AFFINE IMAGE REGISTRATION USING MINIMUM SPANNING TREE ENTROPIES

By

ADITEE KUMTHEKAR

A THESIS PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE UNIVERSITY OF FLORIDA 2004
Copyright 2004

by

Aditee Kumthekar
This thesis is dedicated to my family whose constant and unconditional support has made it possible.
ACKNOWLEDGMENTS

I would like to thank Dr. Anand Rangarajan for his excellent guidance, for the innumerable hours that he spent patiently resolving my difficulties and for his constant encouragement which was the driving force behind this thesis. I would also like to extend my thanks to Jie Zhang who made me see certain aspects of image registration clearly.
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACKNOWLEDGMENTS</td>
<td>iv</td>
</tr>
<tr>
<td></td>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td></td>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td></td>
<td>ABSTRACT</td>
<td>xii</td>
</tr>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>BACKGROUND</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>2.1 Entropy and Mutual Information</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>2.1.1 Joint and Conditional Entropy</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>2.1.2 Relative Entropy and Mutual Information</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>2.1.3 Differential Entropy</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>2.1.4 Renyi Entropy</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2.2 Histogram method for Estimating Entropy</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>2.3 Minimum Spanning Trees</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>2.4 The Problem of Registration</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>2.5 Image Transformations</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>2.6 Independent Component Analysis</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>THEORY OF MINIMUM SPANNING TREE ENTROPIES</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>3.1 Entropy Estimation for Registration</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>3.2 History</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>3.3 Minimum Spanning Trees for Entropy Estimation</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>3.4 Measure of Registration</td>
<td>28</td>
</tr>
<tr>
<td>4</td>
<td>ALGORITHM DESIGN</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>4.1 General Registration Algorithm</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>4.2 Design of the Algorithm</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>4.2.1 Preprocessing by Histograms</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>4.2.2 Features</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>4.2.3 Search Strategy</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>4.2.4 Minimum Spanning Tree Algorithm</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>4.3 Implementation Details</td>
<td>40</td>
</tr>
</tbody>
</table>
### 4.3.1 Hardware and software Details

#### 4.4 Results

- 4.4.1 Same Image Registered with the Same Image
- 4.4.2 Registration of Same Person with Different Expressions
- 4.4.3 Yet Another Registration of Same Person with Different Expressions
- 4.4.4 Registration Result of Images of Two Different People
- 4.4.5 Yet Another Registration Result of Images of Two Different People
- 4.4.6 Comparison with Histogram Based Method
- 4.4.7 Convergence of Renyi Entropy with Number of Samples as Alpha Changes
- 4.4.8 Noise Trials

### 5 DISCUSSION

### APPENDIX ADDITIONAL INFORMATION

### REFERENCES

### BIOGRAPHICAL SKETCH
### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4–1</td>
<td>The ICA values</td>
<td>34</td>
</tr>
<tr>
<td>4–2</td>
<td>Free parameter table</td>
<td>44</td>
</tr>
<tr>
<td>4–3</td>
<td>Noise results over rotation for histogram</td>
<td>61</td>
</tr>
<tr>
<td>4–4</td>
<td>Noise results over rotation for MST</td>
<td>62</td>
</tr>
<tr>
<td>4–5</td>
<td>Noise results over rotation and scale for MST</td>
<td>63</td>
</tr>
<tr>
<td>4–6</td>
<td>Noise results over rotation and scale for MST</td>
<td>63</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Two registered images with different intensity distributions</td>
<td>4</td>
</tr>
<tr>
<td>2-1</td>
<td>Original graph</td>
<td>14</td>
</tr>
<tr>
<td>2-2</td>
<td>MST by Kruskals Algorithm</td>
<td>14</td>
</tr>
<tr>
<td>4-1</td>
<td>Block diagram for image registration</td>
<td>30</td>
</tr>
<tr>
<td>4-2</td>
<td>Flow-chart for the estimating the parameters of the transform</td>
<td>31</td>
</tr>
<tr>
<td>4-3</td>
<td>Effect on registration by varying the number of bins of histogram(Number of Bins are 8, 16 and 32 from top left to bottom right. Maximum should be present at 0.)</td>
<td>33</td>
</tr>
<tr>
<td>4-4</td>
<td>ICA feature images</td>
<td>35</td>
</tr>
<tr>
<td>4-5</td>
<td>The faces used to compare registration for different sizes of ICA blocks</td>
<td>36</td>
</tr>
<tr>
<td>4-6</td>
<td>Figure showing the variations in MI vs. the rotation angle various ICA feature dimensions 1. 4 features 2. 16 features 3. 36 features. Maximum located at 0.</td>
<td>36</td>
</tr>
<tr>
<td>4-7</td>
<td>Figure showing the variations in MI vs. the rotation angle various ICA feature dimensions 1. 4 features 2. 16 features 3. 36 features</td>
<td>37</td>
</tr>
<tr>
<td>4-8</td>
<td>Behavior of MST entropy in 1D and 2D search space</td>
<td>38</td>
</tr>
<tr>
<td>4-9</td>
<td>Plots showing error between the actual MST length and the truncated MST length over various radius factors and over rotations of -10, 0 and 10 degrees</td>
<td>41</td>
</tr>
<tr>
<td>4-10</td>
<td>Figure showing the variations in joint entropy versus the rotation angle with varying number of samples picked from the overlap area when alpha is 0.55 are shown as 1. 100 Samples 2. 300 samples 3. 500 samples 4. 700 samples</td>
<td>42</td>
</tr>
<tr>
<td>4-11</td>
<td>Figure showing the variations in joint entropy versus the rotation angle with varying number of samples picked from the overlap area when alpha is 0.55 are shown as 1. 900 Samples 2. 1100 samples</td>
<td>42</td>
</tr>
</tbody>
</table>
4–12 Figure showing the variations in joint entropy versus the rotation angle with varying number of samples picked from the overlap area when alpha is 0.85 are shown as 1. 100 Samples 2. 300 samples 3. 500 samples 4. 700 samples
4–13 Figure showing the variations in joint entropy versus the rotation angle with varying number of samples picked from the overlap area when alpha is 0.85 are shown as 1. 900 Samples 2. 1100 samples
4–14 Figure showing the from top left to bottom right 1. Face image 2. Same face with applied transformation 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images
4–15 Minimum spanning tree of the unregistered images considering only the intensities of the two images as features
4–16 Figure showing the from top left to bottom right 1. Face image 2. Face obtained after applying the optimal transformation to the unregistered image shown in the figure 4–14 3. Overlap of the two registered images 4. Edge image of the overlapped registered images
4–17 Minimum spanning tree of the registered images considering only the intensities of the two images as features
4–18 Figure showing the from top left to bottom right 1. Face image 2. Face image of the same person with glasses which is unregistered 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images
4–19 Minimum spanning tree of the unregistered images considering only the intensities of the two images as features
4–20 Figure showing the from top left to bottom right 1. Face image 2. Face image of the same person with glasses which is registered 3. Overlap of the two registered images 4. Edge image of the overlapped registered images
4–21 Minimum spanning tree of the registered images considering only the intensities of the two images as features
4–22 Figure showing the from top left to bottom right 1. Face image of a happy person 2. Face image of the same person with sad expression which is unregistered 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images
4-23 Figure showing the from top left to bottom right 1. Face image
2. Face image of the same person with sad expressions which is
registered 3. Overlap of the two registered images 4. Edge image
of the overlapped registered images

4-24 Figure showing the from top left to bottom right 1. A Face image 2.
Face image of the different person 3. Overlap of the two unregistered
images 4. Edge image of the overlapped unregistered images

4-25 Figure showing the from top left to bottom right 1. Face image 2.
Face image of different person which is registered 3. Overlap of the
two registered images 4. Edge image of the overlapped registered
images

4-26 Figure showing the from top left to bottom right 1. A Face image 2.
Face image of the different person 3. Overlap of the two unregistered
images 4. Edge image of the overlapped unregistered images

4-27 Figure showing the from top left to bottom right 1. Face image 2.
Face image of different person which is registered 3. Overlap of the
two registered images 4. Edge image of the overlapped registered
images

4-28 Figure showing the from top left to bottom right 1. A Face image 2.
Face image of the different person 3. Overlap of the two unregistered
images 4. Edge image of the overlapped unregistered images

4-29 Figure showing the from top left to bottom right 1. Face image 2.
Face image of different person which is registered 3. Overlap of the
two images registered using histogram 4. Edge image of the
overlapped registered images

4-30 Figure showing from left to right 1. Face image of a person 2. Face
image of same person which is registered

4-31 Figure showing image 2 as registered by the histogram

4-32 Plot of MI vs. rotation angle when registered by histogram. Maximum
should be at -5 degrees.

4-33 Figure showing image 2 as registered by the MST

4-34 Plot of MI vs. rotation angle as registered by MST. Correct answer
is at -5 degrees.

4-35 Figure showing the from top left to bottom right, MST Renyi joint
entropy vs the samples for 1. alpha = 0.4 2. alpha = 0.45 3.alpha
= 0.5 4. alpha = 0.55
4-36 Figure showing the from top left to bottom right, MST Renyi joint entropy vs the samples for 1. alpha = 0.6 2. alpha = 0.65 3. alpha = 0.7 4. alpha = 0.75 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
AFFINE IMAGE REGISTRATION USING MINIMUM SPANNING TREE
ENTROPIES

By

Aditee Kumthekar

May 2004

Chair: Anand Rangarajan
Major Department: Computer and Information Science and Engineering

The various environmental factors, settings of the image acquisition device and the material properties of a object directly affect the formation of an image. Registration of two images involves aligning the two images obtained under different conditions which is based on several information-theoretic methods. Entropy estimation is an important step in these methods which traditionally requires estimation of the density function of the images. This thesis however directly estimates the entropy without a priori default density estimation. This is achieved by exploiting a powerful theorem which relates the asymptotic length of the minimum spanning tree formed from the samples and the Renyi entropy of the density function underlying the samples. Since the density estimation step is avoided, the careful tuning of several density estimation parameters, which is required by most density estimation methods, is avoided. Also, since the spanning tree method estimates the entropy in high dimensions, this allows us to include feature information into the registration process. Affine registration of face images of different subjects with different gestures and taken under different lighting conditions is demonstrated.
CHAPTER 1
INTRODUCTION

The entire field of computer vision is motivated to solve the problem of designing a system which has the capabilities of human vision. When we look around, our eyes and our brain immediately analyze the relationship between the different objects in our immediate environment. We usually do not have to think long to interpret what we see. It comes naturally to us. However, we have not been able to emulate this amazing system of perception after 40 years of the inception of the field of computer vision. Why is this so? This is because a very complex relationship exists between the physical factors and the way an image is formed. Such factors include reflectivity and absorbency of the illuminated surface, the intensity and the direction of illumination, the environmental factors like the presence of dust and speckle in the atmosphere, rain etc. Not only this but the mode in which an image is acquired – for example, a camera and the parameters that calibrate the instrument – for example, the shutter speed in the camera contribute to the image formation. Viola and Wells [28] have represented this event of image formation as an equation as

\[ v(T(x)) = F(u(x), q), \]  

(1.1)

where \( T(x) \) is the transformation function which determines the orientation of the image; \( u(x) \) is the function which describes the surface of the object; \( q \) represents the other parameters that affect image formation such as lighting, environment etc; \( v() \) is the function that formulates the pixel intensities and \( F(u(x), q) \) is the function which incorporates the various parameters mentioned above which play a
role in the image formation. Modelling the aforementioned function is not a trivial task as there are a vast number of parameters involved.

