepard interpolation, so the interpolation points can not control locally.

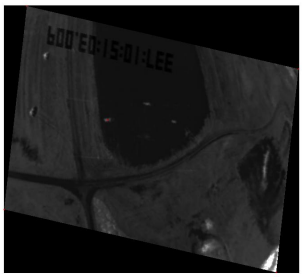

Figure 5.103: transferred image of first pair2

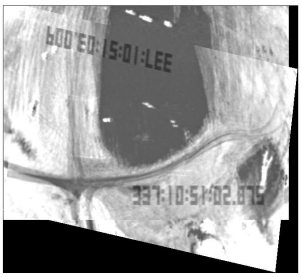

Figure 5.104: 1 point Result image1 of first pair

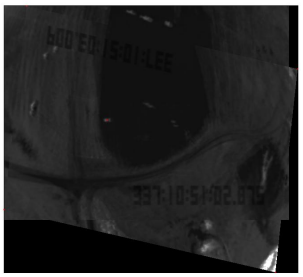

Figure 5.105: 1 point Result image2 of first pair

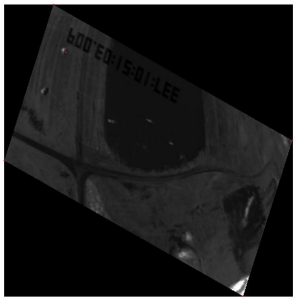

Figure 5.106: transferred image of first pair2

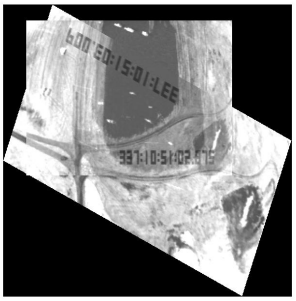

Figure 5.107: 1 point Result image1 of first pair

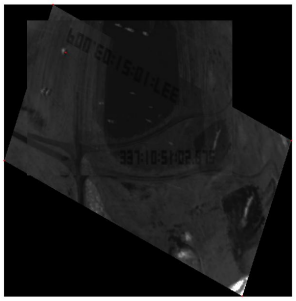

Figure 5.108: 1 point Result image2 of first pair

2. In this paper, we used the multivariate spline function and require smoothness. So, the result is smoothness on the boundary; this is a difference compared to the last paper's result. The results can be seen in Figure 5.127 and 5.128. We use some feature points as input points, Figure 5.127 is the image registration result by using Shepard interpolation, Figure 5.130 is the image registration result by using multivariate spline functions, it is clearly that the result by using multivariate spline function is much better than before.
We add three feature points as input points for the shepard interpolation method, the result is Figure 5.128. Clearly, it is not as good as the result by using multivariate spline functions with 6 interpolation points(Figure 5.130) and 4 interpolation points(Figure 5.129).

Figure from 5.131 to 5.136 are the result image for ANTS, ANTS is designed by UPenn and other University from 2004 until now. ANTS is one of the best automatic-image registration

