JOINT RESTORATION AND COMPRESSION OF DOCUMENT IMAGES WITH BLEED-THROUGH DISTORTION

Patrick Dano

A Thesis submitted to the Faculty of Graduate and Postdoctoral Studies in partial fulfillment of the requirements for the degree of Master of Applied Science, Electrical Engineering

May 2003

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering
University of Ottawa
Ottawa, Ontario, Canada

Contents

List of Tables

List of Figures

Abstract

This thesis presents research on the restoration of documents suffering bleed-through and the joint compression of the original document and its bleed-through corrected version. This topic as presented in this thesis requires four steps: i) registration, ii) segmentation, iii) inpainting and iv) compression. First, a review of existing methods for registration and segmentation are presented, followed by a review of some existing image compression formats for the purpose of compressing the images. Then, the development of efficient registration and segmentation algorithms that are based on previous research are presented, along with a new method for joint document compression. The methods presented in this thesis are compared with previous methods and are shown to have improvements over them. In terms of registration, the method presented here is faster in reaching a more accurate final solution than previous registration methods presented for this problem. The segmentation algorithm presented here is shown to accurately segment portions of bleed-through away from the rest of the document. On the topic of joint compression, it is shown in this thesis that jointly compressing a document with its bleed-through corrected version will reduce the bandwidth required to transmit those document images, when compared to simply compressing the two images individually.

Acknowledgments

Firstly, I would like to thank my supervisor, Dr. Eric Dubois for his help and his time throughout the duration of my graduate studies. Also I would like to thank the University of Ottawa Interfaculty Collaborative Research Initiatives program which was the sponsor of my research. Finally, I thank the National Archives of Canada for their help in providing original archive document images.

Chapter 1

Introduction

1.1 Problem Understanding

Housed within the libraries of the world is a great collection of rare books and handwritten manuscripts. The information contained in these collective works was often unavailable to most people due to the sometimes high expense of time and money required to travel to the locations in which documents of interest were stored. With the recent advent of the Internet, or more specifically, the growth in computer and telecommunications technology, it is now possible to create online digital libraries enabling most people around the world to potentially access any document anywhere. Along with the advent of such digital libraries, many problems have been discovered. Two of these problems and their proposed solutions will be discussed in this thesis.

The first step in creating a digital library is to get the documents into digital format. The most popular method of digitizing any picture is to have it scanned; the actual method of digitizing the document is not very important in the scope of this thesis, but it is important to state that the quality of the scans must be fairly high.

The age of the documents and the fact that they may contain writing on both sides bring about the first problem. This is the problem of bleed-through. Bleed-through occurs when ink from one side of the document leaks through to the other side. When scanning such documents, the bleed-through is scanned just as faithfully as any other useful information on the rest of the document, as seen in Fig. 1.1.

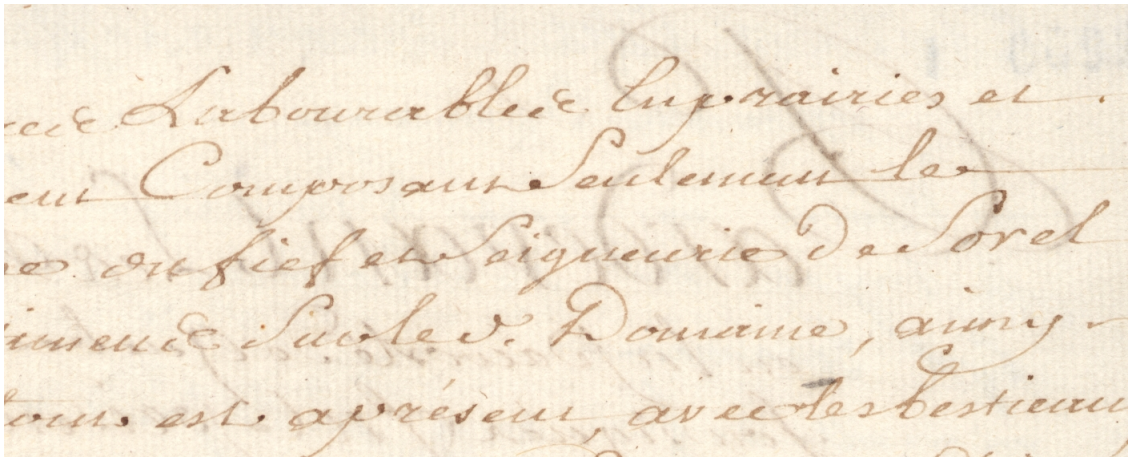

Figure 1.1: An example of an image with bleed-through

The second problem is associated with scanning the documents, and that is the resulting size of the documents. Uncompressed scans of full page documents occupy a large space, and pose a problem for distribution over networks. In order to preserve document quality, a high scanning resolution is required (at least 300 dpi), but at these scanning resolutions the file sizes become large and negatively impact the speed of transmission over networks. At lower resolutions, the resulting image is blocky and does a poor job of representing the original document. Please refer to Fig. 1.2 in order to see the difference in scanning resolutions.

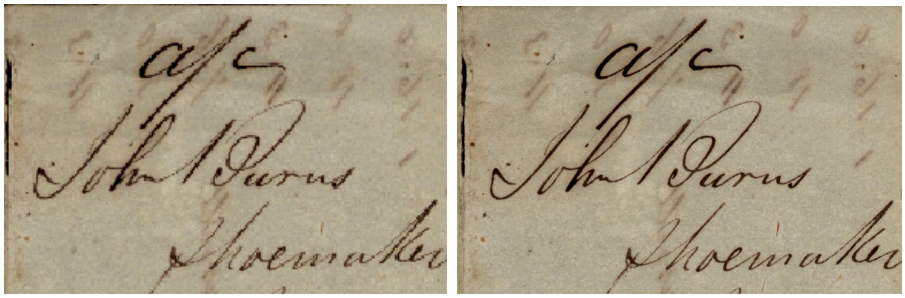

Figure 1.2: Portion of an image scanned at 150 dpi (left) and 300 dpi (right)

The two problems identified above need to be solved in order to successfully create a digital online library that is useful to the general public. The goal of this thesis is twofold: to present a method to restore legible documents from documents degraded by bleed-through, and to discuss methods of compressing the images so they are accessible quickly via the Internet.

1.2 Motivation for the Study

Current literature and image editing applications do not provide any techniques specifically for the restoration of documents with bleed-through. For example, commercial image editing suites such as Adobe Photoshop or Corel Photosuite can only contribute pixel thresholding and contrast enhancement to this type of document restoration. It is possible to use pixel thresholding to remove bleed-through only when the pixels that constitute bleed-through cannot be confused with other pixels, i.e., there is no overlap between the histograms of the bleed-through and the remainder of the document. The obvious problems with this technique is that it will only work with documents containing very light bleed-through (which are rarely a problem in the first place), and that the resulting image will be a bilevel image.

One of the bleed-through removal techniques currently defined involve the following 5 steps:

1. Scanning the front and back sides of the affected document, referred to as *recto* and *verso* respectively.

2. Flipping the verso left to right to yield the *flipped verso*.

3. Aligning the recto and flipped verso images (*registration*).

4. Segmenting the images into four different regions (*segmentation*).

5. Replacing bleed-through regions with an estimate of the background (*inpainting*).

Considering that simple pixel thresholding will not work for most document images of interest, most bleed-through removal techniques perform operations on both sides of the document. The algorithms described in [1] and [2] are better suited for show-through removal rather than bleed-through removal, due to the model used in the analysis. The algorithm described in [3] and [4] is better suited to bleed-through removal and has been adopted as a starting point for the work done in this thesis. The problems with the methods described in [3] and [4] are i) the registration method did not result in good alignment of the recto and verso images, ii) the segmentation method was not satisfactory in segmenting the document into the four defined regions. (See Fig. 2.4 for a visual explanation of the four regions). Furthermore, no previous work has specifically addressed the compression of documents with bleed-through.

1.3 Contributions of this Thesis

The main contributions of this thesis are as follows:

1. I have expanded upon the registration techniques used in [3] and [4], and carried out a comparison and evaluation between the original technique and the technique developed in my work. The new registration technique was developed in order to overcome the two major problems of inaccurate alignment and memory management when working with large files. The misalignment of carefully scanned document images is not very large. In the documents worked on in this thesis, the maximum translation between the recto and flipped verso was 15 pixels in the horizontal direction, and 20 pixels in the vertical direction. This translates to a physical distance of less than $\frac{1}{10}th$ of an inch in either direction at a scanning resolution of 300 dpi.