Many vision related systems attempt having functionalities such as object recognition – identifying the objects from the background region, indexing of image databases – the efficient retrieval of images similar to the query image and object modelling – the mathematical representation, coding and synthetic generation of an image from a model. Many such applications require that the images under consideration be aligned before further processing can be performed on them. Apart from these applications which require alignment as a initial preprocessing step, there are a plethora of applications which rely on accurate image alignment. One such application is "anomaly detection" in medical imaging, which refers to detecting any changes in a particular organ over a period of time either due to disease or to due to treatment, e.g., detecting if the post-operative MRI of patient shows the removal of the tumor that was seen in the pre-operative scan of the patient. In order to do this, we require accurate alignment of the images.

Image registration is precisely this task of spatially aligning two images so that the corresponding pixels in the images refer to the same portion or area in the image. Alternatively, you could also think of image registration as finding the transformation which aligns the two images accurately. You might now wonder why is this not a simple task. Let us see why. We saw that there was a wide array of factors that contribute to the image formation. The challenge of registration lies in matching two images when they are taken at different times, under different conditions and using different viewpoints probably with different image acquisition techniques. Recently, the problem of multimodality based image registration, which refers to registering images under different lighting conditions, has gained importance.
At the core, registration methods can be viewed as a combination of certain choices about the following parameters, namely [4]

- **Feature Space**: Feature space refers to the parameters of the image used to perform registration. In some techniques, the raw image intensities are used directly. These are called intensity based methods. Also, there are feature based methods which extract some information from the raw image (termed as features) and then use these features to perform registration. The features can be extracted via image segmentation or by using information theory based technique.

- **Search space**: This is the set of various transformations applied to the image during the registration process. The spatial transformations run the gamut of rigid, affine, projective, polynomial, non-rigid (morphing) etc. These are explained later.

- **Search strategy**: This term denotes the technique used to search the entire set of transformations. In case of the exhaustive search, the entire space is scanned. In the case of hierarchical search, the set of transformations is searched using a coarse to fine hierarchy. Finally, there are techniques that use local gradient information of image similarity measure to determine a sequence of transformations which progressively minimizes the measure.

- **Similarity metric**: It is the criterion used for judging the quality of the achieved registration. The similarity metric is a figure of merit used to rank order the various spatial transformations.

One might wonder what is this similarity metric anyway? Can’t we simply compare the intensities at the corresponding pixel locations to verify if the images have been registered. This is not so straightforward. In figure 1–1, we show two registered images A and B. Clearly, the intensities in the two images at the same location do not correspond.

There have been many recent efforts focused on information theory based methods. The information-theory based criterion usually provides us with some registration objective function which has to be minimized (or maximized) over a range. One such objective function is the mutual information objective function [28] which was used to perform multimodality based image registration. Mutual Information is a measure which measures the amount of overlapping information content between two images. An interesting metaphor for intensity
Figure 1–1: Two registered images with different intensity distributions

based and information-theory based similarity measures is that the intensity-based technique can be viewed as "What We See Is All Get (WWSIAWG)" while the information-theory based criteria can be viewed as "What We See Is What We Mean (WWSIWWM)". Mutual Information techniques have now become widespread in registration especially in medical imaging.

The Mutual Information between any two images can be computed either from the raw image intensity or from feature vectors extracted from each image. Rarely do we see registration approaches that combine intensity and features. This thesis attempts such a combination.

Histograms are commonly used to estimate the entropy (information content) which is then used to calculate the Mutual Information. When feature based methods are used, the dimensionality of the feature space increases. Histogram methods suffer from the "curse of dimensionality" and thus cannot be easily used to estimate the entropy of a high dimensional feature set. Also the
histogram based methods are extremely sensitive to the number of bins. Apart from this the histograms estimate the discrete Shannon Entropy while, as we explain later, we need to estimate the continuous differential entropy. This calls for the use of non-parametric methods to estimate the entropy in a high dimensional feature space.

Once such technique is to use spanning graphs to estimate entropy. The early work in this area was done by Steele [25] who showed how the length of the minimum spanning tree (MST) of a complete graph from the Euclidean distance between the feature vectors is directly proportional to the integral of an appropriately defined power of the probability density defined on the feature space. Later the group at University of Michigan, Alfred Hero et al. [12, 19] showed how the aforementioned relation between the length of the spanning tree and the integral of the power of the density can be used to estimate the Renyi entropy which in turn can be used to perform registration.

The search space of affine transformations is one such space which includes the six parameters of rotations, scale, shear and translations in $R^6$. The affine transformation is applied globally to the entire image.

In this thesis, we have closely followed Hero’s work and have successfully shown how the MST based approach can be used the to perform registration over six parameters of the affine transformation. We have computed an information-theory based criterion similar to the conventional mutual information using the Renyi entropy as proposed in paper [14] and then minimized it over an affine search space. Also, we have combined the use of intensities and features in the same information theory based registration technique. The Renyi entropy is estimated

\footnote{Information on the entropy, its types and Mutual Information is presented in Chapter 2.}
from the high dimensional features space using the minimal spanning tree-based method.

After this concise introduction, Chapter 2 covers the background material, Chapter 3 covers the theory, Chapter 4 covers the implementation details and finally, the last chapter covers the results, conclusions and future work.
CHAPTER 2
BACKGROUND

2.1 Entropy and Mutual Information

From a broad perspective, the entropy of a random variable is defined as the measure of uncertainty of a random variable. Alternatively, it can also be defined as the amount of information required to represent the variable. Thus, the more improbable a random the variable, greater is its entropy. Shannon \cite{24} derived a measure of entropy for discrete variables. Mathematically, the entropy of \( H(X) \) of a random variable \( X \) is defined as

\[
H(X) = -\sum_{x \in \chi} p(x) \log p(x).
\tag{2.1}
\]

Normally, the logarithm is taken to the base 2, in which case the entropy is expressed in terms of bits. Thus, entropy can also be expressed as the length of the code to represent the random variable. If \( g(x) \) are the actual values that a variable \( X \) can assume, then the expected values of \( g(X) \) is written as

\[
E_p g(X) = \sum_{x \in \chi} g(x)p(x).
\tag{2.2}
\]

Thus, we can write the entropy of a random variable as

\[
H(X) = E_p \log \frac{1}{p(X)}.
\tag{2.3}
\]

This leads to a conclusion that

\[
H(X) \geq 0 \text{ as } 0 \leq p(x) \leq 1 \text{ implies that } \log \frac{1}{p(X)} \geq 0.
\tag{2.4}
\]
2.1.1 Joint and Conditional Entropy

Joint and conditional entropy extend the concept of entropy to a pair of distinct random variables. Two random variables are said to be statistically independent when their joint density is the product of their marginal densities. Thus, when the two random variables are independent, the knowledge of one does not have any effect on the knowledge of the other variable. On the other hand, when the variables are totally dependent, that is, if $Y = f(X)$ then, $Y$ is not random when $X$ is known.

The mathematical representation of joint entropy is

$$H(X, Y) = - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(x, y)$$

and the conditional entropy of a variable given the other variable is defined as

$$H(Y|X) = - \sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \log p(y|x) = -E_{p(x,y)} \log p(Y|X).$$

By the chain rule we get

$$H(X, Y) = H(X) + H(Y|X).$$

Two variables are said to be independent if

$$H(Y|X) = H(Y)$$

that is knowledge of $X$ has no effect on the knowledge of $Y$. Thus we have

$$H(X, Y) = H(X) + H(Y).$$

2.1.2 Relative Entropy and Mutual Information

The relative entropy is the measure of the distance between the two probability distributions. To explain it better, suppose there a random variable
with probability distribution \( p \) and whose entropy is \( H(p) \). Since we don’t know the exact distribution of \( p \) we assume that it is \( q \) with entropy \( H(q) \). The relative entropy \( D(p||q) \) is the difference in the number of entropy bits of \( p \) and \( q \).

It is also called as the Kullback-Liebler distance and is mathematically represented as

\[
D(p||q) = \sum_{x\in X} p(x) \log \frac{p(x)}{q(x)}
\]  

(2.11)

Mutual information \( (I(X;Y)) \) is defined as the amount of information that one random variable has about other. It is the relative entropy between the joint and the product distributions of the two random variables, expressed as

\[
I(X;Y) = D(p(x,y)||p(x)p(y))
\]

\[
= \sum_{x\in X} \sum_{y\in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}.
\]

(2.13)

It can also be shown that

\[
I(X;Y) = H(X) + H(Y) - H(X,Y)
\]

\[
= I(Y;X) + I(X;Y) \geq 0.
\]

(2.15)

Additional properties of mutual information are given in the book by Cover and Thomas [8].

2.1.3 Differential Entropy

While entropy is a measure of uncertainty in the discrete domain, differential entropy is a measure of uncertainty of a variable in the continuous domain. It is defined as

\[
h(X) = -E_x[\log(p(X))]
\]

\[
= -\int_{-\infty}^{\infty} p(x) \log(p(x))dx
\]

(2.16)
The major difference between entropy and differential entropy is that the
differential entropy can take negative values. Thus, there is no direct relationship
between code length and differential entropy since code length cannot be negative.

2.1.4 Renyi Entropy

The entropy described in Section 2.1.3 is differential shannon entropy. Renyi
entropy is a generalized version of the Shannon entropy. The Renyi entropy \( h_\alpha(X) \)
of order \( \alpha \) is defined as,

\[
h_\alpha(X) = \frac{1}{1 - \alpha} \log \left[ \int f^\alpha(x)dx \right], \quad (2.18)
\]

for \(-\infty < \alpha < \infty, \alpha \neq 1\).

The discrete version of the Renyi entropy is defined as,

\[
H_\alpha(X) = \frac{1}{1 - \alpha} \log \left[ \sum_{i=1}^{N} x_i^\alpha \right]. \quad (2.19)
\]

When \( \alpha \rightarrow 1 \) we obtain the Shannon entropy function from the Renyi entropy
function as follows,

\[
h(X) = h_1(X) = - \int p(x) \log(p(x))dx \quad (2.20)
\]

In the discrete domain, as \( \alpha \rightarrow 1 \), \( H_\alpha(X) = H(X) \).

Viola and Wells [28] have stated that the task of registering two images is
equivalent to maximizing the mutual information between the two images.

2.2 Histogram method for Estimating Entropy

Histogram based density estimation divides the range of the data into bins
and then computes the normalized frequency of the data placed in the bins. The
normalized frequency counts sum up to one. A one dimensional histogram of data
with \( n \) samples and \( b \) bins is,

\[
f(a_i) = \frac{\text{count}a_i}{n} \quad \text{where } a_i \text{ denotes the bin number } i \text{ and } 0 < i \leq b. \quad (2.21)
\]
For multivariate data distributed across \( d \) dimensions, histogram-based density estimation by histograms is generally performed by constructing \( M \) bins along each of the \( d \) dimensions. If the number of bins is large then the histogram will fail to capture the shape of the probability distribution function due to high variance will be large and the reduced bias. However, as number of bins are decrease, the probability distribution function becomes spiky as the variance narrows and the bias increases. Algorithms that determine the correct bin size of the histogram include the Sturge’s algorithm \cite{26} which states that for \( n \) samples, \( M = 1 + 3.322 \times \log_{10} n \). Some other approaches predict bin size from a priori knowledge of data. In our thesis, we have chosen the bin size statistically.