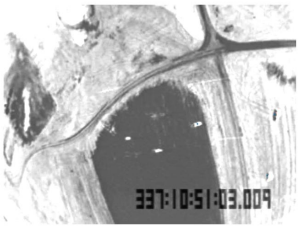

Figure 5.109: 3 point image1 of 2nd group

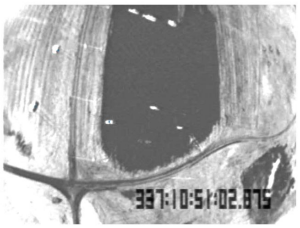

Figure 5.110: 3 point image2 of 2nd group

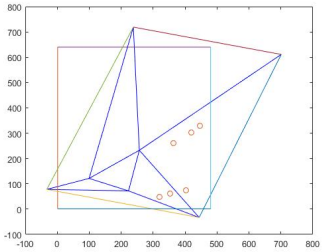

Figure 5.111: 3 point Delaunay Triangulation

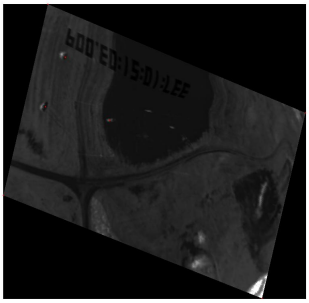

Figure 5.112: transferred image of 2nd group

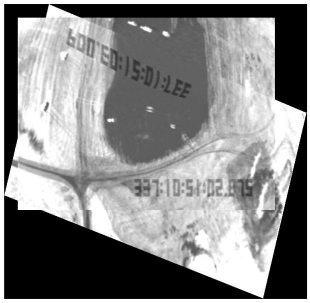

Figure 5.113: 3 point of 2nd group

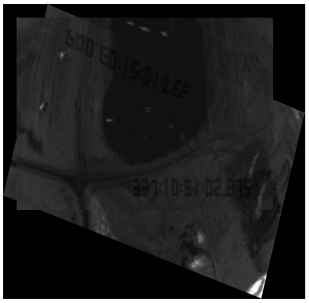

Figure 5.114: 3 point of 2nd group

tool-kit at present. Compare with ANTS result images, it is very clear that the result image which produced by our method is much better.

NOTE: Here is the Similarity Metric Acronyms for ANTS: **CC** = Fast Cross Correlation, **PR** = Cross Correlation, **MSQ** = Mean Squared difference, **MI** = Mutual Information.

5.3.2 Comparison for satellite images

In this section, we will compare the result image for satellite images. We will do comparison with our method between different case. we will also do comparison between our method and other method by using the same input satellite image.

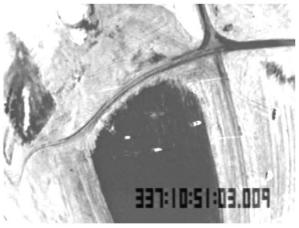

Figure 5.115: 3 point image1 of 2nd group by PAMSM

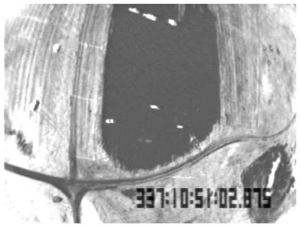

Figure 5.116: 3 point image2 of 2nd group by PAMSM

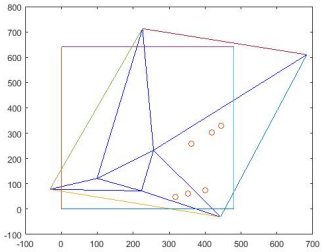

Figure 5.117: 3 point Delaunay Triangulation by PAMSM

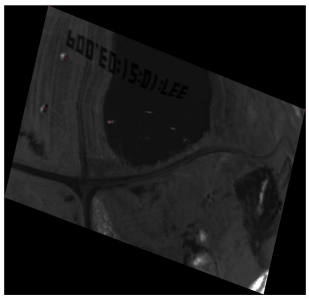

Figure 5.118: transferred image of 2nd group by PAMSM

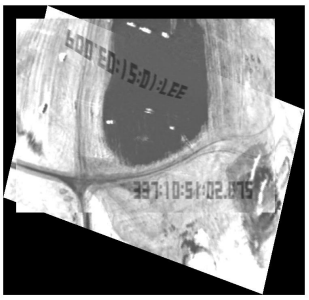

Figure 5.119: 3 point of 2nd group by PAMSM

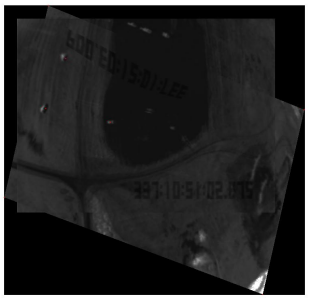

Figure 5.120: 3 point of 2nd group by PAMSM

Firstly, as we talk about in section 5.2, We will show the result image of 4 pair of corresponding feature points by using AAMSM in section 5.3. now, we will do comparison by using the AAMSM with different Adding Point Method which we talked in section 4.1. There are 3 group output image. Figure from 5.137 to 5.139 are result images of 4 feature points by using AAMSM with Adding points Method 1, Figure 5.138 is Delaunay triangulation, Figure 5.137 is the transferred image and the selected feature points which shows as the red dot points. Figure 5.139 is the registration result image. Figure from 5.140 to 5.142 are result images of 4 feature points by using AAMSM with Adding points Method 2, Figure 5.141 is Delaunay triangulation, Figure 5.140 is the transferred image and the selected feature points which shows as the red dot points. Figure 5.142 is the registration result image. Figure from