2. I have introduced a segmentation method that can faithfully recognize the four regions of a document that have been defined for this method of bleed-through reduction.

3. I have introduced an inpainting scheme so as to replace any portions of the

document that are bleed-through with an accurate representation of the background. The inpainting scheme introduced in this thesis is quite simple, yet results in document images that are easier to read and understand.

4. I have defined a compression method that takes into account the redundancy between a document image and its bleed-through corrected version. The method defined in this thesis reduces the combined size of the original image and its bleed-through corrected version when compared to simply compressing the two images individually. The topic of joint compression has not been covered for this application in any previous work, and is covered here in order to allow for any reader of a bleed-through corrected document to have access to the original document as well as its bleed-through corrected version. This is done in case a person reading the documents feels that the corrected version may be missing some important details that could be on the original.

1.4 Organization of this Thesis

In Chapter 2 of this thesis, I present the argument against simple pixel thresholding for bleed-through removal and explain why another method of restoration is needed. Chapter 2 also contains a brief introduction to the steps involved in the proposed method of restoring documents with bleed-through.

In Chapter 3, I discuss and evaluate different methods of image registration and how they can be made to apply to this application.

Next, in Chapter 4, the topic of image segmentation is discussed. A short discussion is given concerning the work previously done in the field along with the work presented in this thesis. Chapter 4 also includes a short section on image inpainting.

In Chapter 5, I discuss different methods of compressing the images. A large number of image compression methods exist that could be used in order to compress these images for distribution over a network. Each of these methods are tailored for specific applications, with some towards document compression. A few of these formats will be discussed and benchmarked for our purposes in order to find a suitable

image compression format that provides both very good readability of the document, and a very good compression rate in order to facilitate document distribution.

Chapter 6 concludes the discussion of the overall technique along with a presentation of the results achieved. Furthermore, future work remaining is also discussed in this chapter.

Chapter 2

Background on Bleed-Through Removal in Documents

2.1 Thresholding for Bleed-Through Removal

First, I begin with a short explanation of why a new method of document restoration is needed.

As discussed in the introduction, it is possible to use pixel thresholding in order to remove bleed-through from scanned documents, though it would only work for documents with light bleed-through. Fig. 2.1 shows an image of a document with light bleed-through.

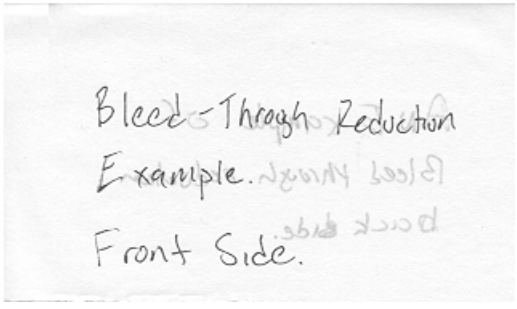

Figure 2.1: Document with light bleed-through

Obviously this document is not of great interest to anyone anywhere, but it will suffice for explaining why another method of document restoration is needed. Fig. 2.2 is the result of thresholding the above image so that no bleed-through shows.

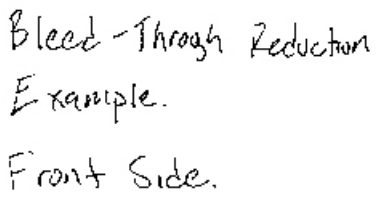

Figure 2.2: Thresholded version of the image in Fig. 2.1

It is clear that most of the information on the front side of the page remains, while all the extra "noise" is eliminated. However, along with the bleed-through, all the background texture and some writing is also removed. Also, it is important to note that the output image is bilevel only.

This may not be a problem for some documents (such as newspapers), but many documents that are of interest to scholars contain severe bleed-through, and perhaps some background information. For instance, the image shown in Fig. 1.1 is written on parchment paper, and has a certain texture that is reproduced in the scan. Let us look at the thresholded output of that particular image.

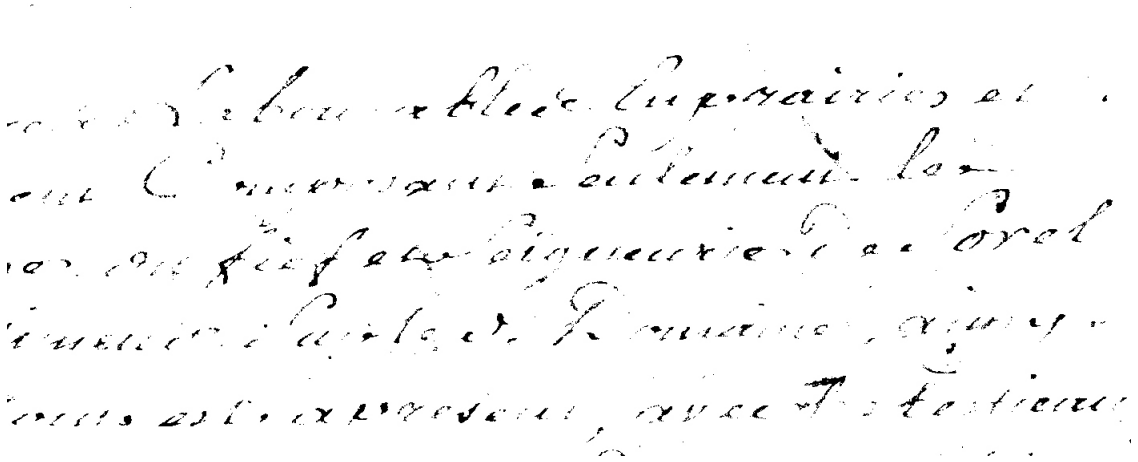

Figure 2.3: Thresholded output of the image shown in Fig. 1.1

Clearly, this document looks nothing like the original, and it is also illegible. Fig. 1.1 is a typical document found in archives. The output provided by the thresholding is not very useful to anyone who wishes to learn anything from the document. Thresholding fails to keep a faithful reproduction of the original document. This is why another method of document restoration is needed.

Any document restoration algorithm that is to be used for documents such as shown in Fig. 1.1 must keep a faithful representation of the original document, but without the bleed-through, so that anyone using it can have the pleasure of seeing it at it was intended to be seen when it was originally written.

2.2 Major Elements of the Proposed Document Restoration Method

In this section I introduce the three main parts of new algorithms that deal with document restoration. I begin with a short introduction of the registration method described in [4] and the segmentation technique described in [3]. Following that, I give a brief explanation of the show-through reduction techniques given in [1] and [2],

and explain why those techniques are not well suited for this purpose.

2.2.1 Image Registration

The algorithms described in this thesis require that the recto and flipped verso images be perfectly aligned. Since these two images are generally scanned in two separate operations, perfect alignment of the two images is generally not acheived, so image registration is required. Refer to Fig. 3.3 for an example of images with bad alignment. There has been a lot of work done over the years in image registration, though not much of it can be directly applied to this application. These registration algorithms assume two images that contain a lot of redundant information, whereas in any document with bleed-through, the only redundant information consists of areas of bleed-through. It is possible to modify registration techniques to make them work for our application. One registration technique that is available for our application, described in [4] has been modified for use in this application. The algorithm described in [4] uses feature points in the image in order to align the recto and verso images, although the results generated were not quite satisfactory for further use. The registration technique described in this thesis uses [4] as a starting point, the details of the algorithm are described in Chapter 3.

2.2.2 Image Segmentation

The topic of image segmentation is also widely studied in the field of image processing, yet again, most work done on this subject is not related to the segmentation we require here. Most image segmentation algorithms are interested in identifying two parts of an image, foreground and background. In our application, we are interested in defining four regions, namely i) recto writing, ii) background, iii) bleed-through, iv) overlapping bleed-through and recto writing. One segmentation technique is detailed in [3]. The image in Fig. 2.4 shows the four different regions and what they look like. The main difference between the algorithm defined in this thesis and the algorithm in [3] is that the algorithm described here incorporates the correlation of the neighbourhoods surrounding a pixel of interest. The details of this algorithm are

described in Chapter 4.