Histogramming suffers from the curse of dimensionality. That is, if the number of data dimensions are denoted by \( d \), with \( M \) bins in each direction then the total number of bins will be \( M^d \). Such exponential growth in \( M \) incurs drawbacks. Firstly, as the \( M \) is large in higher dimensional data, more data points will be needed to populate the histogram. Otherwise, it will result in a sparsely populated histogram with numerous empty bins. Secondly, data in higher dimensions tends to be correlated and thus projects itself into selected dimensions. This makes histogram based method of density estimation unsuitable for higher dimensional data.

\section*{2.3 Minimum Spanning Trees}

Before describing the theory behind the use of Minimum Spanning Trees for registration later in chapter 3, let us see define a Minimum Spanning Tree. Minimum Spanning Trees(MSTs) are used in solving many real world problems. For example, consider a case of a network with \( V \) nodes with \( E \) undirected connections between nodes. This can be represented as a connected, undirected graph \( G = (V, E) \) containing \( V \) vertices and \( E \) edges. Now suppose that all the edges are weighted, i.e., for each edge \((u, v) \in E\) we have an associated weight
A weight can be used to represent real world quantities such as cost of a wire, propagation delay etc between two nodes in a network. A spanning tree is defined as a acyclic graph that connects all the vertices. A minimum spanning tree is a spanning tree with the minimum weight. Suppose we represent the spanning tree as $T \subseteq E$, which connects all the vertices, and whose total length is $w(T)$, then the minimum spanning tree is defined as,

$$\min(w(T)) = \sum_{(u,v) \in T} w(u, v).$$

**(2.22)**

**Generic MST algorithm.** The book by Cormen et al. [7] gives a supported analysis of minimum spanning tree algorithms. The MST algorithm falls in the category of greedy algorithms. Greedy Algorithms are algorithms that make the best choice at each decision making step. In other words, at every step, greedy algorithms make the locally optimum choice and hope that it leads to a globally optimum solution. The greedy MST algorithm builds the tree step-by-step, incorporating the edge that causes minimum increase in the total weight at each step, without adding any cycles to the tree. Suppose there is a connected, undirected graph $G = (V, E)$ with the weight function $w : E \rightarrow R$. While finding the minimum spanning tree for graph $G$, the algorithm manages at each step an edge-set $S$ which is some subset of the MST. At each step, edge $(u, v)$ is added to subset $S$ such that it does not violate the MST property of $S$. This makes $SU(u, v)$ a subset of the Minimum Spanning Tree. The edge which is added at each step is termed a ”safe edge”. The algorithm can be written as

**GENERIC-MST(G, w)**

1. $S \leftarrow \emptyset$
2. **while** $S$ does not form a spanning tree
3.  **do** find a safe-edge $(u,v)$ which can added in $S$
4.  $S \leftarrow S \cup (u,v)$
5. return $S$

There are two popular algorithms for computing the Minimum Spanning Tree, Prim’s algorithm and Kruskal’s algorithm (refer [7]). We have used the Kruskal’s Algorithm to find the Minimum Spanning Tree. Its description follows.

Kruskal’s algorithm for MST. Kruskal’s algorithm is an extension of the generic MST algorithm described in the preceding sub-section above. In the Kruskal’s algorithm the set $S$, which is a subset of the minimum spanning tree, is a forest. At each step, the Kruskal’s Algorithm finds the safe edge to be added as the edge with the minimum weight that connects two forests. Initially, the edges are sorted in the decreasing order of their weights. At each step, one finds the minimum edge in the graph not already present in the minimum spanning tree, connects two forests together. This process is repeated until, till all the vertices are included in the graph. The algorithm can be written as,

MST-Kruskal(G,w)
1. $S \leftarrow \emptyset$
2. for each vertex $v \in V[G]$
3. do MAKE-SET(v)
4. sort the edges of $E$ by non-decreasing weight $w$
5. for each edge $(u, v) \in E$, in the order of nondecreasing weight
6. do if FIND-SET($u$) $\neq$ FIND-SET($v$)
7. then $S \leftarrow S \cup (u, v)$
8. UNION(u,v)
9. return $S$

In steps 1-3, the subset $S$ is initialized to null and $V$ number of forests each with a single vertex are created. Step 4 sorts the edge set $E$ in a non decreasing order of weight. In steps 5-8, an edge $(u, v)$ is found such that the endpoint $u$ belongs to one forest and endpoint $v$ belongs to other forest. This edge is incorporated in
the subset $S$. The algorithm stops when all vertices are included in the tree. An
illustration of how the Kruskal’s algorithm works is shown in figure 2–2,

A more formal proof of why the Kruskal’s Algorithm works [15] is as follows

Let $G$ be a undirected, connected graph. $t$ is the minimum spanning tree by
Kruskal’s algorithm and $t'$ is the actual minimum spanning tree. One shows that
trees \(t\) and \(t'\) have same costs. Let \(E(t)\) and \(E(t')\) are the edges in both the trees where each set contains \(n - 1\) edges, provided that the total number of vertices in a graph are \(n\).

**PROOF:**

Case 1: If \(E(t) = E(t')\) then the lengths \(t\) and \(t'\) are equal. Thus Kruskal’s Algorithm generates the minimum spanning tree in this case.

Case 2: \(E(t) \neq E(t')\). Assume that there is an edge \(p\) such that \(p \in E(t)\) and \(p \notin E(t')\). If we include \(p\) in \(t'\), then \(t'\) will have \(n\) edges and, thus, a unique cycle in \(t'\) with edges \(p, e_1, e_2, \ldots, e_k\). Of these \(k + 1\) edges, there will be at least one edge \(e_j\) not in tree \(t\). The weight of \(e_j\) must be greater than or equal to the weight of \(p\) because, if the weight is less than that of \(p\), then the greedy Kruskal’s algorithm would include it while forming the tree \(t\). If any such edge which forms a cycle is removed from tree \(t'\) a different tree \(t''\) is formed whose cost is less than or equal to the cost of tree \(t'\). This is because \(weight(e_j) \geq weight(p)\). Thus, tree \(t''\) is a minimum cost tree. By repeating the above mentioned transformation, we can transform tree \(t'\) into tree \(t\). Thus, the tree generated by the Kruskals algorithm is a minimum spanning tree. The time complexity of Kruskals minimum spanning tree is \(O(|E|\log|E|)\)

**Prim’s spanning tree.** Unlike Kruskal’s spanning tree, in Prim’s spanning tree the subset \(s\) is always a tree and never a forest. The algorithm is given by,

\[
\text{MST-Prim}(G, w, r) \\
G: \text{Graph, } w: \text{Weights, } r: \text{Root} \\
1. \quad Q \leftarrow V[G] \\
2. \quad \text{for each } u \in Q \\
3. \quad \text{do } key[u] \leftarrow \infty \quad \text{key(v): minimum weight of any edge connecting vertex } v \text{ to the tree.} \\
4. \quad key[r] \leftarrow \infty \\
5. \quad \pi[r] \leftarrow \text{NIL } \pi[v]: \text{names of the parent of } v \text{ in tree.}
\]
6. While $Q \neq 0$

7. do $u \leftarrow \text{Extract-Min}(Q)$

8. for each $v \in Q$ and $w(u, v) < \text{key}[v]$

9. then $\pi[v] \leftarrow u$

10. $\text{key}[v] \leftarrow w(u, v)$

The time complexity of this algorithm is $O(|E| + |V| \log |V|)$.

More efficient algorithms for computing the minimum spanning trees are given in Chazelle [5] and Pettie and Ramachandran [20]. Some algorithms [10] compute the minimum spanning trees dynamically, as the vertices are added or removed dynamically.

### 2.4 The Problem of Registration

Image registration can be defined as the problem of finding the best transformation that, when applied to one image, aligns the image with the another image. Brown [4] and Pluim et al. [21] surveyed image registration methods which can be classified as the various algorithms actually used to perform registration and the applications of the registration techniques.

Based on the application of the registration techniques, the methods can be classified as, registration if images with different modalities, registration of different anatomical objects etc. The problem of registration of images with different modalities is called multimodality registration. This problem involved the registering of two images taken via different sensors or under different lighting conditions. The example of multimodality registration is registration of MRI, and CT images. In our thesis, we have registered face images which are obtained via a camera of different people, under different lighting conditions and with different expressions.

Based on the methods used for registration, the techniques are classified as based on preprocessing of images, the type of transformations applied to images,
the interpolation method used, the search strategy used, the different features of
the image used and the measure of registration.

**Mutual information for registration.** Based on the measure of registration,
several statistical criteria are used. In these criteria, majority of the methods
involve entropic measures to estimate how the images have been aligned. These
entropic measures involve estimating the entropy of the images and formulating
different measures based on them. Mutual information is one such measure, which
is defined as

\[
I(X, Y) = H(X) + H(X | Y)
\] (2.23)

\[
I(X, Y) = H(X) + H(Y) - H(X, Y).
\] (2.24)

Some methods use normalized mutual information to register the images which is
computed as,

\[
NMI(X, Y) = \frac{H(X) + H(Y)}{H(X, Y)}.
\] (2.25)

Observe that \(H(X | Y)\) is a measure of the conditional probability \(p(x | y)\) i.e. of a
particular value \(x\) in \(X\) given the value in \(Y\) is \(y\). Thus, the Equation (2.23) can be
interpreted as the measure of the uncertainty about \(X\) minus the uncertainty about
\(X\) when \(Y\) is known. It can also be stated that \(I(X, Y) = I(Y, X)\). Thus, when the
images are aligned, the amount of information the images have about each other
is maximized. As a result, registration involves finding the transformation that
maximizes the mutual information.

Another view about of information is obtained from equation (2.24), which
describes mutual information as the sum of marginal entropies minus the joint
entropy. One might argue that, as the images are transformed, the marginal
entropies of the images remain constant and the only factor that changes is the
joint entropy. From this it would seem that, maximizing the mutual information
is equivalent to minimizing the joint entropy of the two images. But this is not
true, as the entropies are calculated from the overlap areas while computing mutual information. The marginal entropies of the images thus, change with the joint entropy, as the number of pixels in the overlap area change with a given transformation. In such a scenario, if the overlap region is very small, then the joint histogram will be a sharp peak corresponding to the overlap of the one image and the background samples from second image, resulting a low joint entropy. This will give an incorrect answer, which ironically favors the separation of the two images. By adding the marginal entropies term in Equation (2.24), a penalty is added since the marginal entropies will also be low when the images are misaligned. This makes mutual information less sensitive to the problem of background.

2.5 Image Transformations

An image transformation is a function that maps the location of points in one image to a different location in an other image. The transformations can be global or local. Global transformations mean applying the same mapping for the entire image while a local transformation consists of a collection of several mapping functions, each of which is specific to a particular region in the image. Apart from these broad classes, image transformations can be classified as,

- Rigid: involves translations and rotations alone.
- Affine: involves translations, rotations, scale and shear.
- Perspective: is mainly used to map a 3D image to a 2D image by projecting the data.
- Non-Rigid: In which mapping involves introducing morphing and distortions of an image.

In this study, we have used the affine transformation and thus will elaborate more on that.