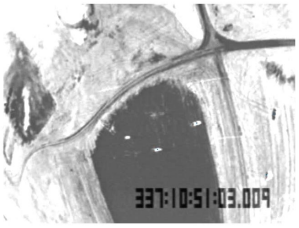

Figure 5.121: 4 point image1 of 2nd group by PAMSM

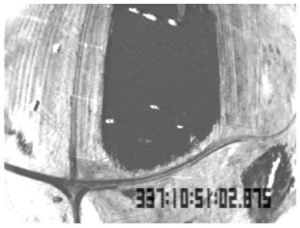

Figure 5.122: 4 point image2 of 2nd group by PAMSM

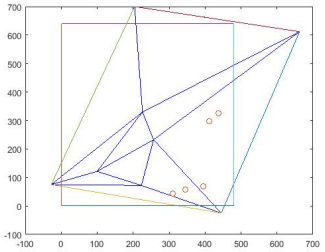

Figure 5.123: 4 point Delaunay Triangulation by PAMSM

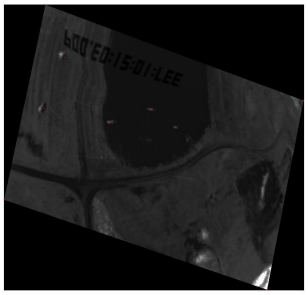

Figure 5.124: transferred image of 2nd group by PAMSM

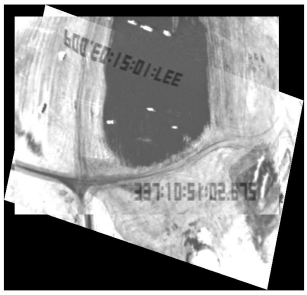

Figure 5.125: 4 point of 2nd group by PAMSM

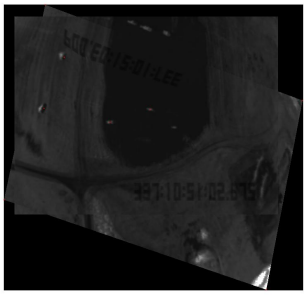

Figure 5.126: 4 point of 2nd group by PAMSM

5.143 to 5.145 are result images of 4 feature points by using AAMSM with Adding points Method 3, Figure 5.144 is Delaunay triangulation, Figure 5.143 is the transferred image and the selected feature points which shows as the red dot points. Figure 5.145 is the registration result image. Comparing with the transferred image, we selected the same 4 feature points to do the image registration for all of those 3 Adding Point Method, and we use the AAMSM for all case. The only difference is that we use different Adding Point Method to do comparison. Compare with the Delaunay Triangulation Figure for those 3 case, one of feature points which is a circle located at right-bottom doesn't belong to the overlap part for the Adding Point Method 1 and 2 case. This is unreasonable. For the Adding Point Method 3 case, all the left 3 feature points belong to the overlap part. This is reasonable,