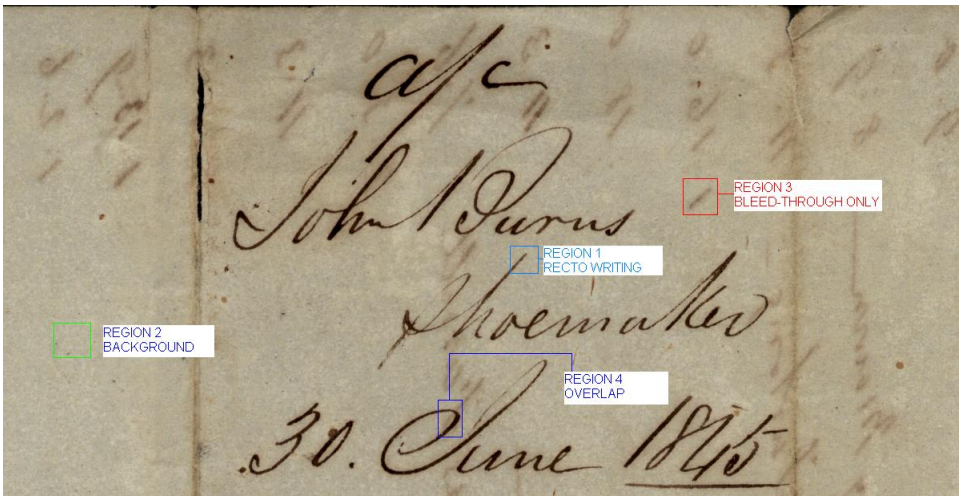

Figure 2.4: Image showing the regions of interest for the segmentation procedure

2.2.3 Image Inpainting

The topic of image inpainting has also been studied in image processing, although most of the work done in this field concerns itself with the removal of unwanted parts of images which may include text, scratches or items within the image itself. For an introduction to work in image inpainting, refer to [5] and [6] for techniques in local inpainting. Any image inpainting technique can be adapted to work for our application. In this thesis, a simple technique involving calculating the background average of an image was used to generate an estimate of the background. This is also described in more detail in Chapter 4.

2.3 Methods of Restoring Images suffering Show-through

There are image restoration schemes available that do not explicitly use the three steps described above. These schemes are detailed in [1] and [2]. I will give a brief explanation of these schemes, although I will not go into their details since I do not think that these methods are directly applicable to our problem. The restoration methods described in [1] and [2] concern themselves mostly with show-through removal, rather than bleed-through removal. Show-through occurs during the scanning of a document such that the verso is visible through the recto, whereas bleed-through is when ink bleeds through the paper, and is physically present on the recto of the document.

The model chosen by Sharma in [1] is based on the apparatus that is used to convert the document into a digital format (the scanner). If there is no printing on either side, then the light that strikes the optical sensor of the scanner has two components, light that hits the paper and is reflected back to the sensor, and light that passes through the paper and is reflected back through it by the white backing of the scanner. Refer to Fig. 1 and Fig. 2 of [1] for a visual explanation. This model only allows for distortion in the image that is due to a reflection of light from the white backing of the scanner back through the paper and not from ink that has actually seeped through the paper, which is the problem that is identified in this thesis. This is therefore not an acceptable model for the problem of bleed-through. In addition, this method also does not require perfect alignment of the recto and flipped verso images. Slight differences in the alignment of the images are accounted for in the restoration model since Sharma proposes the use of a non-static filter. An adaptive algorithm is used to update the coefficients of the restoration filter. Sharma suggests the use of the Least Mean Square (LMS) algorithm for this purpose.

In Chapter 3, a description of the registration algorithm and how it was modified from the original algorithm described in [4] is given. Chapter 4 contains a description along with a thorough discussion of the segmentation algorithm. In Chapter 5, the

topic of joint image compression will be covered, the main point of Chapter 5 is not to provide a new method of compressing document images, but to make use of current image compression standards to compress both the original document and its bleed-through corrected version.

Chapter 3

Image Registration Procedures

The process of registering the two sides of the scanned document is quite important in the system of bleed-through removal developed here. Documents that are not well registered will not yield good results. In this chapter I first give a short introduction to the various types of image registration techniques available, then I give detailed explanations of some registration techniques that are relevant to this project. Afterwards I explain the technique adopted and modified in order to make it more suited to our needs. Afterwards, the performance of all these techniques will be evaluated.

3.1 General Image Registration Techniques

In this section, I mostly concern myself with general image registration techniques and why they are not completely suited to our needs. Most of the information presented here is taken from [7].

I begin by introducing the four components of defining a registration scheme, and relate them to the registration scheme that is used within this project. As described in more detail in [7], a registration scheme is made up of a combination of choices for the following four components.

1. Feature Space: This is a subset of the images that will be used for matching the two images. In our case, this is the full set of pixels of the images.

2. Search Space: This is the class of transformations that are to be used to align the images. In our case we want to be able to perform a transformation with six parameters, two for translation and four for rotation, scaling and skew. Specifically, I will use the affine transformation.

3. Search Strategy: This is the rule that defines how to choose the next set of transformational parameters so as to eventually reach the optimal set. In our case, we use an optimization that minimizes the difference between the two 8-bit gray scale images.

4. Similarity Metric: This can be thought of as a benchmark for each set of transformational parameters. According to our search strategy, which is a minimization of the total squared difference between the two images (recto and flipped verso), each set of parameters is compared so that the optimal set of parameters is the one with the lowest difference metric, hence the highest similarity metric.

Most of the research done in image registration has been for applications such as (and there are many more) i) Integrating information of a scene that is taken from two different sources (for example, a stereoscopic image), ii) finding changes in a scene from which two images are taken at different times, iii) finding changes in a scene of which two images are taken under different conditions, and so on.

Clearly, each of these examples are of a set of images that contain a lot of redundant information among each other. Specifically, these images contain a significant amount of overlap that will be used in order to align them. In our application, the two images are different from each other except for the areas where bleed-through is present, these areas may be large or they may be small; it is not prudent to assume either. For this reason, a new registration scheme needed to be developed. However there has been some previous work done on this topic for this application, and I will describe that work and the modifications to it in this chapter.

3.2 Registration Methods Considered in the Project

Three different types of registration techniques were analysed and benchmarked in terms of

1. Quality of registration
2. Time required for registration

All three of the registration methods are feature based matching methods, meaning that they try to match the corresponding features of both images using an affine transformation with six parameters. The ideal transformational parameters found are those that correspond to the lowest total squared difference between the pixels of the two images.

Two of the three registration methods used are automatic, meaning that the algorithm does not need any user intervention in order to find the corresponding features. The third method is a manual point matching method in which the user inputs a set of corresponding feature points.

Before I explain any of the methods, I need to define some terms. Let $f_r[m, n]$ be the recto of the scanned document, and let $f_v[m, n]$ be the flipped verso of the scanned document. The range of $f_r[m, n]$ and $f_v[m, n]$ is from 0 to 1, where 0 is black, and 1 is white. If the original images are colour images, only the luminance component of the images are used for registration. The points $[m, n]$ are integer coordinates of an image sampled on a rectangular lattice with even spacing in both the horizontal and vertical directions. The spacing of each pixel depends on the sampling resolution, but is usually either $\frac{1}{300}$ or $\frac{1}{600}$ of an inch. In order to match feature points of the images we need one of the sides to be flipped left to right, so that the bleed-through in one image corresponds to the writing in the other. In order to do this, $f_v[m, n]$ must be a copy of the back side scan that is flipped left to right. Before registration, the coordinate systems of $f_r[m, n]$ and $f_v[m, n]$ are not aligned. This could be due to the misalignment of the image during the scanning of the recto and verso and/or slight distortion during the scanning process. The objective of the registration is to find parameters that will align any discrepancies that are introduced during the scanning

process. In short, what we need is to find is an affine transformation that can map corresponding feature points in $f_v[m,n]$ to $f_r[m,n]$. I denote the aligned $f_v[m,n]$ as $f_v^a[m,n]$ where

$$f_v^a[m,n] = \mathcal{A}_t(f_v[m,n]) \tag{3.1}$$

where $\mathcal{A}_t$ is a linear but shift-variant operator, and $\mathbf{t} = [a_{11}\ a_{12}\ a_{13}\ a_{21}\ a_{22}a_{23}]$ is a set of transformation parameters for the affine case. Here, a_{13} and a_{23} are the horizontal and vertical shifts respectively, and the remaining four are used for rotation and/or skew. $f_v^a[m,n]$ is obtained by interpolation of $f_v[m,n]$ using a bicubic interpolation algorithm.

Given the above notation, the three registration methods are described in the following subsections.

3.2.1 Manual Method

In this section I describe a method of manual registration that was used. This method is not ideal in terms of practical use due to the high level of human involvement that is required. However, the method is good for returning the transformation parameters that are associated with the lowest differences of any method. This method is mostly used to show that perfect alignment of $f_r[m,n]$ and $f_v[m,n]$ is possible.