Affine transformations. Affine transformations involve rotation, scale, shear and translation. The composite transformation matrix is decomposed into scale,
rotation, shear, rotation as suggested in the paper [11] and translation. This decomposition of can be written as,

\[
T = \begin{bmatrix}
a & b & 0 \\
c & d & 0 \\
e & f & 1
\end{bmatrix}
\]

that is the composite matrix, which can be further decomposed as

\[
\begin{bmatrix}
a & b \\
c & d
\end{bmatrix} = \begin{bmatrix} 2^s & 0 \\
0 & 2^s \end{bmatrix} \begin{bmatrix} \cos(\theta) & \sin(\theta) \\
-\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} 2^t & 0 \\
0 & 2^{-t} \end{bmatrix} \begin{bmatrix} \cos(\phi) & \sin(\phi) \\
-\sin(\phi) & \cos(\phi) \end{bmatrix}
\]

where \( s \) is the scale parameter, \( t \) is the shear parameter, \( \theta \) and \( \phi \) are the rotation angles and \( e \) and \( f \) are the translations in the x and y direction.

We thus, perform our search over six degrees of freedom, two rotations, one scale, one shear and two translations. An interesting property of this affine transformation that it does not allow reflections and that it gives a nice decomposition of the composite affine matrix which allows the user to individually set different parameters.

### 2.6 Independent Component Analysis

Registration of two images can be performed by using the pixel intensity values, or by extracting features from image. Features can be extracted simply by passing the image through a set of filters, each of which detects one or more features of interest. Independent Component Analysis is one such technique for extracting features.

The motivation for the technique of Independent Component Analysis, was to solve the "cocktail party problem". This problem can be posed as follows:

---

1 The paper suggests decomposing into scale, rotation and two shears. However, it also mentions that the second shear can be replaced by a second rotation.
Suppose there are two people speaking simultaneously in a room, and we have two microphones to capture their voices. The output from the two microphones can be written mathematically as

\[ x_1(t) = a_{11}s_1 + a_{12}s_2 \]
\[ x_2(t) = a_{21}s_1 + a_{22}s_2 \]  

where \( x \) represents the output from a microphone, \( t \) is the time, \( s \) represent the sources and \( a \) represents the weights that are assigned to the signals from the microphones. ICA attempts to find the original sources of the signals (i.e. the speakers in the above case) from a mixed signal input (i.e. from microphones). ICA thus, involves recovering independent sources of signals from only the sensor observations, which are the unknown weighted linear mixtures of the signals from independent sources. Thus, ICA searches for a linear mixing matrix that, when multiplied by the original multivariate mixed data, gives statistically independent components of the data.

**Definition of independent components.** Two variables are said to be statistically independent when no variable has information about the other variable. The joint probability of two independent variables is the product of the marginal probabilities of the two variables. Thus, two random variables \( y_1 \) and \( y_2 \) are considered to be statistically independent when, \( p(y_1, y_2) = p(y_1).p(y_2) \). For \( n \) such variables, the joint density must be a product of \( n \) marginal densities of the variables. Independent variables are also uncorrelated in nature. Also, independent variables are as non-Gaussian as possible. The reason behind this is that the Gaussian distribution is symmetric in nature, which makes it impossible to find the direction of the components of the mixing matrix.

Now the obvious question arises is what are the different ways of measuring independence.
Measuring independence. Non-Gaussianity and, thus, the independence of a variable, can be measured by using an information theory metric called negative entropy, also called negentropy. Previously we stated that the differential entropy of a random variable $x$ is given by the formula,

$$h(x) = - \int p(x) \log p(x) dx.$$  \hfill (2.28)

Gaussian variables have the largest entropy among all variables of equal variance. On the other hand, the variables that have "spikes" in their probability distribution functions (less Gaussian) have less entropy. Based on this observation, negentropy is defined as

$$J(y) = H(y_{gauss}) - H(y).$$  \hfill (2.29)

Thus, the negentropy is always positive, except for gaussian variables, when negentropy is zero.

The FASTICA algorithm. We have used the FASTICA package (Available At: http://www.cis.hut.fi/projects/ica/fastica/, Accesses at: March 2004) [16] to find the ICA of images. This package implements the FASTICA algorithm which constructs a neural network, the weights of which are updated based on a learning rule. Roughly, the algorithm is as follows:

1. Negentropy is used as a measure of the non-gaussianity.
2. The cost function is an approximation of the negentropy.
3. Initially random weights are assigned at the starting point.
4. Then the value of the weights in the subsequent iteration is estimated by the Newton-Raphson method.
5. The above step is repeated until convergence.

Assuming that the images are a mixture of signals from statistically independent sources, we can use ICA to extract features from the images.
CHAPTER 3
THEORY OF MINIMUM SPANNING TREE ENTROPIES

In this chapter, we consider minimum spanning trees for registration.

3.1 Entropy Estimation for Registration

In the previous section it was shown that, mutual information requires the estimation of entropy that is calculated by estimating the probability density of the underlying data. For image registration, it might be necessary to estimate discrete instead of continuous densities. There are two broad approaches to estimate the probability density: parametric methods and non-parametric methods. Both methods require the estimation of parameters. However, non-parametric methods estimate only the width parameters like the bin size in histograms, window size in parzen window density estimation described in [9], etc, while parametric methods estimate more complicated parameters which describe the underlying density such as mean, variance etc. An overview of methods for density estimation can be found in the book by Christopher Bishop [3]. To elaborate a little bit,

- **Parametric Methods**: assume a density model for the underlying distribution of data. These methods then attempt to find the parameters of the particular model which fit optimally to the data. One of the drawbacks of this approach is the assumption of the underlying density model might not be true. The common parametric density estimation techniques are the Hidden Markov Models(HMM), Bayesian Networks etc. The HMMs model the underlying distribution by having different states and the transitions from one state to other due to the underlying probability and bias as described in [22]. Also the Bayesian Networks model a causal system in which the occurrence of a event at one node can trigger several changes in the connecting nodes as described in depth in [18]. Both the above mentioned approaches have to model complex graphs, which are defined by a large set of parameters.

- **Non-parametric Methods**: These methods make no assumption about the density model which underlies the data, and lets the data decide the nature of its distribution. Histogram methods, parzen windows are examples of
such methods. Such methods require estimation of the width parameter or smoothing factor which is the number of bins in histograms, the parzen window size \( \sigma \) which determines how well the density estimation “fits” the data.

- Semi-Parametric methods: Bishop [3] advances a class for density estimation that tries to combine the best features of parametric and non-parametric methods. Techniques such as mixture models fall in this category. These techniques assume Gaussian component densities of the data and thus have to estimate Gaussian like parameters.

Since both the parametric and semi-parametric methods assume a specific underlying distribution, they have to estimate several distribution specific parameters such as mean, variance and covariance \((\mu_a, \sigma_a, \Sigma_a)\). In contrast, non-parametric methods estimate a smoothing parameter which is usually some kind of a width parameter. Image registration requires determining the pose parameters in a particular transformation space. If parametric or semi-parametric methods are used for density estimation in registration, then the density model parameters need to be estimated apart from the pose parameters. In contrast, non-parametric methods require the estimation of width(smoothing) parameters and the pose parameters only. Thus, non-parametric density estimators are generally preferred over parametric or semi-parametric methods for image registration.

Traditional entropy estimation, either discrete or continuous, requires a priori density estimation. However, the approach taken in this study the entire density estimation. Renyi entropy is estimated directly using the samples from the image, via the relationship between the asymptotic length of the minimum spanning tree and the Renyi entropy. The Renyi entropy and its properties are discussed in Section (2.1.4). The following Section discusses the chronology of the development of the theory underlying the entropy estimation by a minimum spanning tree.

3.2 History

In the late 50’s, an interesting viewpoint for estimating entropy using graph algorithms came into existence. It began in 1959 when Beardwood, Halton and
Hammersley [2] first analyzed the travelling salesman problem (TSP) and stated that the TSP length of a set of multi-dimensional feature vector is related to the entropy. Later, in 1988 J. M. Steele [25] followed their work and stated that if a fully connected graph is constructed from a multi-dimensional feature space such that the weight of an edge between any two points is the weighted Euclidean distance between the points, then the asymptotic length of a minimum spanning tree constructed from such a graph is related to the underlying probability density.

Later, the group at University of Michigan led by Dr. Alfred Hero showed how the asymptotic length of the minimum spanning tree is directly proportional to the Renyi entropy (refer Section 2.1.4) [17]. They further pointed out how this relationship between d and minimum spanning trees can be further expanded to a variety of graphs like k-nearest neighbors or Steiner Trees [13]. The practical applications of such graphs are emphasized in [12] with image registration being one such application.

Minimum spanning trees and Renyi entropy were initially discussed in Section (2.3) and (2.1.4) respectively. Some relevant portions are reiterated here as and when needed. Suppose there are n independent multivariate points in a d dimensional space, \(x_i \in R^d\) and that there is a graph of these points with \(V = x_1, x_2, \ldots, x_n\) vertices and \(E = (x_i, x_j); 1 \leq i < j \leq n\) where each edge e in the edge set E is denoted by a weight function, \(\psi(|e|)\). The task of finding a minimum spanning tree involves finding the tree with the minimum weight among all the trees which span the all the vertices in the set V

\[
M(x_1, x_2, \ldots, x_n) = \min_T \sum_{e \in T} \psi(|e|),
\]

(3.1)

where \(T\) is a set of all the graphs with vertex set that includes all points in the space \(R^d\), \(\psi\) is the weighting function that associates weights with the edges of the
minimum spanning trees which can be defined as

$$\psi(x) = x^\gamma \text{ where } 0 < \gamma < d.$$  (3.2)

Let the points in feature space $R^d, d \geq 2$ have a independent random
distribution $\mu$ with the $\psi$ function satisfying the criteria $\psi(x) \sim x^\gamma$ and $0 < \gamma < d,$
and let $f$ be the density of the continuous part of $\mu$ with $c(\gamma, d)$ a positive constant
depending upon $\gamma$ and $d,$ then Steele [25] proved the relationship between the
length of a minimum spanning tree and some power of the density function $f$ as,

$$\lim_{n \to \infty} n^{-d-\gamma} M(X_1, X_2, \ldots, X_n) = c(\gamma, d) \int_{R^d} f(x)^{d-\gamma} dx.$$  (3.3)

The next section shows how the equation (3.3), after a little bit of algebraic
manipulations, presents a clear link between the minimum spanning tree and the
Renyi entropy.