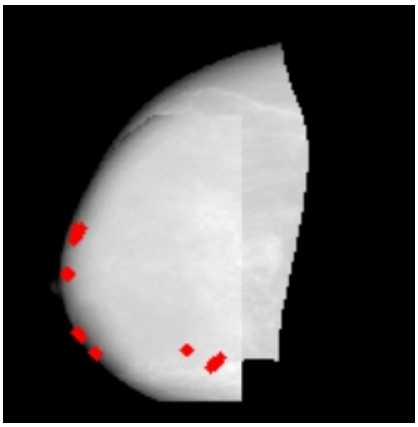

Figure 5.127: 6 Point Shepard Interpolation

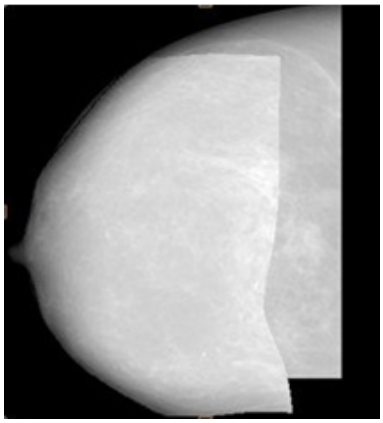

Figure 5.128: 9 Point Shepard Interpolation

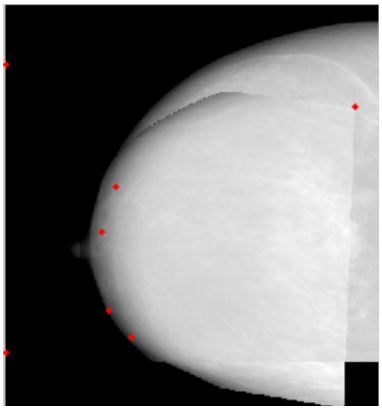

Figure 5.129: 4 Point Multivariate Spline Function

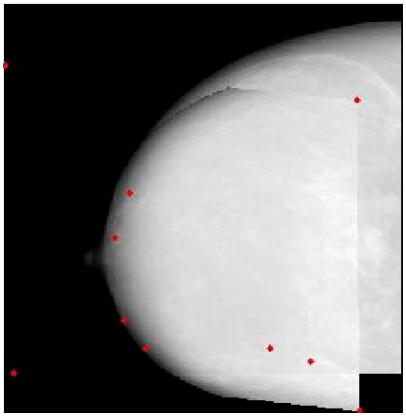

Figure 5.130: 6 Point Multivariate Spline Functions

so Adding Point Method 3 is better with less error than other 2 case. Compare with the result image, the Adding Point Method 3 case is better. Compare with the error, the error of Adding Point Method 1 case is 6.1417, the error of Adding Point Method 1 case is 4.5487, he error of Adding Point Method 3 case is 4.0595. So the Adding Point Method 3 is better than other two cases.

Secondly, We will use two non-automatic image registration methods which called Breast Image Registration Using Non-Linear Local Affine Transformation and Multi-variate Spline Functions Method, and two automatic image registration methods which are ANTS and this

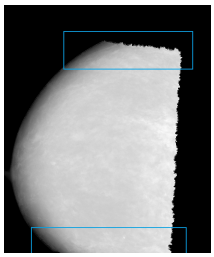

Figure 5.131: CC2

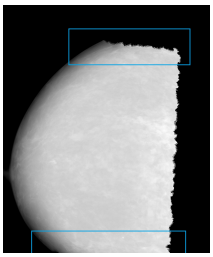

Figure 5.132: CC4

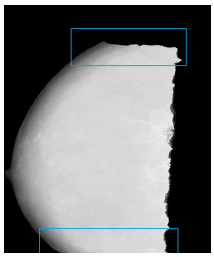

Figure 5.133: MI

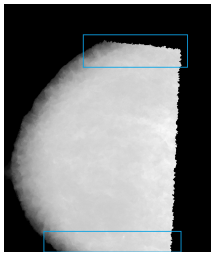

Figure 5.134: PR2

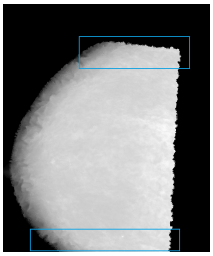

Figure 5.135: PR4

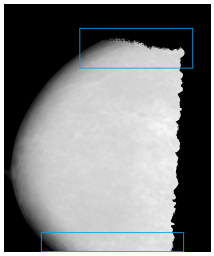

Figure 5.136: MSQ

paper method to do the comparison. Of cause, we will use the same input image, so I use the first group satellite images see Figure 5.14 and 5.15.

Firstly, we do image registration by using 4 pair of corresponding feature points case for comparison. Look at the figure from 5.146 to 5.148. Figure 5.146 is the result image by using the PAMSM with 4 pair of corresponding feature points. Figure 5.147 is the result image by using the Multi-variate Spline Functions with 4 pair of corresponding feature points. Figure 5.148 is the result image by using the Non-Linear Local Affine Transformation with 4 pair of corresponding feature points. Compare this paper result with other two result image, this paper result looks better than the Non-Linear Local Affine Transformation results. Looks as good as the Multi-variate Spline Functions result, the difference is that didn't use same 4 pair of corresponding feature points for registration, although the Multi-variate Spline Functions result is non-automatic image registration. I should say that the edge finding method in this paper works very well.

Secondly, we do image registration by using 5 or 6 pair of corresponding feature points case