This algorithm works by finding the transformation from a corresponding set of matched points that must be provided by the user. The user must take care to ensure that all the points are matched exactly across $f_r[m,n]$ and $f_v[m,n]$. $\mathcal{A}_t$ is then found by performing a least squares matrix solution.

We know from equation (3.1) that the aligned verso is a transformation of the scanned verso by a certain set of parameters. Assuming that this transformation is an affine transformation, it can be re-written (for a single point) as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} n \\ m \end{pmatrix} + \begin{pmatrix} a_{13} \\ a_{23} \end{pmatrix} \tag{3.2}$$

If the points $p' = (x', y')$ and $p = (n, m)$ were known, where p is a point on the recto and p' is on the verso, then the parameters $a_{11}\ a_{12}\ a_{13}\ a_{21}\ a_{22}\ a_{23}$ could be found

if we had at least three or more known matched points. Note that if we only had three points, then we would get an exact solution, and would not need to use a least squares optimization. However, due to errors in matching the points, it is better to use more than 3 points distributed evenly throughout the image, and find the least squares solution.

In addition, note that according to equation (3.2), the points $p^{'} = (x^{'}, y^{'})$ are not necessarily specified to be integer. The new image $f_v^a[m, n]$ is obtained by interpolating $f_v[m, n]$ on the set of points obtained by equation (3.2).

In order to write the least squares solution, I rewrite equation (3.2) so that it is easier to manipulate into the general case of M points.

$$\begin{pmatrix} x^{'} \\ y^{'} \end{pmatrix} = \begin{pmatrix} n & m & 0 & 0 & 1 & 0 \\ 0 & 0 & n & m & 0 & 1 \end{pmatrix} \begin{pmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ a_{13} \\ a_{23} \end{pmatrix}$$

Next, the formulation is done for a general case of M matched points.

$$\begin{matrix} & & & \mathbf{X} & & & & \mathbf{t} & & \mathbf{x}^{'} \end{matrix}$$
$$\begin{pmatrix} n^{(1)} & m^{(1)} & 0 & 0 & 1 & 0 \\ 0 & 0 & n^{(1)} & m^{(2)} & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ n^{(M)} & m^{(M)} & 0 & 0 & 1 & 0 \\ 0 & 0 & n^{(M)} & m^{(M)} & 0 & 1 \end{pmatrix} \begin{pmatrix} a_{11} \\ a_{12} \\ a_{21} \\ a_{22} \\ a_{13} \\ a_{23} \end{pmatrix} = \begin{pmatrix} x^{(1)^{'}} \\ y^{(1)^{'}} \\ x^{(2)^{'}} \\ y^{(2)^{'}} \\ \vdots \\ x^{(M)^{'}} \\ y^{(M)^{'}} \end{pmatrix}$$

With this, we can solve for $\mathbf{t}$ using the least squares solution,

$$\min_t ||\mathbf{x}^{'} - \mathbf{X}\mathbf{t}||^2$$

yielding the standard result as found in [8]

$$\mathbf{t} = (\mathbf{X}^T\ \mathbf{X})^{-1}\ \mathbf{X}^T\ \mathbf{x}^{'} \tag{3.3}$$

Using the solution for $\mathbf{t}$, we can use equation (3.1) in order to find $f_v^a[m,n]$, the aligned verso.

The results for this method are shown next. The following set of images (recto and verso) were used to obtain a set of registered images. This is only a small portion (377 x 908 pixels) of a larger image shown in Appendix A.

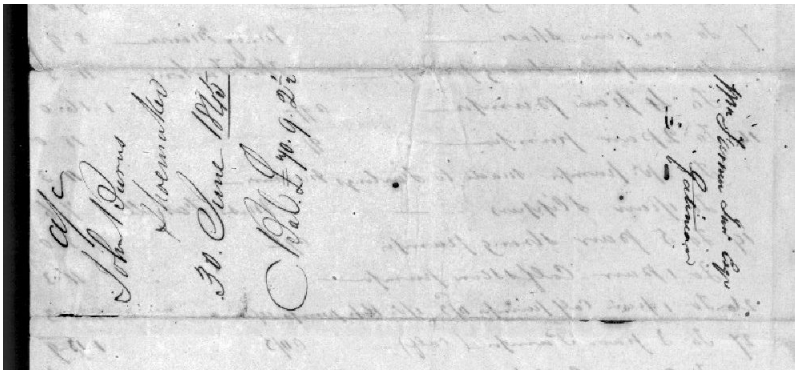

Figure 3.1: Recto of non registered image

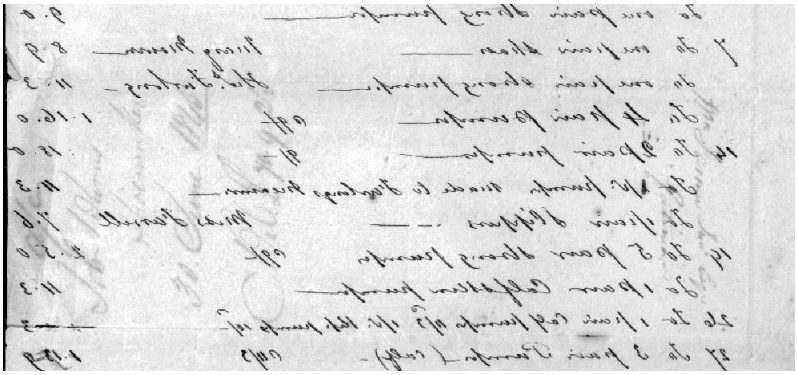

Figure 3.2: Verso of non registered image

The image shown in Fig. 3.3 is the superposition of the images shown in Fig. 3.1 and Fig. 3.2. You can clearly see the misalignment in the coordinate systems of the image. The images in Fig. 3.3 and Fig. 3.4 were generated using a pixel by pixel average operation in Adobe Photoshop.

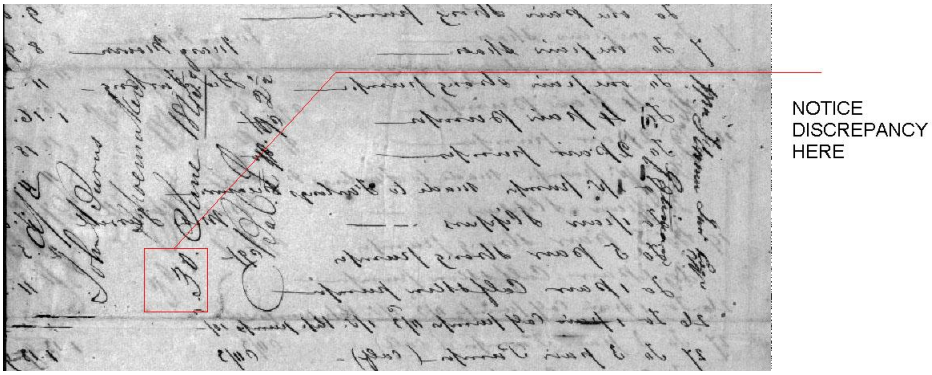

Figure 3.3: non-registered recto-verso superposition of the above images

The following image is the superposition of the recto and registered verso of the image using a 9 point manual registration. The red areas specify where the 9 points were taken from each image.

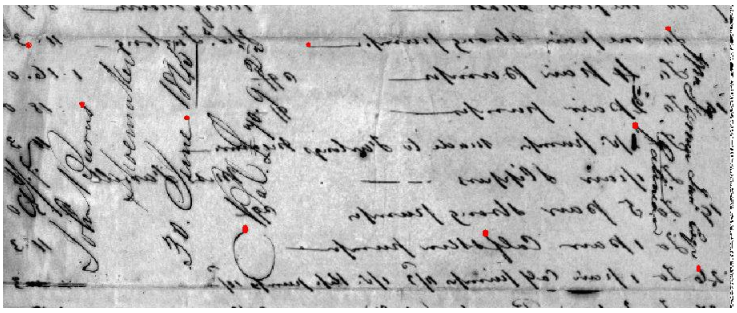

Figure 3.4: Registered recto-verso superposition of the above image using a 9 point manual registration

Clearly the coordinate systems of the two images are now more aligned than they were before the registration was performed. Upon close examination of the above image, one can see that the images are quite well registered.

This is a good sign, as it means that there is a way to get almost perfect registration of the two images. In the following sections, methods of automating the registration are presented and compared.