### 3.3 Minimum Spanning Trees for Entropy Estimation

We discussed about alpha Renyi Entropy or simply the Renyi entropy in
Section (2.1.4) and represented it mathematically in Equation (2.18) as,

$$h_\alpha(X) = \frac{1}{1 - \alpha} \log \left[ \int f^\alpha(x) dx \right].$$  (3.4)

The relationship between the density function and the length of the minimum
spanning tree is specified in Equation (3.3). If we rename the length of the
Minimum Spanning Tree in Equation (3.3) as,

$$M(X_1, X_2, \ldots, X_n) = L(X_n) = \min_{e \in T} \sum_{e} e_{ij}^\gamma,$$  (3.5)

where $e_{ij} = \|x_i - x_j\|$ denotes the Euclidean distance between points $i$ and $j$ in a $d$
dimensional feature space, we can rewrite the Equation (3.3) as,

$$\lim_{n \to \infty} n^{-(d-\gamma)/d} L(X_n) = c(\gamma, d) \int_{R^d} f(x)^{d-\gamma} dx$$  (3.6)
Now taking the log on both sides and substituting $\alpha = (d - \gamma)/d$ we obtain,

$$\log \lim_{n \to \infty} n^{-\alpha} L(X_n) = \log(c(\alpha, d)) + \log(\int_{\mathbb{R}^d} f(x)^{\alpha} dx). \quad (3.7)$$

Note that the constant $c$ which was a function of $\gamma$ and $d$, now becomes a function of $\alpha$ and $d$ due to the substitution of $\alpha$. Dividing the above equation by $1 - \alpha$ on each side yields,

$$\frac{1}{1 - \alpha} \log(\int_{\mathbb{R}^d} f(x)^{\alpha} dx) = \frac{1}{1 - \alpha} (\log \lim_{n \to \infty} n^{-\alpha} L(X_n) - \log(c(\alpha, d))) \quad (3.8)$$

From the Equation (2.18), the left hand side of the preceding equation can be reduced to

$$h_\alpha(x) = \frac{1}{1 - \alpha} (\log \lim_{n \to \infty} n^{-\alpha} L(X_n) - \log(c(\alpha, d))). \quad (3.9)$$

The short derivation above points as to how the minimum spanning tree length converges to the Renyi entropy. The above Equation (3.9) is the fundamental result from which we estimate the entropy approximately in the thesis. It is the approximate estimation of the entropy as the constant $c$ (which depends on $\alpha$ and the dimensions $d$) is unknown and cannot be computed analytically. The strength of Equation (3.9) lies in the fact that the length of the minimum spanning trees can be easily computed in polynomial time specified by $O(|E| \log |E|)$ (refer [7]).

Observe that Equation (3.9) is a very clean equation which doesn’t require the estimation of any smoothing parameter which has to be chosen carefully e.g., the number of bins in the histogram or the ”sigma” in the parzen window estimation of entropy. Of course we have to compute the length of the minimum spanning tree which is trivial. For the number of bins in the histogram and the parzen window width estimator, on the other hand, there is no explicit criteria of selection of the smoothing(width) parameter.

The explanation of the rate of convergence of MST entropy follows. The rate of convergence of the algorithm depends mostly on the number of samples that we
choose to compute the MST. A very nice paper by Alfred Hero et. al ("Convergence rates of minimal graphs with random vertices" by A. O. Hero, J. A. Costa and B. Ma” submitted to ”IEEE Transactions on Information Theory” in March 2003) formulated the rates of convergence of the minimum spanning tree for piece-wise linear densities and then extended their proposition for a more general form of densities. We next discuss a proposition for the rate of convergence of minimum spanning trees. We assume that the number of dimensions $d \geq 2$, that $1 \leq \gamma \leq d-1$ and $X_1, X_2, \ldots, X_n$ are independent sample points in the space $[0, 1]^d$ with density $f$ with support $S \subseteq [0, 1]^d$

**Proposition: Marginal block density.** The density $f$ has $m^d$ levels. For the quasi-additive Euclidean functional $L_\gamma$ of order $\gamma$, the following bound exists,

$$| E[L_\gamma(X_1, X_2, \ldots, X_n)]/n^{(d-\gamma)/d} - c(\gamma, d) \int_S f^{(d-\gamma)/d}(x)dx | \leq O((nm^{-d})^{-1/d}).$$

(3.10)

In the Equation (3.10), the term $L_\gamma$ denotes the length of the minimum spanning tree whose edges have the weight of the Euclidean distance between the points raised to the power $\gamma$ as described in (3.5).

The Equation (3.10) means that the difference between the expected value of the length of the minimum spanning tree normalized by $n^{d-\gamma/d}$ and the product of the constant $c$ and the integral of the density function is asymptotically less than $(nm^{-d})^{-1/d}$. For example, if $d = 2$, then the upper bound on the difference mentioned above which is the error is $O((nm^{-2})^{-1/2})$. That is as number of samples $n$ increases, then the bound becomes tighter when we can say that the error reduces as the number of samples increase. Observe from equation (3.10) that, as the dimensionality of the feature space increases, the rate of convergence decreases and thus will need more samples for convergence.

The preceding proposition is important in the context of image registration, as the images have fixed levels of color values and thus can be said equivalent to
the piece-wise density considered above where there are a total of \( m^d \) levels in the 
\( d \) dimensional space. Interested readers should refer to the paper, ("Convergence rates of minimal graphs with random vertices" by "A. O. Hero, J. A. Costa and B. Ma" submitted to "IEEE Transactions on Information Theory" in March 2003) for a generalization of the above result to the generalized class of densities.

The paper by Hero et. al. ("Convergence rates of minimal graphs with random vertices" by "A. O. Hero, J. A. Costa and B. Ma" submitted to "IEEE Transactions on Information Theory" in March 2003) mentions that the aforementioned convergence criteria can also be applied to the graph constructions like the Steiner tree graph, minimal matching graph or the TSP tour all along with their power weighted variants. However, the minimum spanning tree is chosen based on the paper [12] which recommends the MST as the best method to estimate entropy due to its polynomial runtime. There exist many well known algorithms for implementing the MST which supports use of the minimum spanning tree as the density estimator.

3.4 Measure of Registration

Since Viola and Wells [28] and Collignon et. al. [6] successfully used the Shannon mutual information to perform registration, it has been used as a measure to perform registration in many systems. In this study, the minimum spanning tree as described in the above section are used to estimate the approximate Renyi entropy of the random variables. This entropy is further used to compute the measure of registration.

The definition of the Renyi mutual information was first formulated by Alfred Renyi in the book [23] as,
\[
I_\alpha(x) = \frac{1}{1-\alpha} \log \int_{-\infty}^{\infty} \frac{f_X(x)^\alpha}{\prod_{i=1}^{N} f_{X_i}(x_i)^{\alpha-1}} \, dx. \tag{3.11}
\]
Unfortunately, Renyi mutual information cannot be easily computed. Thus instead of the Renyi mutual information, we use a similar measure which is simply the sum of the marginal entropies minus the joint entropy similar to the Shannon mutual information. This measure was proposed by Hild et al in the paper [14] as,

$$\sum_{i=1}^{N} H_\alpha(x_i) - H_\alpha(x) = \frac{1}{1 - \alpha} \log \frac{\int_{-\infty}^{\infty} f_X(x)^\alpha}{\int_{-\infty}^{\infty} \prod_{i=1}^{N} f_X(x_i)^\alpha} dx_i.$$  (3.12)

In case of two images, the above measure is the sum of the two Renyi marginal entropies minus the Renyi joint entropy as,

$$H_\alpha(x_1) + H_\alpha(x_1) - H_\alpha(x) = \frac{1}{1 - \alpha} \left( \log \frac{L_{x_1}}{n^\alpha} + \log \frac{L_{x_2}}{n^\alpha} - \log \frac{L_{x_{12}}}{n^\alpha} \right)$$  (3.13)

where $L_x$ denotes the length of a minimum spanning tree constructed from a power weighted graph computed as $L_x = \min_T \sum_{e \in T} e_x^\gamma$.

The preceding measure is very similar to the Renyi mutual information as both are positive and both tend to zero when the joint probability is equal to the product of the marginal probabilities (when the variables are independent as mentioned in the paper [14]). However, the main reason why the measure is preferred over the actual entropy is that it can be calculated using the MST entropy algorithm. Thus, our algorithm maximizes this measure to perform registration.

Also, maximizing the mutual information is similar to minimizing the joint entropy of the images when the search space is narrow. This is so as the marginal entropies remain more or less constant in a narrow search space. Referring to the Equation (3.12), we see that, in a narrow space as the marginal entropies are more or less constant, the negative of joint entropy is maximized. In other words, joint entropy is minimized. In some cases (especially when we results are based on one parameter like rotation) we have opted to minimize joint entropy instead of maximizing the mutual information.
CHAPTER 4
ALGORITHM DESIGN

This chapter first describes the overall design of a generalized registration technique followed by the various design decisions while designing this algorithm, a brief algorithm description and then implementation details.

4.1 General Registration Algorithm

The process of image registration can be represented procedurally as shown in Figure 4–1. The first step preprocesses the images such that the images become appropriate for registration. Preprocessing involves performing operations on image like noise removal, smoothing of the image, cropping etc. Human Intervention can occur depending on the type of images considered for registration. The next step is a quick estimation the exact transformation parameters by scanning the search space in low resolution. And finally, when we have the coarse parameters for the transformation, the region around the rough parameters is scanned in order to get an accurate estimate. This multi-resolution helps as it speed up the registration process. Alternatively, if the time requirements for estimating the measure( that we discussed in Section 3.4) are less stringent, then the initial rough estimation step is omitted all together and the search space is searched at fine resolution. The flow

![Flowchart](image)

Figure 4–1: Block diagram for image registration
We apply the transformation on only one of the two images (transforming image) and the other image is unchanged (reference image). At each step of the algorithm, the entropic measure between the two images is calculated. If the measure has reached the maximum (or minimum depending on its nature) in the space, then the parameters of the transform are declared as the optimal parameters. If not, then the next transform to be applied to the transforming image is computed and that transform is applied to the transforming image and the

Figure 4–2: Flow-chart for the estimating the parameters of the transform
steps are repeated until the optimum value of the measure is found in the search space.

4.2 Design of the Algorithm

Design of a registration algorithm involves a sequence of decisions that need to be taken at each step as discussed in this Section. As now what becomes obvious from the previous discussion that crux of this algorithm is estimation of the Renyi entropy which is continuous. This estimation can be done by using the non-parametric Renyi measure based on the the minimum spanning trees based on the equation (3.8) which is expressed as

$$\frac{1}{1 - \alpha} \log(\int_{\mathbb{R}^d} f(x)^\alpha dx) = \frac{1}{1 - \alpha} (\log \lim_{n \to \infty} n^{-\alpha} L(X_n) - \log(c(d - d\alpha, d))).$$  

From this equation, the estimation of entropy is approximate because of the presence of the constant term $c$ that only depends on $\alpha$ and the dimensions $d$.

The first step therefore is estimating parameters by performing rough image registration using histograms.

4.2.1 Preprocessing by Histograms

The main reason for using the histograms is that they are computationally efficient. This allows us to get a ballpark estimation very quickly. The entropies (both the joint and marginal) are estimated from the overlap area of the two images. The reason for estimating the joint entropy from the overlap area is obvious. However we also estimate the marginal entropy from the overlap area as as the images are transformed, the marginal entropy of the overlap area also changes dynamically with every transformation.

The important parameter which decides how well the histogram performs is the bin size, which is chosen based on statistical analysis. The Figure 4–3 elaborates empirical results of histogram analysis when the number of bins are varied.
Figure 4-3: Effect on registration by varying the number of bins of histogram (Number of Bins are 8, 16 and 32 from top left to bottom right. Maximum should be present at 0.)
As we can see as the number of bins decrease (bin size is large) the histogram is noisy and there are many peaks around the desired value maximum in the histogram. However, if the number of bins increase, the peak disappears as the data is over fitted.

After histogram analysis, we extract features from the image and use these feature images to perform registration as follows,

### 4.2.2 Features

Image features are extracted based on a local neighborhood of each pixel, called tags [1]. The basic of idea of tag-based methods is to account for the local topology surrounding each pixel. Around each pixel in the image, a $n \times n$ matrix is extracted and vectorized, to obtain a vector of length $n^2$ for every pixel. This is equivalent to quantizing the image such that it reflects local topology due to which the extracted features become more invariant to change in the individual pixel intensity.