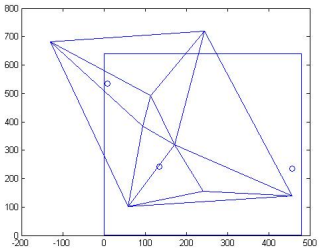

Figure 5.137: Delaunay Triangulation by add1

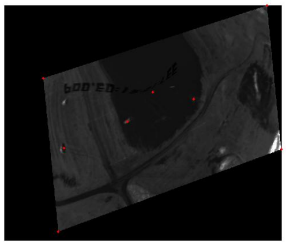

Figure 5.138: transferred image by add1

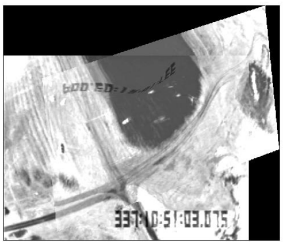

Figure 5.139: result by add1

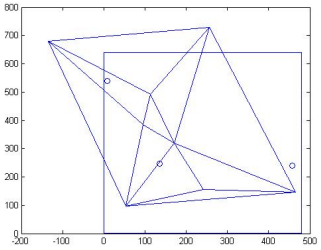

Figure 5.140: Delaunay Triangulation by add2

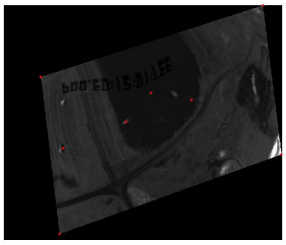

Figure 5.141: transferred image by add2

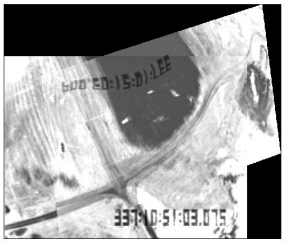

Figure 5.142: result by add2

for comparison. For this paper method, we only use 5 pair of corresponding feature points to do image registration. For other two method, use 6 pair of corresponding feature points to do image registration. Look at the figure from 5.149 to 5.151. Figure 5.149 is the result image by using the PAMSM with 5 pair of corresponding feature points. Figure 5.150 is the result image by using the Multi-variate Spline Functions with 6 pair of corresponding feature points. Figure 5.151 is the result image by using the Non-Linear Local Affine Transformation with 6 pair of corresponding feature points. Compare this paper result with other two result image, this paper result looks much better than the Non-Linear Local Affine Transformation results. Looks as good as the Multi-variate Spline Functions result, as both of them has only one clearly missing match area, although the Multi-variate Spline Functions result is non-automatic image registration and one more feature points is used.

Now, look at the ANTS's result image, see figure from 5.152 to 5.157. In my mind, this

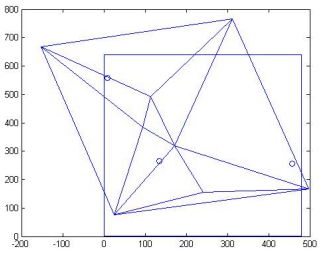

Figure 5.143: Delaunay Triangulation by add3

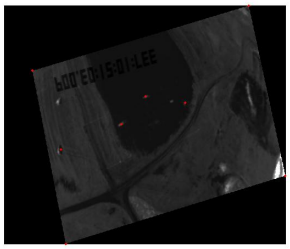

Figure 5.144: transferred image by add3

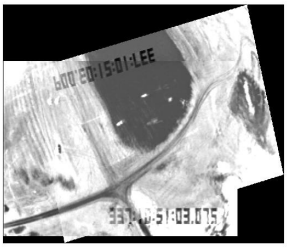

Figure 5.145: result by add3

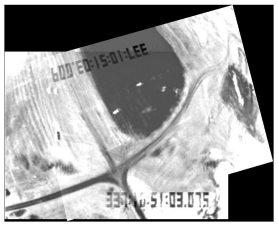

Figure 5.146: 4 point by PAMSM

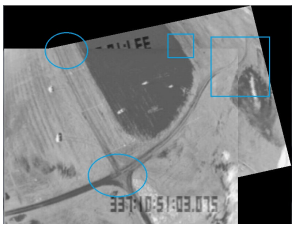

Figure 5.147: 4 point by Multi-variate Spline

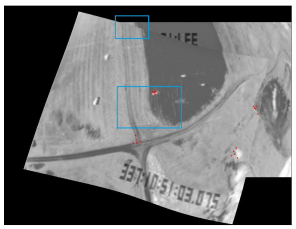

Figure 5.148: 4 point by Non-Linear Local Affine

is really clearly that all the result images from figure 5.146 to figure 5.151 are much better than all of the ANTS result images.

So, this paper's method, called automatic image registration by using multi-variate spline functions method is the best Image Registration Method at present.