3.3 Automatic Registration Methods

In this section I describe the two automatic registration methods. In order to formulate this process, I need to first define a few more terms.

In both automatic methods, $\mathcal{A}_t$ is found by minimizing the total squared difference between the two images. As noted in [3] and [4], this difference can be denoted $D(\mathbf{t})$, where it is calculated as,

$$D(\mathbf{t}) = \sum_{m=1}^{P} \sum_{n=1}^{Q} \left(f_r[m,n] - \mathcal{A}_t(f_v[m,n])\right)^2 \tag{3.4}$$

The optimal parameters ($\mathbf{t}$) that define $\mathcal{A}_t$ are found by,

$$\hat{\mathbf{t}} = \arg\min_t D(\mathbf{t}) \tag{3.5}$$

Where P and Q are the number of rows and columns in the image. In short, the above equations mean that we are trying to find the best estimated transformation parameters ($\mathcal{A}_t$) by minimizing the squared difference between $f_r[m,n]$ and the transformed $f_v[m,n]$.

3.3.1 Global Method

Here, I discuss the first method of automatic registration used. This registration method was developed previously and is detailed in [3] and in [4]. During the formation of this method, the authors hypothesized that not all the pixels of the image are required for an estimate of $\mathcal{A}_t$. Given this, the sum in equation (3.4) does not have to be over all m, n but for a smaller subset of m, n. Equation (3.4) can be re-written as follows.

$$D(\mathbf{t}) = \sum_{m=1}^{\lfloor \frac{P}{K_1} \rfloor} \sum_{n=1}^{\lfloor \frac{Q}{K_2} \rfloor} \left(f_r[K_1 m, K_2 n] - \mathcal{A}_t(f_v[K_1 m, K_2 n)\right)^2 \tag{3.6}$$

where K_1 and K_2 are integer.

This method finds the six transformation parameters by first finding the two best global translation parameters a_{13} and a_{23}, and then using a minimization algorithm to find the final parameters. I explain this in more detail.

All the global methods first find the overall translation. This requires that a border be defined around the image which will be lost when the images are completely registered. This border is an area of pixels over which the images are translated over each other so as to find the initial coarse translational parameters.

In the first step, the initial transformation parameters are set to $a_{11} = a_{22} = 1$ and $a_{12} = a_{21} = 0$. In the first pass, the best values for a_{13} and a_{23} are found by first fixing $a_{23} = 0$ and finding a value for $\hat{a}_{13}^{(1)}$ that minimizes $D(\mathbf{t})^{(1)}$. Then, with a_{13} fixed at $\hat{a}_{13}^{(1)}$, the value of $\hat{a}_{23}^{(1)}$ that minimizes $D(\mathbf{t})^{(1)}$ is found. The values of a_{13} and a_{23} are any multiples of 0.5 that fall within the range defined by the border. In the second pass, the order in which the translation parameters are found is reversed. Fix $a_{13} = 0$ and find the value of $\hat{a}_{23}^{(2)}$ that minimizes $D(\mathbf{t})^{(2)}$. Then with a_{23} fixed at $\hat{a}_{23}^{(2)}$, find the value of $\hat{a}_{13}^{(2)}$ that minimizes $D(\mathbf{t})^{(2)}$. Finally, choose $\hat{a}_{13}^{(1)}$, $\hat{a}_{23}^{(1)}$ or $\hat{a}_{13}^{(2)}$, $\hat{a}_{23}^{(2)}$ which correspond to the smaller value (j) of $D(\mathbf{t})^{(1)}$ or $D(\mathbf{t})^{(2)}$. Then, the starting point for the subsequent minimization is $a_{11} = a_{22} = 1$, $a_{12} = a_{21} = 0$, $a_{13} = \hat{a}_{13}^{(j)}$ and $a_{23} = \hat{a}_{23}^{(j)}$. Using this search method, with a translation range of 10 pixels in both the horizontal and vertical directions, we have to calculate $D(\mathbf{t})$ 400 times. If we do a full search on all the points within the translation range, $D(\mathbf{t})$ will have to be calculated 10,000 times. The reduction in complexity is huge if we use the proposed search method. This search method will work every time if there is a proper match between the recto and flipped verso within the allotted translation range.

Once the best translational parameters are found, all six parameters then need to be found. This is done using the built in MATLAB function fminunc(). This function is an unconstrained multivariable non-linear minimization function. The function will find the minimum of an input function, which is equation (3.6), using the input parameters $\mathbf{t}$ as found earlier. Once this optimization is completed, the six transformation parameters are used to align the verso image onto the same coordinates as the recto image.

This method produced results that were visually good, meaning that the best visual results corresponded with a lowest total squared difference. However it was also observed that the values of the difference returned with this method were not as low as the values that were calculated using the manual method described in the

previous section, and therefore the registration was not perfect. Increasing the number of pixels used in equation (3.4) resulted in lower difference values. In fact, it was found that the lowest possible difference was found when all the pixels of the image were used to find the transformation parameters $\mathcal{A}_t$. This created a memory management problem as the images are of a large size, which in turn result in extremely large processing times due to resources being allocated to memory paging.

The following table lists the results of the minimization algorithm using different subsets of pixels of the images shown in section 3.2.1. Memory paging was not as big an issue with this image as is it a fairly small image (988x 457 pixels). This table serves only to show the difference between using a subset of pixels and all the pixels.

Number of Pixels	K1	K2	% of pixels	overall value of D	Time to Register (s)
6916	10	5	2	7549	15.2
12350	7	4	3.6	7151	22.6
22932	5	3	6.7	6792	41.8
42903	4	2	12.5	6754	78.7
57267	3	2	16.7	6751	104.2
171158	1	2	50	6748	351.8
171158	2	1	50	6748	347.6
342316	1	1	100	6747	664

Table 3.1: Results for registering the images in Fig. 3.1 and Fig. 3.2

The table shows the differences of the value of D returned by the algorithm using subsets of pixels of different sizes. Using a very small subset of pixels is not satisfactory for a good registration. The registration algorithm would only yield satisfactory if all (or near to all) the image pixels are used. The argument against a small difference in D, as seen in the difference between a pixel percentage of 16% and 100%, is significant since the small changes in D will reflect slightly imperfect registration, and will negatively impact the segmentation process.

Table 3.2 shows the time required to register images that are large enough to cause page swapping. The images are the full images of the image portions shown in Fig. 3.1 and Fig. 3.2, and can be found in Appendix A. The size of the images are 2028x1301 pixels.

Number of Pixels	% of Pixels	Time to Register (seconds)
8448	0.36	14.79
19448	0.82	43.53
31850	1.34	59.51
70070	2.95	151
203908	8.57	421.96
408148	17	848
611611	26	1252
2378508	100	did not finish ($>$ 9 hours)

Table 3.2: Time required to Register images based on the number of pixels used to register

Even though the algorithm was returning transformation parameters that corresponded with the lowest differences that could be found, it was obvious that the time that it required to do so could be drastically reduced. This led to the formulation of another automatic method which I describe next.

3.3.2 Global Registration Using Optimizations of Local Transformation Parameters

Due to the unsatisfactory results of the algorithm described in the previous section, I modified the algorithm in [3] so that it would allow for large images. This algorithm is a mix of the ideas of the manual algorithm and the above automatic algorithm.

This method takes the input images, and breaks it up into a set of smaller blocks. In this case each image was broken up into 6 horizontal and 6 vertical blocks. Each block is registered using the same method described in the previous section. The center point of each block from the recto image is taken as p, and the center point of each block from the verso image is taken as $p^{'}$. This set of 36 matched points is then used to find the parameters of $\mathcal{A}_t$ using equation (3.3). These parameters are then optimized using the built in MATLAB function fminunc(). The result is then used to find the aligned verso using equation (3.1).

The results produced with this method were not satisfactory. In some instances it

produced better results than the previous automatic method, but in almost all others, it produced results that were worse.

Fig. 3.5 is one result of this method. In this image, it is obvious that some blocks were registered fairly well, while others were not registered so well.

Figure 3.5: Overlap of an image registered using the hybrid block method

These results led to the hypothesis that one set of six global transformation parameters was not the ideal solution for images of large size. Another solution using smaller subsections of images is detailed in the next section.

3.3.3 Local Registration of Image Subsections

This method of registration splits the images into a set of row sub-blocks, and each sub-block is then registered and segmented individually. Each block should be chosen carefully so as to consider the border pixels lost during the registration. This method solves two problems. First, it allows for each block within an image to have its own set of transformation parameters. Second, it solves the problem of memory management.