Once the tags are extracted from the image, the decision that laid ahead was how to extract features. We opted for statistical based features since they extract features based on the information content of the image. As mentioned in Section (2.6), the ICA algorithm attempts to find statistically independent components of the image. We have used the FASTICA [16] algorithm to compute the ICA of the images.

The Figure 4–4 shows the feature images as computed by ICA. The actual values of the ICA components can be seen from the table 4–1.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.2845</td>
<td>0.2875</td>
<td>-0.1523</td>
<td>0.1468</td>
</tr>
<tr>
<td>0.3425</td>
<td>-0.5310</td>
<td>-0.4016</td>
<td>0.5821</td>
</tr>
<tr>
<td>0.4048</td>
<td>0.0869</td>
<td>-0.4269</td>
<td>-0.0605</td>
</tr>
<tr>
<td>0.0597</td>
<td>0.0197</td>
<td>0.0251</td>
<td>0.0777</td>
</tr>
</tbody>
</table>

Table 4–1: The ICA values
Figure 4–4: ICA feature images
of the table 4–1 are responsible for the edge images of the original image while the fourth ICA component simply smooths the image.

One point here is how many ICA features to extract or how much should be the dimensionality of the feature space. There are several interesting observations in this case. Figure 4–6 represents the mutual information vs. the rotation angle for various sizes of ICA blocks. The images for registration are shown in Figure 4–5. From the Figure 4–6 we can see that the registration worsens if we pick too high a feature dimension.
Another interesting observation is seen when we pick the faces shown in figure 4–30 we get the MI Vs. rotation plots as shown in Figure 4–7. From the Figure 4–7, we see that the acceptable registration result is obtained when we use 16 ICA features. Thus the number of ICA features to use relies heavily on the type of image to register.

Once features are extracted from the image, the next step is to fine tune the registration by estimating the entropy using the minimum spanning trees.

4.2.3 Search Strategy

Our prototype MST matching algorithm uses a brute force search that scans the entire transformation space to perform registration. One of the reasons for not doing this is the way the MST entropies behave in the search space. The behavior is shown in 4–8, from which we can see that the MST entropy is not smooth and has several local maxima. Thus, normal search strategies like the gradient descent etc cannot be used as they rely on a smooth objective function.
Figure 4–8: Behavior of MST entropy in 1D and 2D search space.
Thus, in order to be not overwhelmed by the design of an efficient search algorithm for finding the global maxima for a non-smooth objective function, we used a brute force search which is guaranteed to find a global maximum.

4.2.4 Minimum Spanning Tree Algorithm

This is the final step of the registration. Ideally, we would use the entire overlap area of the two images to compute the minimum spanning tree and then estimate the entropy. From Equation (3.8), we see that the entropy is sensitive to the number of samples used to compute the minimum spanning tree and thus we need to maintain this number constant during each computation. So, we choose a fixed number of sample points randomly from the overlap area of the image and use only those to estimate the entropy.

Minimum Spanning Tree computation. Assume that we pick \( n \) sample points each time to compute the minimum spanning tree. Now here we have the points in some \( d \) dimensional space, where the weights of the edges between two points are the Euclidean distance between the points. Also, initially we assume that there is an edge between a point and all the other \((n - 1)\) points in the space i.e. that the graph is fully connected.

In such a scenario, we have \( n \) points and \( n^2 \) edges in the Euclidean space, the classical MST algorithm will have the complexity of order \( O(n^2 \log n) \). As this is a polynomial complexity algorithm, the search time increases tremendously as \( n \) increases. One of the papers [19] mentions about resolving this issue by reducing the number of edges. This has been done by placing a sphere around each point in the \( d \) dimensional hyperspace. Only the edges connecting the given point to its neighbors that lie within the sphere are retained for analysis. The radius of the sphere is calculated based on the following formula,

\[
\text{radius}_p = \text{mean}_p + \frac{\max - \text{mean}_p}{\text{QFactor}} \times \text{RFactor} 
\] (4.2)
where

radius\(_p\) denotes the radius at point \(p\).

mean\(_p\) denotes the mean of distances that connect point \(p\) to all other points.

max denotes the maximum distance between any 2 points in the d dimensional space.

QFactor denotes the quantization factor.

RFactor denotes the factor which decides magnitude of the radius.

We varied the RFactor it over a specific values of the radii and chose the one which gave the radius with minimum edges and error. The plots are shown in 4–9.

As a result, every point has a sphere of a different radius, which is determined primarily by how close the point is to the other points in the hyperspace.

Deciding the number of samples. The Figures 4–10 and 4–11 illustrate that when the value of \(\alpha\) is kept about 0.55, then 500 samples seem sufficient for registration. However a nice plot with explicit maximum is obtained at about 1100 samples. A similar plot for value of alpha as 0.85 is shown in 4–12 and 4–13 which illustrates that a very good result is obtained at 1100 samples.

Thus evidently as more samples are better for accuracy. Fewer samples are picked when speed desired.

The minimum spanning tree was computed by using the Kruskal’s Algorithm implementation provided by the Boost Graph Library (Available at: http://www.boost.org/libs/graph/doc/table_of_contents.html, Accessed on: March 2004). The pseudo code for this implementation can be found in the APPENDIX section.

4.3 Implementation Details

4.3.1 Hardware and software Details

- CPU: Pentium IV, 2.8 GHz, 2 GB RAM.
- OS: Linux.
- Languages: Matlab and C++.
All the code was written in Matlab. The program used the Kruskal Algorithm provided by the Boost Graph Library (Available at: http://www.boost.org/libs/graph/doc/table_of_contents.html, Accessed on: March 2004) which is written in C++. The interface between the Matlab and the C++ code is done using the Matlab MEX Function Interface.

Figure 4-9: Plots showing error between the actual MST length and the truncated MST length over various radius factors and over rotations of -10, 0 and 10 degrees
Figure 4–10: Figure showing the variations in joint entropy verses the rotation angle with varying number of samples picked from the overlap area when alpha is 0.55 are shown as 1. 100 Samples 2. 300 samples 3. 500 samples 4. 700 samples

Figure 4–11: Figure showing the variations in joint entropy verses the rotation angle with varying number of samples picked from the overlap area when alpha is 0.55 are shown as 1. 900 Samples 2. 1100 samples
Figure 4–12: Figure showing the variations in joint entropy verses the rotation angle with varying number of samples picked from the overlap area when alpha is 0.85 are shown as 1. 100 Samples 2. 300 samples 3. 500 samples 4. 700 samples

Figure 4–13: Figure showing the variations in joint entropy verses the rotation angle with varying number of samples picked from the overlap area when alpha is 0.85 are shown as 1. 900 Samples 2. 1100 samples
Table 4–2: Free parameter table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
<th>Value</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>$\alpha$ in Renyi Entropy</td>
<td>0.5</td>
<td>The value suggested by the Hero et. al in paper [19].</td>
</tr>
<tr>
<td>BlockSize</td>
<td>Size of block around each pixel i.e. the number of ICA features</td>
<td>2 or 4</td>
<td>Based on analysis in section 4.2.2, figures 4–6, 4–7.</td>
</tr>
<tr>
<td>radius</td>
<td>radius of the sphere around each point in feature space to truncate the edges</td>
<td>5</td>
<td>Based on analysis shown in Figure 4–9</td>
</tr>
<tr>
<td>numBins</td>
<td>number of bins for histogram search</td>
<td>16</td>
<td>Based on analysis shown in Figure 4–3</td>
</tr>
<tr>
<td>numberSamples</td>
<td>Number of random samples used to compute Renyi entropy</td>
<td>500 or 1000</td>
<td>Picked based on whether speed or accuracy is desired. Figure 4.2.4</td>
</tr>
<tr>
<td>searchSpace</td>
<td>Affine parameters over which to search</td>
<td>Rotations[-20,20] , Scale[-0.5,0.5] , shear[-0.5,0.5] , translations[-5,5] in each direction</td>
<td>Based on the time required to cover the space</td>
</tr>
</tbody>
</table>

- **Time:** The time taken for the entire algorithm to run for a 64*64 images with histogram preprocessing and then final registration using MST entropy using 1000 samples is approximately 8 hours.

4.3.2 Free Parameters

The free parameters are tabulated in Table 4–2

4.4 Results

This section summarizes the different results obtained from the registration algorithm. This section shows that the registration by estimating entropy by using the MST produces acceptable results. All the face images are obtained from the Yale face database (Available At:
4.4.1 Same Image Registered with the Same Image

This result shows how the MST changes as the images are registered. As both the images considered are the same image in this case, the visual representation of the change in the MST from the unregistered case to one in the perfectly registered case is very apparent.

The parameters values for the following tests:

\( \alpha = 0.5 \)

Samples = 500

Number of bins of Histogram = 16

Block size around pixel = 2 x 2

The figure 4-14 displays the original unregistered image and their overlap images.

The Figure 4-15 shows the minimum spanning tree of the two unregistered images. The feature space in this representation is the intensity of both the images for the purpose of visualization.

The figure 4-16 shows the how the images are registered.

The MST of the two registered images is shown in figure 4-17. Observe how nicely the points collapse in one line as the images become registered.

4.4.2 Registration of Same Person with Different Expressions

In this section, here are the registration results for the images of the same person in different facial expressions. The parameters chosen are:

\( \alpha = 0.5 \)

Samples = 500

Number of bins of Histogram = 16

Block size around pixel = 2*2
Figure 4-14: Figure showing the from top left to bottom right 1. Face image 2. Same face with applied transformation 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images

Figure 4-15: Minimum spanning tree of the unregistered images considering only the intensities of the two images as features
Figure 4–16: Figure showing the from top left to bottom right 1. Face image 2. Face obtained after applying the optimal transformation to the unregistered image shown in the figure 4–14 3. Overlap of the two registered images 4. Edge image of the overlapped registered images.

Figure 4–17: Minimum spanning tree of the registered images considering only the intensities of the two images as features.
The images before registration are shown in Figure 4–18 and the MST for the unregistered case is in Figure 4–19.

After performing registration the results are shown in Figure 4–20 and the MST for the registered case is in Figure 4–21.

If we see at the images of the minimum spanning trees then we see that when the two images are unregistered then the points are scattered all over the place. However when the images are registered, a cluster of points is formed at the 45 degree line in the MST space reducing the length of the MST.

4.4.3 Yet Another Registration of Same Person with Different Expressions

In this section, we show the registration results for another images of the same person in different expressions. The parameters chosen are:

\[ \alpha = 0.5 \]

Samples = 500
Figure 4–19: Minimum spanning tree of the unregistered images considering only the intensities of the two images as features

Figure 4–20: Figure showing the from top left to bottom right 1. Face image 2. Face image of the same person with glasses which is registered 3. Overlap of the two registered images 4. Edge image of the overlapped registered images
Figure 4–21: Minimum spanning tree of the registered images considering only the intensities of the two images as features

Number of bins of Histogram = 16
Block size around pixel = 2 x 2

The images before registration are shown in Figure 4–22 After performing registration the results are shown in Figure 4–23

From the previous two results with the images of same person and different expressions, we can see that the algorithm successfully performs registration. An important point to note here is that the image size was 64 x 49 (3136 pixels) out of which 500 were used to estimate entropy in order to perform registration by using MST unlike histograms which need all the samples in the area in order to estimate entropy. Plots of convergence for the Renyi $\alpha$ entropy verses the number of samples are shown later in this section.