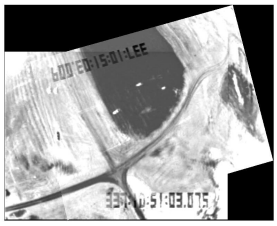

Figure 5.149: 5 point
by PAMSM

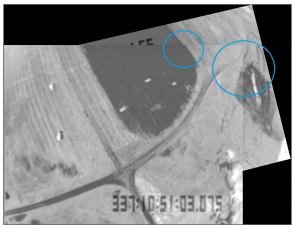

Figure 5.150: 6 point
by Multi-variate Spline

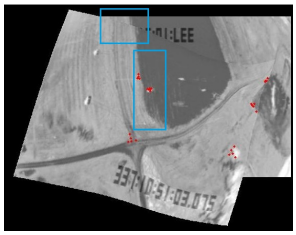

Figure 5.151: 6 point
by Non-Linear Local Affine

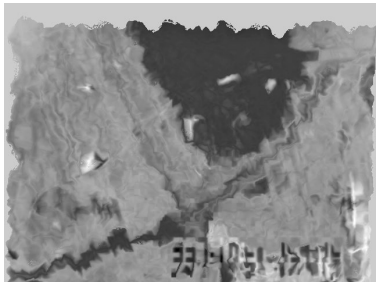

Figure 5.152: CC2

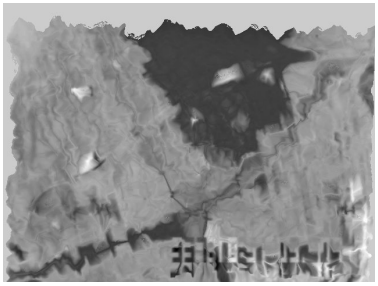

Figure 5.153: CC4

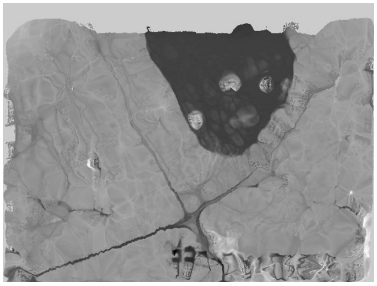

Figure 5.154: MI

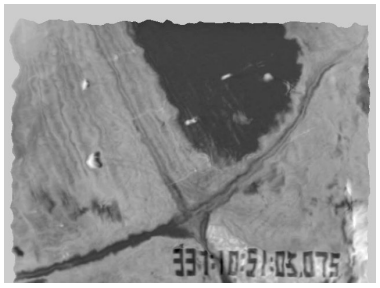

Figure 5.155: PR2

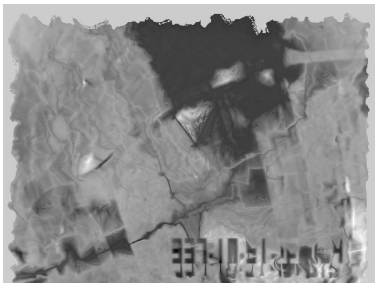

Figure 5.156: PR4

Figure 5.157: MSQ

Chapter 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion for semi-automatic image registration by using Multivariate Spline Function

In this paper, we found the expression of the multi-variate spline function. The principal contributions of this thesis have been demonstrated by a new image registration method by using multi-variate spline functions where this method can be used and works well for all kinds of image registration problems. Compared with the previous similar articles, this paper gives the first on the application of the multivariate spline functions and demonstrates a suitable way to do image registration by using multivariate spline functions. Our main contributions have been:

- In section 3.3.2 N-D M-points Least-Squares Algorithm, the general form of Least-Squares Algorithm(3.4) have been calculated which has m corresponding pair of points and the vector space is n-dimensional with m $\geq$ n+1. Here are the results:

$$X * B * B^T = C * B^T$$

where $X = [A\ b]$, corresponding pair of points $X_i = (x_{i1}, x_{i2}, ..., x_{in})$ and $Y_i = (y_{i1}, y_{i2}, ..., y_{in})$

$$B = \begin{pmatrix} s_1x_{11} & s_2x_{21} & \dots & s_mx_{m1} \\ s_1x_{12} & s_2x_{22} & \dots & s_mx_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ s_1x_{1n} & s_2x_{2n} & \dots & s_mx_{mn} \\ s_1 & s_2 & \dots & s_m \end{pmatrix}$$

$$C = \begin{pmatrix} s_1 y_{11} & s_2 y_{21} & \dots & s_m y_{m1} \\ s_1 y_{12} & s_2 y_{22} & \dots & s_m y_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ s_1 y_{1n} & s_2 y_{2n} & \dots & s_m y_{mn} \end{pmatrix}$$

Where $t_i = s_i^2$ and m $\geq$ n+1.