The transformation parameters of each sub-block are found using the method described in section (3.3.1) where all the pixels are used. This allows for good registration for each of the image sub-blocks, and consequently the whole image.

This method solves the problem of the large processing times attributed to the automatic alogorithm, as listed in Table 3.2. Since each image is split up into a set of

B horizontal blocks, each sub-block is $\frac{1}{B}th$ the size of the original image. The value of B can be adjusted according to the size of the original image. Since each sub-block is registered independently of the others, this means that the segmentation algorithm must be run on each block rather than on the whole image. Within each block, the verso is aligned with the recto, and the recto is then segmented using the aligned verso. Following this, the recto is aligned to the verso using the inverse of $\mathcal{A}_t$, and the verso is then segmented using the aligned recto. This is done so as to not introduce any visual discrepancies in the output images. Since each block has a different set of transformation parameters, the resulting output image must be generated from the original image instead of the independent transformed blocks so as to avoid any breaks in the image that may be introduced by the individual transformation.

The following table shows the time advantage gained by using this block algorithm. Please note that in computers with large amounts of memory (larger than 1 Gigabyte), that this will not be an issue, and the original automatic method can be used, though that will result in a single transformation over the whole image. Also, note that the time required to register each image using the original automatic code is not listed (refer to Table 3.2) because I set a limit of 9 hours to register the images. The original algorithm did not produce an output in 9 hours for any of the images.

Image	Time required(min:sec)
Shoemaker Image	54:56
Bail Image	32:53

Table 3.3: Time required to register the images using the block registration method

In contrast to the above table, I obtained 1GB of memory and ran the original (non-blocking) algorithm on the images. I only concerned myself with the time required to complete the registration process, since the accuracy of the registration was good enough using the block method. The results are as follows.

The above table shows that even with plenty of memory (virtual memory was disabled) the blocking method results in a lower amount of time to register the images. Therefore I can conclude that the blocking method of registration serves to i)reduce

Image	Time required (hour:min:sec)
Shoemaker Image	1:19:51
Bail Image	3:49:22

Table 3.4: Time required to register the images using the full automatic method without memory limitations

the amount of time required to register two images, regardless of available memory, though the difference could be due to the time required to reach a final solution, and ii) to allow for different transformation parameters across different blocks of the image.

The following table gives the results of D across both images using both registration algorithms.

Image	Full Registration	Block Registration	9-point registration
Shoemaker Image	20881.44	17717.41	20391.61
Bail Image	48720.28	45027.25	43962.37

Table 3.5: Value of D across both images using all registration algorithms

3.4 Summary

In conclusion, using the comparisons and results presented in this chapter, I can easily conclude that the block automatic registration method results in a better registration in less time than the full automatic registration method. The results given by the blocking algorithm were good enough to use in the next stages of this project.

Chapter 4

Document Image Segmentation and Bleed-through Removal

In order to identify portions of the document image that are bleed-through, a good segmentation of the document is needed. In our case, it does not suffice to only segment the background from the foreground as the foreground may well be both recto writing and bleed-through.

4.1 Earlier Work in Document Segmentation

Before I begin an explanation of the algorithm developed, I include a short description of work that was previously done in this field. There are two earlier methods studied for this project. One technique of bleed-through removal was introduced by E. Dubois and A. Pathak in [3]. This was the basis of the work presented in this chapter. Another similar topic, show-through removal, as presented by G. Sharma in [1] was also studied for this thesis. However, the work done by Sharma is not directly applicable for our application.

First I introduce all the assumptions made by the authors of this earlier work, since it is crucial to the understanding of the algorithms. I also note the assumptions which I have carried from these studies into my own algorithm development.

The technique introduced in [3] assumes that there is an ideal copy of the recto and verso images (without bleed-through). In addition, the document in question can be broken up into 4 different classes, as follows:

Class	Description
1	Writing exists on the front side, no bleed-through is present.
2	No writing exists, but bleed-through is present.
3	No writing exists, and no bleed-through is present (blank paper).
4	Writing exists, and bleed-through is present. In this case, the ideal front side is close in value (pixel intensity) to the scanned flipped back side.

Table 4.1: List of defined areas in a document image with bleed-through

Refer to Fig. 2.4 for a visual example of the above regions. Furthermore, as explained in the introduction, this method requires scanning both sides of the document, and flipping (left to right) the verso.

4.1.1 Introduction to the Segmentation Technique of Dubois and Pathak

The technique presented in [3] proceeds to formulate the restoration as follows. The formulation in [3] is adapted to be consistent with the terminology used in this thesis. A model of the scanned front side ($f_r[m,n]$), where the samples [m,n] are on a rectangular sampling lattice, is broken down into components as a linear combination of an ideal background f_{br} (that of the paper), the ideal recto image f_{wr}, and an attenuated flipped ideal verso f_{wv}. Mathematically written as:

$$f_r[m,n] = f_{br}[m,n] + f_{wr}[m,n] + \alpha f_{wv}[m,n] - (1+\alpha) \tag{4.1}$$

where α is the attenuation factor.

The scanned verso image is also modelled in the same manner. In this method, the coordinate systems of the recto and flipped verso must be perfectly aligned. If this is not the case, this model will not yield very good results. Assuming that we have images that are well registered, then the ideal recto (or verso) image can be estimated.

The approach used to estimate the front side image is to use the partitioning information listed in Table 4.1. Based upon those classes, every pixel [m,n] can be identified as belonging in one of those areas. In this method, the only category of interest is the second one, the case of no writing, but bleed-through being present. This is the case which will need to be corrected. If a pixel [m,n] has properties

1. $f_v[m,n] < T$ and
2. $f_r[m,n]/f_v[m,n] > \alpha_0$

where, T and α_0 are some suitably chosen thresholds, then the pixel falls into Area 2. In words, this means that if at a certain point in the image the flipped back side is smaller than some threshold T (i.e darker), and the ratio of the front side pixel and the back side pixel is larger than some threshold α_0, then the pixel [m,n] is classified into area 2.

This is the brief explanation of that technique; please refer to [3] for a more detailed explanation. Again, this formulation was modified so that it is consistent with the terminology in this thesis. The major differences in terminology between this thesis and the terminology in [3] are as follows; i) the verso image here ($f_v[m,n]$) is defined to be flipped left-right, whereas in [3], it is defined as the verso scan with a linear operator that performs a left-right flip applied to it; ii) the range of the recto and verso scans is the same across both methods. However, in this thesis, 0 corresponds to black, and 1 to white, whereas in [3], 0 corresponds to white, and 1 to black. The major differences end there.

4.2 Segmentation Algorithm

The work done in [1] is not very well suited to our needs in this project because the work in [1] deals primarily with show-through. The mathematical model as described by [1] makes many assumptions which apply only to show-through and not to bleed-through. The approach taken in this thesis makes minimal assumptions and will work to remove both show-through and bleed-through.

The algorithm developed for this project is an extension of the ideas given in [3]. The classification in Table 4.1 was taken and applied to any further formulation done. It is important to note that of the four areas shown in 4.1, it is most important to identify the areas of bleed-through. The other three areas can be allowed to have some incorrect classification since they will not be altered in the removal of the bleed-through.

In this section, I describe the segmentation technique developed for this project in detail. I should also mention that there was not a specific mathematical model for a document that I used in order to find a formulation for the segmentation of a document, though I did make some assumptions about the documents of interest. The assumptions are listed in the sections that follow.

4.2.1 Background Detection

The segmentation of background is the easiest of all the sections. An assumption is made that in most documents the majority of the pixels are background pixels. In order to find all background pixels, an image histogram is used to find the lowest pixel value that has the largest number of occurrences, then a value that is 10 percent darker is used as a threshold to find background areas. The reason for this is that the background in most images is not uniform and is usually interspersed with darker pixels which give the background a texture. Using a threshold value that is 10 percent darker than found in the histogram gets most of these background fluctuation pixels and classifies them as background.