4.4.4 Registration Result of Images of two Different People

We next discuss registration results for the images of the different persons. The parameters chosen are:

$\alpha = 0.5$
Figure 4-22: Figure showing the from top left to bottom right 1. Face image of a happy person 2. Face image of the same person with sad expression which is unregistered 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images

Figure 4-23: Figure showing the from top left to bottom right 1. Face image 2. Face image of the same person with sad expressions which is registered 3. Overlap of the two registered images 4. Edge image of the overlapped registered images
Samples = 500
Number of bins of Histogram = 16
Block size around pixel = 4x4
Initial state of the unregistered images is shown in Figure 4-24
After performing registration the results are shown in Figure 4-25
The above result shows that though it is doing registering the two images, the result is not accurate for dissimilar images and there is definitely scope for improvement. In the next result therefore I changed some parameters and got some improvements in the result.

4.4.5 Yet Another Registration Result of Images of Two Different People

Now are the registration results for the yet another images of the different persons. The parameters chosen are:

\[ \alpha = 0.9999 \]

Samples = 1000
Figure 4-25: Figure showing the from top left to bottom right 1. Face image 2. Face image of different person which is registered 3. Overlap of the two registered images 4. Edge image of the overlapped registered images

Number of bins of Histogram = 16
Block size around pixel = 4*4

Also I scaled the search range for scale to perform even finer search.

Initial state of the unregistered images is shown in Figure 4-26

After performing registration the results are shown in Figure 4-27

The above result is a trifle better than the previous one.

4.4.6 Comparison with Histogram Based Method

In order to do this comparison, we selected two sets of images of two different people. In one case, we perform registration over six parameters for the faces considered in section 4.4.5 and we registered the faces over all the six parameters namely rotation, scale, shear and translations.

Initial state of the unregistered images is shown in Figure 4-28

After performing registration the results are shown in Figure 4-29
Figure 4-26: Figure showing the from top left to bottom right 1. A Face image 2. Face image of the different person 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images

Figure 4-27: Figure showing the from top left to bottom right 1. Face image 2. Face image of different person which is registered 3. Overlap of the two registered images 4. Edge image of the overlapped registered images
Figure 4-28: Figure showing the from top left to bottom right 1. A Face image 2. Face image of the different person 3. Overlap of the two unregistered images 4. Edge image of the overlapped unregistered images

Figure 4-29: Figure showing the from top left to bottom right 1. Face image 2. Face image of different person which is registered 3. Overlap of the two images registered using histogram 4. Edge image of the overlapped registered images
We see from the figure 4-29 that the histogram method performs as well as the MST when the intensity differences in the two images are not that obvious.

Later we performed the comparison on two images in which the lighting on one of the images was from the left side while in the other image the light was from the right side and then compared results over the rotation only. The images that were considered are shown in figure 4-30

The second image is rotated by 5 degrees and perfect registration finds the right answer at -5 degrees. The results with the histogram are shown in 4-31

The plot of the MI vs the rotation is shown in 4-32. Notice how the maximum MI is seen at 9 degrees instead of -5 degrees, which is a clear error.

The results with the MST are shown in Figure 4-33

The plot of the MI vs the rotation is shown in Figure 4-34. Notice how the maximum MI is seen at -5 degrees as expected.
Figure 4-32: Plot of MI vs. rotation angle when registered by histogram. Maximum should be at -5 degrees.

Figure 4-33: Figure showing image 2 as registered by the MST

Figure 4-34: Plot of MI vs. rotation angle as registered by MST. Correct answer is at -5 degrees.
Thus we conclude, that in the images with lot of intensity variations, the registration by using a minimum spanning tree definitely outperforms that by the histogram.

4.4.7 Convergence of Renyi Entropy with Number of Samples as Alpha Changes

Recall from the previous discussion in the background and theory section that we need to estimate the $\alpha$ Renyi entropy that as $\alpha \rightarrow 1$ converges to the Shannon differential entropy. The question here is what value of $\alpha$ to choose such that the Renyi entropy estimated converges quickly in fewer samples.

This result documents the plots of the Renyi entropy computed using the minimum spanning trees vs. the number of samples as we vary the $\alpha$. The result is shown in Figures 4–35, 4–36, 4–37.
Figure 4-36: Figure showing the from top left to bottom right, MST Renyi joint entropy vs the samples for 1. alpha = 0.6 2. alpha = 0.65 3. alpha = 0.7 4. alpha = 0.75
Figure 4–37: Figure showing the from top left to bottom right, MST Renyi joint entropy vs the samples for 1. alpha = 0.8 2. alpha = 0.85 3. alpha = 0.9 4. alpha = 0.95
Table 4-3: Noise results over rotation for histogram

<table>
<thead>
<tr>
<th>Noise Variance</th>
<th>Rotation Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0</td>
</tr>
<tr>
<td>0.02</td>
<td>0</td>
</tr>
<tr>
<td>0.03</td>
<td>0</td>
</tr>
<tr>
<td>0.04</td>
<td>0</td>
</tr>
<tr>
<td>0.05</td>
<td>1</td>
</tr>
<tr>
<td>0.06</td>
<td>1</td>
</tr>
<tr>
<td>0.07</td>
<td>1</td>
</tr>
<tr>
<td>0.08</td>
<td>1</td>
</tr>
<tr>
<td>0.09</td>
<td>1</td>
</tr>
<tr>
<td>0.1</td>
<td>1</td>
</tr>
</tbody>
</table>

The graphs in Figures 4-35, 4-36 and 4-37, show that the behavior of the Renyi entropy is very erratic for all the values of alpha and in fact does not seem to converge quickly.

4.4.8 Noise Trials

Noise in the image may affect the registration results. This section compare the histogram technique to the MST technique when the Gaussian noise with varying variance was added to the images. The images were then registered over rotation and over rotation and scale and the error by both the techniques was compared.

- **case 1: Registration over rotation only.**

  Range of rotation: -10 to 10 degrees

  Correct Answer at: 0 degrees.

  The error for the histogram method for various noise added can be tabulated as in Table 4-3. As seen in Table 4-3, the histogram performs well for low noise but gives an error of 1 degree for a noise variance greater than 0.05.

  The error for the MST method for noise added can be seen in Table 4-4. From this table, we can see that the MST performs well for a low noise case and again
Table 4–4: Noise results over rotation for MST

<table>
<thead>
<tr>
<th>Noise Variance</th>
<th>Rotation Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0</td>
</tr>
<tr>
<td>0.02</td>
<td>0</td>
</tr>
<tr>
<td>0.03</td>
<td>0</td>
</tr>
<tr>
<td>0.04</td>
<td>0</td>
</tr>
<tr>
<td>0.05</td>
<td>0</td>
</tr>
<tr>
<td>0.06</td>
<td>3</td>
</tr>
<tr>
<td>0.07</td>
<td>0</td>
</tr>
<tr>
<td>0.08</td>
<td>3</td>
</tr>
<tr>
<td>0.09</td>
<td>0</td>
</tr>
<tr>
<td>0.1</td>
<td>2</td>
</tr>
</tbody>
</table>

introduces error (in fact larger) than the histogram as the variance of the noise added increases.

Case 2: Registration over rotation and scale.

Rotation Range: -10 to 10 degrees.

Scale Range: -0.5 to 0.5

The correct answer expected for both rotation and scale is at 0.

Initially, some trials were made by just adding noise to the images and then registering the raw images. With this option however the MST approach performed badly and gave an error for values low noise cases with variance of 0.01. The histogram method clearly outshone the MST method in this case. So later we used a smoothing filter to smoothen out the images before registering then with the MST and the histogram.

When smoothing Weiner filter was applied to the images after adding noise, the histogram method gave a ZERO error consistently for variance from 0.01 to 0.08.

Results when the MST was used are tabulated in the table 4–5. This table shows how the MST error worsens dramatically in a two parameter search space.
Table 4-5: Noise results over rotation and scale for MST

<table>
<thead>
<tr>
<th>Noise Variance</th>
<th>Rotation Error</th>
<th>Scale Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.03</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0.04</td>
<td>4</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4-6: Noise results over rotation and scale for MST

<table>
<thead>
<tr>
<th>Noise Variance</th>
<th>Rotation Error</th>
<th>Scale Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.02</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.03</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0.04</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0.05</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>0.06</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.07</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>0.08</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

In this above result shown in Table 4-5, 1000 samples from the overlap area of the images were used to compute the MST length. Another set of results when 1500 samples were used to compute the MST are tabulated in the Table 4-6. From the Table 4-6 we can see that the accuracy of registration by MST decreases progressively as an error is consistently introduced.

This result illustrates the behavior of the MST entropy when Gaussian noise was added to the image and then the image was smoothed with a particular filter before registering. The are different types of noise and filters that can be used to remove the noise. This relationship between accuracy of registration by MST and the different types of noise and filters needs to be studied further.
CHAPTER 5
DISCUSSION

The estimation of entropy has its applications in a variety of areas with image registration being one such application. Recently, information theoretic measures such as mutual information have become very popular in image registration. Estimating the mutual information requires the estimation of entropy.

This thesis used minimum spanning tree algorithms to estimate the entropy from a finite set of samples. The MST can be computed easily in polynomial time. One of the good attributes of the MST-based entropy estimation is that no further parameter tuning is required.

However when applying these entropies to perform registration several important issues need to be dealt with. The MST entropy does not approximate the Shannon entropy and instead approximates the Renyi entropy seen in equation (3.3). Thus it cannot estimate the Shannon mutual information which is widely used in registration. In fact it cannot even estimate the original Renyi mutual information that we saw in (3.11). Instead we have used a information theoretic measure which is the difference between the sum of the marginal entropies and the joint entropy as was suggested in paper [14]. This measure has properties similar to that of the Renyi mutual information.

There are several open issues that need to be resolved in this case. We would like to undertake a systematic comparison with other established density and entropy estimators such as Parzen windows [9]. Parzen windows estimate both the Shannon entropy as well as the Renyi entropy. Thus they can be also used to estimate the Shannon mutual information. The density function by the Parzen
windows can be computed from the following equation,

\[ f_n(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left(-\frac{||x - x_i||^2}{2\sigma^2}\right). \] (5.1)

This equation can be further used to estimate the \(\alpha\) Renyi Entropy which is mentioned in the equation (2.18). Note that when using the Parzen Window density estimator, it can compute the entropy for \(\alpha > 1\) whereas for the MST the condition on the alpha is that it should be less than 1. Thus careful formulation has to be done of all the parameters in order to do a comparison between these density estimators.

Apart from this, even in the regime of MST entropies that are used for registration, several important changes can be made to the method proposed in this thesis. One of them is that, in this thesis we have simply scanned the entire space to find the maximum (Refer back to the section 4.2.3). Clearly, a good optimized search algorithm to search over such a space would be beneficial in cutting the time required drastically.

Also, one area of improvement is in feature selection. Right now, we have used ICA features based on the rationale that ICA returns statistically independent features. But a detailed study on the features can help improve the registration results. There are several other methods of feature selection like the principal component analysis, use of image processing filters like Gabor filters etc. [27]. However, the noise analysis that we did in the section (4.4.8), suggests that when noise is present it requires smoothing of images as these methods rely heavily on taking the derivatives of the images. Thus presence of noise can in fact have a negative effect on the registration when derivative-based features are used, vis-a-vis intensity-based registration.