- In section 3.5.1 generalization of Powell-Sabin Scheme on matrix variables. Theorem 3.5.1 has been proved, which is the generalization of Powell-Sabin Scheme on matrix variables. Here is the theorem:

  Theorem 3.5.1: There is a unique binary quadratic polynomial interpolation function by using method 2.

- In section 3.5.2 expression of multi-variate spline function. The multi-variate spline function has been found to calculate all the affine transformations pixelwisely. This is most important and most difficult part for our method.

6.2 Conclusion for automatic image registration by using Multivariate Spline Function

In this paper, we introduce the automatic image registration by using Multivariate Spline Function, there are two method, one is Adding point Automatic Image Registration by using Multivariate Spline Method, the other is Pre-Selection Method for Automatic Image Registration by using Multivariate Spline Method. Our conclusion are as follow:

- In chapter 4, the new method PAMSM and AAMSM work well and much better than current method such as ANTS. Both of them has error analysis. For AAMSM, the error will go to 0. For PAMSM, the error will go to a quit small number ε which $\varepsilon > 0$.

- In section 4.1, the Adding Point Method 3 is better than other 2 method.
- Compare with AAMSM and PAMSM which is introduce in this paper, AAMSM is fester and give us a better result image when there are not enough pair of corresponding feature points. PAMSM can obtain a better result image when there are enough pair of corresponding feature points.
- There are 3 adding point method, the Adding point Method 3 is better than other two method. it will give us less error result image.
- In chapter 2, we introduce a new method for recognition the object in the given figure. This method works well for satellite image.
- The edge of Object recognition method for multivariate spline method is not works well for MRI breast image. As we lost of focus close curve, See figure 6.1. It just like close curve located everywhere of the image or there is no close curve.

6.3 Future Work

In the future, there are some important and interesting works for us to complete.

- In chapter 2, the Boundary selection Method doesn't work well for MRI image, we need try to find a Method which works for MRI. We will try to improve our method by using Watershed Method and Matrix Expansion Method. By the way, Matrix Expansion is given by Xiquan Shi which is not published yet. She found a base for all dimension Matrix space, and decomposed matrix fast with Fourier Expansion form.

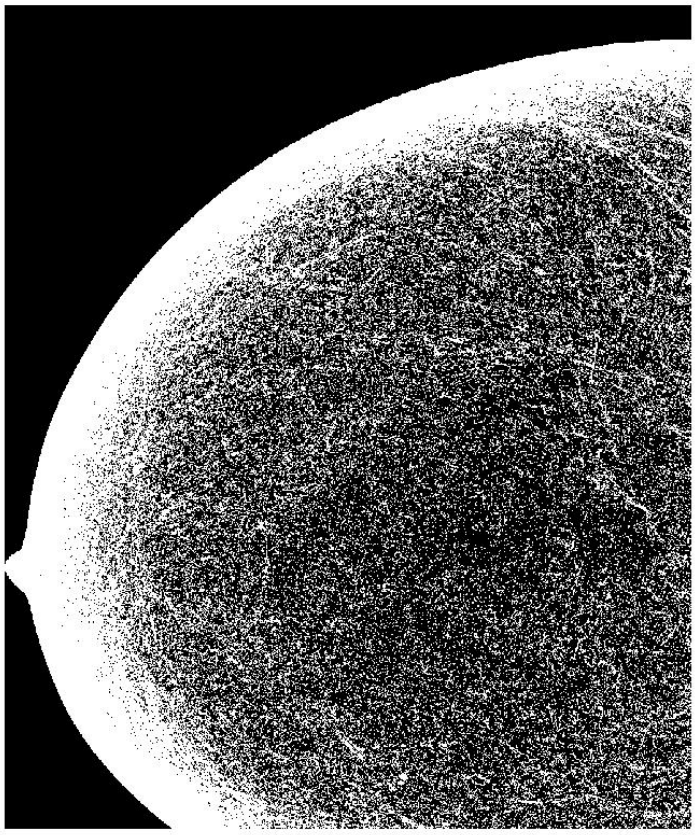

Figure 6.1: MRI bread image by edge find method

- In section 3.2 feature points. We choose the feature points manually. So, in the future, it is very useful to find a good method that can be automatically selected feature points.