4.2.2 Further Segmentation

In this section I explain how the recto writing segmentation portion works. It should be clear that when doing the segmentation both the recto and registered verso of the same document are considered. Both images are traversed pixel by pixel using two windows. Within the area defined by the first window (of size MxM) the minimum value within the neighbourhoods (recto and verso) are calculated. These values are

$$r_{min}[i,j] = \min_{(k,l)\epsilon W} f_r[i-k, j-l] \tag{4.2}$$

for the recto, and

$$v_{min}[i,j] = \min_{(k,l)\epsilon W} f_v[i-k, j-l] \tag{4.3}$$

for the verso, where the window (W) is $W = \{(k,l)| -\frac{M-1}{2} \leq k \leq \frac{M-1}{2}, -\frac{M-1}{2} \leq l \leq \frac{M-1}{2}\}$, where M is odd. In the test cases it was found that for a 300 dpi image, M should be set to 5. The reason for a window is to define a local area around a pixel over which to find the minimum and maximum values. In a typical document with bleed-through, the ink does not seep from recto to verso in a perfect point, but there is usually some spreading. If we only consider one pixel, the segmentation algorithm will not correctly classify the pixels that are affected by the spreading of the ink. This will leave a "ghosting" effect in the resulting segmented image. Refer to Fig. 4.1 for a visual explanation. In Fig. 4.1, the original area of interest in the image is shown first, followed by the segmentation maps as produced by the segmentation algorithm. The colour white shows pixels that are background, black shows pixels that are writing, green shows pixels that are bleed-through, and purple shows areas of recto writing and bleed-through.

On close inspection of the segmentation map with a neighbourhood size of 1, one can see an outline of both purple and black around the green pixels, which shows that there will be a "ghosting" effect in the resulting restored image. In the segmentation map with a neighbourhood of 15, the segmentation itself is very thick and includes many background pixels as pixels of writing. It is for this reason that a window of size 5 was chosen to be the most suitable window size.

Figure 4.1: Original image (top-left), segmented output with window size set to 1 (top-right), segmented output with window size set to 5 (bottom-left), segmented output with window size set to 15. (bottom-right)

Within the areas defined by the second window (of size NxN). The correlation coefficients of the recto and verso neighbourhoods are calculated using the built-in MATLAB function corr2(). According to [9] $r = corr2(A, B)$ computes the correlation coefficient between A and B, where A and B are matrices of the same size. The function corr2() computes the correlation coefficient using

$$r = \frac{\sum_m \sum_n (A_{mn} - \bar{A})(B_{mn} - \bar{B})}{\sqrt{(\sum_m \sum_n (A_{mn} - \bar{A})^2)(\sum_m \sum_n (B_{mn} - \bar{B})^2)}} \tag{4.4}$$

where $\bar{A}$ and $\bar{B}$ are the mean values of the matrices A and B.

Now I introduce the comparison operations that are done in order to be able to classify what segment pixel $[m, n]$ falls in. In order to perform these comparisons, the following expression is calculated at each pixel $[i, j]$,

$$X[i, j] = \frac{r_{min}[i, j] - v_{min}[i, j]}{r_{min}[i, j] + v_{min}[i, j]} \tag{4.5}$$

Depending on the value of $X[i, j]$, a proper segmentation of the image can be made.

Before the results of equation (4.5) are explained, some values used in the segmentation algorithm must be described. These values are *bleed-through threshold* (T_{bt}) and *correlation threshold* (T_c).

The bleed-through threshold (T_{bt}) is the value used to distinguish areas of definite writing (Area 1) from areas of possible bleed-through. The correlation threshold (T_c) is the value that is used to classify areas of possible bleed-through into either definite bleed-through (Area 2) or into areas of writing and bleed-through (Area 4). The values for T_{bt} and T_c were chosen after running numerous tests on a number of images. However, the basic reasoning behind these thresholds are given next.

An explanation of equation (4.5) must be given at this point. Keep in mind that all pixel values are positive, and that a darker pixel has lower value than a lighter pixel. Given these restrictions, then the value of X is a ratio that results in a value whose absolute value is always less than 1. The division by $(r_{min} + v_{min})$ is used in order to create a larger difference in the value of X for pixels that are close in intensity. For example, if $r_{min} = 0.1$ and $v_{min} = 0.15$, we have $X = -0.2$. If the division is not present, we would have $X = -0.05$. The importance of this difference

in explained in the text that follows.

The absolute value of X describes the difference between the minimum and maximum pixel values within a neighbourhood. If $|X|$ is close to 1 then the darkest pixel and the lightest pixel in a neighbourhood have a large discrepancy in their luminance, and if $|X|$ is close to 0, then the two pixels are close in luminance. This will help in segmenting the images since (as shown in Table 4.1) Areas 1 and 2 will have pixels that have a large discrepancy in luminance ($|X|$ close to 1), and Area 4 will have pixels with a small discrepancy in their luminance ($|X|$ closer to 0). Furthermore, if X is negative, which means that $r_{min} < v_{min}$, then it is clear that the recto neighbourhood contains a darker pixel than the verso neighbourhood, meaning that the current pixel $[i, j]$ is most probably writing. If X is positive, then checking $|X|$ can give a good idea of how close, in luminance, the front side pixel is to the back side pixel.

Now the following three comparisons can be made using the value of X. Recall that the background pixels have already been segmented from the original image.

If X is smaller than T_{bt}, then we know that the recto neighbourhood contains a darker pixel than the verso neighbourhood, and we can classify the current pixel $[i, j]$ as recto writing.

The next step in the segmentation process is the classification of bleed-through pixels. Since the correlation of the square areas of the recto and verso around the pixel is used, we must first define a size of the correlation neighbourhood, and then the value of T_c. The size of the correlation neighbourhood depends on the scanning resolution of the document, but in most cases falls in between 15 and 25. A value of $N = 15$ was used in all the results shown in the appendices.

The correlation of the two areas results in a ratio that describes the similarity between the two sides. The higher the ratio is, the more similar the two sides are. Using the idea that pixels that are bleed-through must come from an area that is similar on both the recto and flipped verso, means that pixels that are bleed-through must have surrounding areas with high correlation.

If X is greater than T_{bt}, then we know that the darkest pixel in the recto neighbourhood is lighter than the darkest pixel in the verso neighbourhood, and that this

pixel *might* be bleed-through. At this point, the correlation of the square areas of the front side and back side around the pixel is used. If this correlation is greater than T_c, then the pixel is classified as bleed-through. Otherwise, the pixel is classified at bleed-through and writing.

In more mathematical terms,

$$Class[i,j] = \begin{pmatrix} 1 & X \leq T_{bt} \\ 2 & X > T_{bt} \; and \; \mathbf{corr2}(f_r[i-k, j-l], f_v[i-k, j-l] > T_c \\ 4 & X > T_{bt} \; and \; \mathbf{corr2}(f_r[i-k, j-l], f_v[i-k, j-l] \leq T_c \end{pmatrix}$$

where $(k,l) \epsilon W_N$, and $W_N = \{(k,l) | -\frac{N-1}{2} \leq k \leq \frac{N-1}{2}, -\frac{N-1}{2} \leq l \leq \frac{N-1}{2}\}$

The next step we take is to determine an acceptable value for T_{bt}. First, we can definitely assume that if X is negative, then the pixel is recto writing. We also want to catch pixels of the image that are both recto writing and bleed-through. This is the case where X is a value that may be positive, but close to zero. After some experiments were done across many different areas of different images, it was concluded that a good value to make a distinction at is 0.05. In fact, any value smaller than 0.1 seems to yield good results.

Therefore if the calculated value X is less than 0.1, then the current pixel is taken as recto writing. But if $X > 0.1$ then the correlation must be used. This is why there is a distinction between classes 1 and 4. The processing done for both these categories is the same, but due to the differences in the value of X in these regions, the segmentation is easier to perform when four areas are defined.

The next step is to determine the value of T_c that is acceptable in terms of segmenting the image into proper categories. After some tests were done, it seems that the value of 50 percent correlation is a good value to make the distinction. Fig. 4.2 shows the difference in the segmentation algorithm using different values for T_c.

Upon close inspection of this set of images, one can see that there is more purple around areas of bleed-through and writing. This means that as T_c is increased, the segmentation algorithm classifies more pixels as writing and bleed-through, rather than just bleed-through. It is from these results that a value of 50% was chosen for the value of T_c.