An interesting extension to the registration by using MST entropies can be pursued if instead of using Euclidean distance among the feature points, we project
the points in a different space (e.g., Hilbert space) and then use the distance in the Hilbert space to compute the MST length. Whether such a extension improves the registration is another open area of research.

In fact we also noticed in section (4.4.8) that the histogram method is less sensitive to noise. Thus a way to register by using a high dimensional histogram is also a possibility. However, in order to do that innovative ways have to be devised to compute histograms in a multi dimensional space.

In conclusion, in this thesis, we use the MST entropies to register just two face images at a time searching over the space of affine transformations. This technique can also be expanded to register several images at once. To bring this into reality however a good search algorithm is a necessity. It also requires appropriate selection of features in order to extract most relevant information. For example, the features used to register face images may not be suitable for medical images. Also, MST entropies can be used for non-rigid image registration, provided it is been made extremely computationally efficient.

Thus, this thesis bolsters the fact that the MST entropies can be used for registration by implementing a registration algorithm which searches over the affine transformation space.
APPENDIX
ADDITIONAL INFORMATION

This section covers the pseudocode of the algorithm.

Pseudo-code. The program consists of the following main functions:

- **FunctionRegister**: This function is the main function which calls all the subfunctions.
- **FunctionRoughRegister**: This function performs the rough image registration using intensity and histograms for entropy estimation.
- **FunctionExtractICAFeatures**: This function extracts the ICA features from the image.
- **FunctionMSTRegister**: This function performs the fine registration using the ICA features and the minimum spanning tree for estimating the entropy.

and the following supplementary functions:

- **FunctionAffineTransform**: This function performs the affine transformation of the input image according to the given parameters.
- **FunctionJointEntropyByHistogram**: This function computes the joint histogram and accordingly estimates the Shannon joint entropy.
- **FunctionMarginalEntropyByHistogram**: This function computes the histogram and accordingly estimates the Shannon marginal entropy.
- **FunctionEntropyByMST**: This function estimates the Renyi entropy (Joint and Marginal) using the minimum spanning tree.

Here are the main functions explained in depth:

**FunctionRegister**

**Input**:

*Image1, Image2*: The Images.
*BlockSize*: Size of the square block around each pixel (i.e. 2 x 2, 3 x 3 etc).
*Alpha*: The $\alpha$ factor in the Renyi Entropy.
**NumBins**: The number of bins used to compute the histogram.

**Output**: 

**AffineParameters**: The affine transformation parameters which when applied to the second image give a registered images. The parameters of affine are two rotations, scale, shear and translations in the x and y direction.

**MaxMeasure**: The Mutual Information like measure when the for the parameters above.

**Algorithm**: 

// Perform rough image registration of the images using intensity + histogram 
RoughAffineParameters = FunctionRoughRegister(Image1, Image2, NumBins);

// Extract ICA features from Input Images 
[icaFeatures1, icaFeatures2, ICAFFeatures] = FunctionExtractICAFeatures(Image1, Image2, BlockSize);

// Perform fine image registration of the images using ICA Features, + histogram 
AffineParameters = FunctionMSTRegister(ICAFeatures, Alpha, RoughAffineParameters);

EndFunction

**FunctionRoughRegister**

**Input**: 

*Image1, Image2*: The Images. 

*NumBins*: The number of bins used to compute the histogram.

**Output**: 

*RoughAffineParameters*: The affine transformation parameters which when applied to the second image give a roughly registered images. The parameters of affine are two rotations, scale, shear and translations in the x and y direction.
Algorithm:

// The Image1 is kept fixed while Image2 is transformed over the search space. Over all the six parameters of the affine search space using a rough resolution to scan the space do

tempImg = FunctionAffineTransform(currentTransformValues);

djointEntropy = FunctionJointEntropyByHistogram(overlap areas of the Image1 and tempImg);

marginalEntropy1 = FunctionMarginalEntropyByHistogram(overlap area of Image1);

marginalEntropy2 = FunctionMarginalEntropyByHistogram(overlap area of Image2);

currentMI = marginalEntropy1 + marginalEntropy2 - jointEntropy;

if ( currentMI > maxMI )

maxMI = currentMI;

RoughAffineParameters = current affine parameters;

endIf

endWhile

return RoughAffineParameters;

EndFunction

FunctionExtractICAFeatures

Input:

Image1, Image2: The Images.

BlockSize: Size of the square block around each pixel(i.e. 2 x 2, 3 x 3 etc).

Output:

icaFeatures1: The ICA features of image1 icaFeatures2: The ICA features of image2 ICAFeatures: The ICA features of the two images
Algorithm:

For both Image1 and Image2 do

// Get the block neighborhood of each pixel in both the images.
BlockInput1 = GetBlockNeighborhoodImage(Image1, blockSize);
BlockInput2 = GetBlockNeighborhoodImage(Image1, blockSize);

// Compute the ICA features. We have used the FastICA package [16] to do so. The package is freely available for use.
icaFeatures1 = FASTICA(BlockInput1);
icaFeatures2 = FASTICA(BlockInput2);

// Now combine the ICA features
ICAFeatures = icaFeatures1 \cup icaFeatures2;

return icaFeatures1, icaFeatures2, ICAFeatures;

EndFunction

FunctionMSTRegister

Input:

icaFeatures1: The ICA features of image1.
icaFeatures2: The ICA features of image2.
ICAFeatures: The ICAFeatures of the Images combined.

Alpha: The Alpha used to compute the Renyi entropy

RoughAffineParameters: The Ballpark estimate from the histograms around which the search is performed.

Output:

FinalAffineParameters: The affine transformation parameters which when applied to the second image give a registered images. The parameters of affine are two rotations, scale, shear and translations in the x and y direction

MaxMI: The value mutual information when the images are totally aligned.
Algorithm:

// The Image1 is kept fixed while Image2 is transformed over the search space.

Over all the six parameters of the affine search around the ballpark estimate RoughAffineParameters using a fine resolution to scan the space do

// First Pick Random Samples from the joint as well as the individual feature space.

(samplJoint, sample1, sample2) = pickRandomSamples(ICAFeatures, icaFeatures1, icaFeatures2);

tempImg = FunctionAneTransform(currentTransformValues);

jointEntropy = FunctionEntropyByMST(sampleJoint);
marginalEntropy1 = FunctionEntropyByMST(sample1);
marginalEntropy2 = FunctionEntropyByMST(sample2);

CurrentRegistrationMeasure = marginalEntropy1 + marginalEntropy2 - jointEntropy;

if( CurrentRegistrationMeasure > maxMeasure )

maxMeasure = CurrentRegistrationMeasure;

FinalAffineParameters = current affine parameters;

endIf

endWhile

return FinalAffineParameters, maxMeasure;

EndFunction

Now the pseudo code of the supplementary functions is:

FunctionAneTransform

Input:

Image: The Image to transform.
TransformParameters: Parameters by which the image is to be transformed

Output:

TransformedImage: The transformed image after applying the transformation.

Algorithm:

// The transform parameters are 2 rotation angles (θ and φ), scale, shear, and translations (xT and yT)

\[
\begin{bmatrix}
a & b \\
c & d \\
\end{bmatrix} = \begin{bmatrix}
2^{\text{scale}} & 0 \\
0 & 2^{\text{scale}} \\
\end{bmatrix} \times \begin{bmatrix}
\cos(\theta) & \sin(\theta) \\
-\sin(\theta) & \cos(\theta) \\
\end{bmatrix} \times \begin{bmatrix}
2^{\text{shear}} & 0 \\
0 & 2^{-\text{shear}} \\
\end{bmatrix} \times \begin{bmatrix}
\cos(\phi) & \sin(\phi) \\
-\sin(\phi) & \cos(\phi) \\
\end{bmatrix}
\]

affineTransformationMatrix = \begin{bmatrix}
a & b & 0 \\
c & d & 0 \\
xT & yT & 1 \\
\end{bmatrix};

// Now call the MATLAB’s Image Transform Routine which transforms the given image according to the given affine transform matrix and fills the empty areas according to the interpolation method specified.

TransformedImage = FunctionMATLABImTransform(Image, affineTransformationMatrix, interpolation);

return TransformedImage;

EndFunction

FunctionJointEntropyByHistogram

Input:

Image1, Image2: The Images.

NumBins: The number of bins used to compute the histogram.

Output:
**Joint Entropy**: Joint Shannon Entropy of the two images computed by the histogram method

**Algorithm**:

\[
\text{interval} = \frac{\text{Range}}{\text{NumBins}};
\]

//Compute the two dimensional normalized joint histogram of the two images.

\[
\text{jointHist} = \text{FunctionJointHistogram}(\text{Image1, Image2, interval});
\]

\[
\text{jointEntropy} = 0;
\]

\[
\text{for } i = 1 \text{ to NumBins}
\]

\[
\text{for } j = 1 \text{ to NumBins}
\]

\[
\text{jointEntropy} = -1 \times \text{jointHist}(i,j) \times \log \text{jointHist}(i,j);
\]

\[
\text{endFor}
\]

\[
\text{endFor}
\]

**return jointEntropy**;

EndFunction

**FunctionMarginalEntropyByHistogram**

**Input**:  
*Image*: The Image.  
*NumBins*: The number of bins used to compute the histogram.

**Output**:

* marginalEntropy: Marginal Shannon Entropy of the image computed by the histogram method

**Algorithm**:

\[
\text{interval} = \frac{\text{Range}}{\text{NumBins}};
\]

//Compute the one dimensional normalized histogram image.

\[
\text{marginalHist} = \text{FunctionMarginalHistogram}(\text{Image, interval});
\]

\[
\text{marginalEntropy} = 0;
\]
for i = 1 to NumBins
    \[ marginalEntropy = -1 \times \text{marginalHist}(i,j) \times \log \text{marginalHist}(i,j); \]
endFor

return marginalEntropy;
EndFunction

FunctionEntropyByMST

Input:

\textit{featureMatrix}: The feature matrix whose entropy is to be calculated. The size of the matrix is \( N \times d \).
\[ [N \ d] = \text{size(featureMatrix)}; \]

where \( N \) :- Number of sample points and \( d \) :- Their dimension \( d \geq 2 \)

\textit{Alpha}: The Alpha Parameter that is used in computing the Renyi Entropy.

Output:

\textit{Entropy}: Renyi Entropy of the feature matrix computed by the Minimal Spanning Tree method.

Algorithm:

/* In this function we attempt to compute the approximate Renyi Entropy based on the formula 3.8. We do not know the factor \( C(\alpha, d) \) and thus it is approximate. */

\[ \gamma = (d - \text{Alpha}) / d; \]

/* Compute the minimal spanning tree and its length. We use the Kruskal Algorithm provided in the ”Boost Graph Library” (Available at:

lengthMST = FunctionBOOSTKruskal(featureMatrix);

gamma = (d - Alpha) / d;

Entropy = \[ \frac{1}{1-\text{Alpha}} \times \log \frac{\text{lengthMST}}{N^{\text{Alpha}}}; \]
return Entropy;

EndFunction
REFERENCES


BIOGRAPHICAL SKETCH

I graduated with a bachelor of engineering degree from the Government College of Engineering, Pune, India. I started my master of science program at the university of Florida in August 2002 and I am planning to graduate in a month! My academic interests lie in computer vision and image processing and this thesis was a great learning experience for me. Apart from this, I enjoy painting and badminton.