- In section 3.5 application of multivariate spline interpolation. We used the binary quadratic polynomial as the function expression on each portions, in the future, we will try to use the binary cubic polynomial as the function expression on each portions, in order to find a good image registration method.

- In section 5.2 the result comparison. We do the comparison with methods, and we will try to do comparison with more others method.

- In order to do the precise error analysis, it is very important that to find a suitable method to do image restoration from 2D to a 3D image. In this way, we can obtain the input image by using the 3D image.

- The boundary selection method should be improved so that it can works well with MRI, such as reference the watershed Method to improved our Method.

[1] Wang Li, Liu R, Zhang L, Duan FQ. The meteorological satellite spectral image registration based on Fourier-Mellin transform. *Spectroscopy and Spectral Analysis*, 33(3):855-858, Mar 2013.

[2] Rong-hong Wang. multivariate spline functions and Their Applications. *Science Press / KLUWER ACADEMIC PUBLISHERS*, 1-2, 2001.

[3] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing.* Second Edition, 3-4, 2001.

[4] Gibson. J.J.. On the analysis of change in the optic array. *Scandinavian J. Psychol.* 18:161-163, 1977.

[5] Nakayama. K. and Loomis. J.M.. Optical velocity patterns. *velocity-sensitive neurons and space perception.* Perception 3:63-80, 1974

[6] Limb. J.O. and Murphy.J.A.. Estimating the vlocity of mowing images in television signals. *Computer Graphics and Image Processing.* 4(4):311-327, 1975.

[7] Jain. R. Militzer. D. and Nagel. H. H.. Segmentation through the detection of changes due to motion. *Computer Craphics and Image Processing* II (1):13-34, 1979.

[8] Jain. R. and Nagel. H. H.. On analysis of accumulative difference pictures from image sequences of real world scenes. *IEEE Trans. on Pattern Analysis and Machin Intelligence* I(2):206-214, 1979.

[9] R. P. Woods, J. C. Mazziotta, and S. R. Cherry. MRI-PET registration with automated algorithm. *J. Comp. Assisted Tomography*, 17(4):536?46, 1993.

[10] M. Betke, H. Hong, and J. P. Ko. Automatic 3D registration of lung surfaces in computed tomography scans. *Fourth International Conference on Medical Image Computing and Computer-Assisted Intervention, Utrecht, The Netherlands*, pp. 725?33, October 2001.

[11] Richard Szeliski and James Coughlan. Spline-Based Image Registration. *Cambridge Research Laboratory, in Cambridge, Massachusetts.* 1994.

[12] D. Rueckert, L. Sonoda, C. Hayes, D. Hill, M. Leach, and D. Hawkes. Nonrigid registration using free-form deformations: application to breast MR images. *IEEE Trans Med Imaging*, 18(8), 712?21, Aug 1999.

[13] Nicholas J. Tustison and James C. Gee. N-D C^k B-spline Scattered Data Approximation. *Penn Image Computing and Science Laboratory, University of Pennsylvania*, 2006.

[14] N. J. Tustison, B. B. Avants, and J. C. Gee. Directly manipulated free-form deformation image registration. *IEEE Trans Image Process.* 18(3):624?35, Mar 2009.

[15] Feiyu Chen, Peng Zheng, Penglong Xu, Andrew D. A. Maidment, Predrag R. Bakic, David D. Pokrajac, Fengshan Liu and Xiquan Shi. Breast Image Registration Using Non-Linear Local Affine Transformation. *Medical Imaging Physics of Medical Imaging.* March 19, 2013.

[16] M. J. D. Powell and M. A. Sabin. Piecewise Quadratic Approximations on Triangles, ACM Transaction on Mathematical Software. 3(4):316-325 December 1977.

[17] Wang Renhong, Wang Wubao, Wang Shoming and Shi Xiquan. The C-Quadratic Spline Space on Triangulations. Acta Mathematicae Applicatae Sinica. 1(2):123-131, 1988.

[18] Les Piegl and Wayne Tiller. The NURBS Book. *Springer*. Second Edition.

[19] Fisher, R. A. "The Use of Multiple Measurements in Taxonomic Problems". Annals of Eugenics. 7 (2): 179188, 1936.

[20] Nobuyuki Otsu. A threshold selection method from gray-level histograms. IEEE Trans. Sys., Man., Cyber. 1979, 9 (1): 6266.