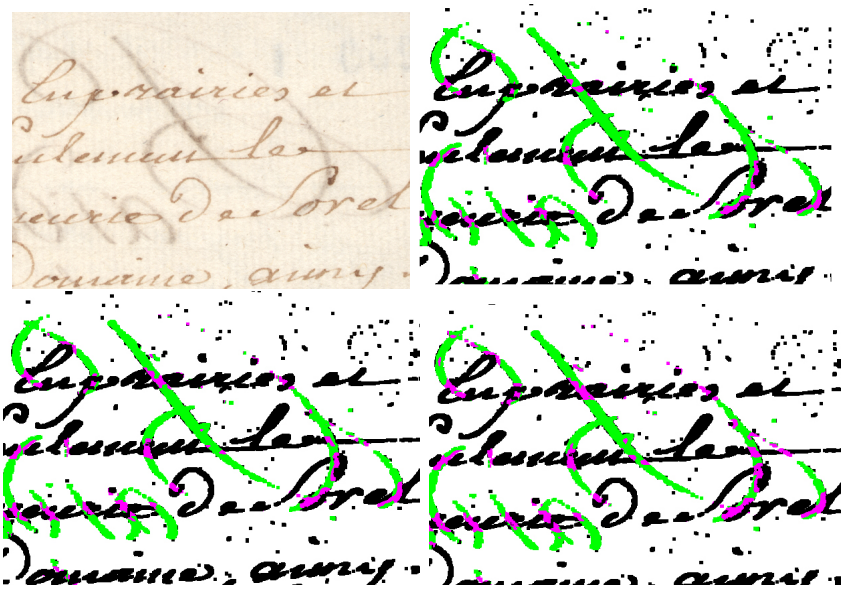

Figure 4.2: Original image (top-left), segmented output with T_c at 50% (top-right), segmented output with T_c at 60% (bottom-left), segmented output with T_c at 70%. (bottom-right)

4.2.3 The Complete Algorithm

Fig. 4.3 shows a flowchart that describes the segmentation algorithm.

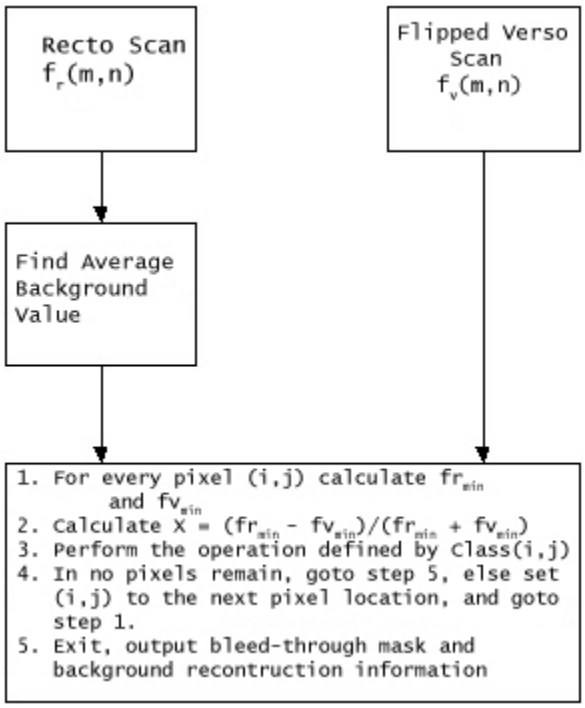

Figure 4.3: Flowchart of the image segmentation routine

Notice that step 5 of the algorithm states that the bleed-through mask and background reconstruction information is to be outputted. This is because of the compression method that we use for this thesis. This will be explained in the next chapter.

4.2.4 Results

The results of this algorithm are shown next. These images are only small portions of the complete document.

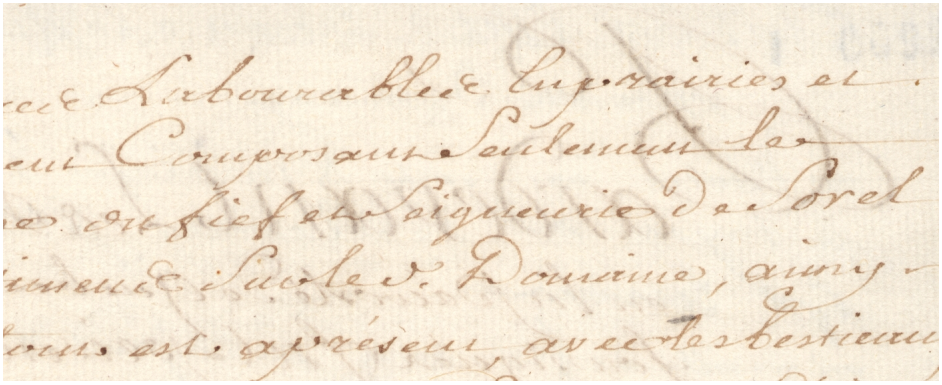

Figure 4.4: Verso of the Bail image before restoration

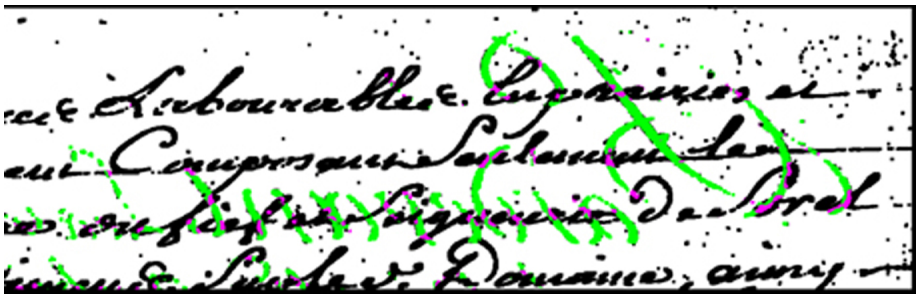

Figure 4.5: Segmentation map of the image shown in Fig. 4.4

Figure 4.6: Recto of the shoemaker image before restoration

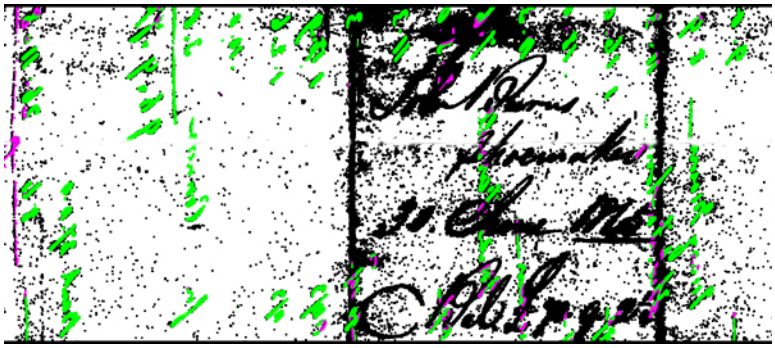

Figure 4.7: Segmentation map of the image shown in Fig. 4.6

One can see that the segmentation algorithm does a good job of segmenting bleed-through from the rest of the document.

4.3 Inpainting

Inpainting, often called retouching, is the technique of removing portions of an image such that the removal is not detectable. As described in [5] the goals of inpainting range from the restoration of damaged paintings and photographs to the removal or replacement of selected objects within a painting or photograph.

Applications of inpainting include the removal of scratches or dust spots in old photographs, the removal of red eye from photographs, and also removing text from photographs (as in postcards). Therefore it is easy to see how image inpainting techniques can be applied for the application of bleed-through removal after the bleed-through has been detected.

Once the bleed-through has been detected using the algorithm described earlier in this thesis, a method for image inpainting can be applied to those areas. It was beyond the scope of this thesis to develop or implement a sophisticated algorithm for inpainting.

4.3.1 Replacement of Pixels Deemed as Bleed-Through

Upon the segmentation of an image into the four major areas defined in Table 4.1, the document background information was analyzed and used to generate an estimate that would be used to fill in the pixels categorized as bleed-through.

There were two methods of pixel inpainting used in this thesis. The first consisted of a basic background average, called the *white paper average* (w_p). This simple calculation provided a very basic estimate of the background. As can be seen in Fig. 4.8 and Fig. 4.9, the restored documents were less visually distracting as the originals were. The second method consisted again of using the background information and the assumption that the background information could be modelled as a Gaussian random process. The mean and standard deviation of the background were calculated, and were then used to generate an estimate of the background. Both these inpainting methods increased the readability of the documents as compared to the original documents. However, looking closely at the images in Fig. 4.8 and Fig. 4.9, one can discern the areas of bleed-through that have been corrected, this is because the areas to be filled in these images were large enough that filling them in with a simple inpainting method would lead to their still being detected visually. Fortunately the restored documents prove more readable than the originals, which leads one to conclude that the simple inpainting methods were satisfactory for the scope of this thesis.

Fig. 4.8 and Fig. 4.9 show the results of applying inpainting to the images shown in Fig. 4.4 and Fig. 4.6 respectively. It is obvious that these results are easier to read than the original images.

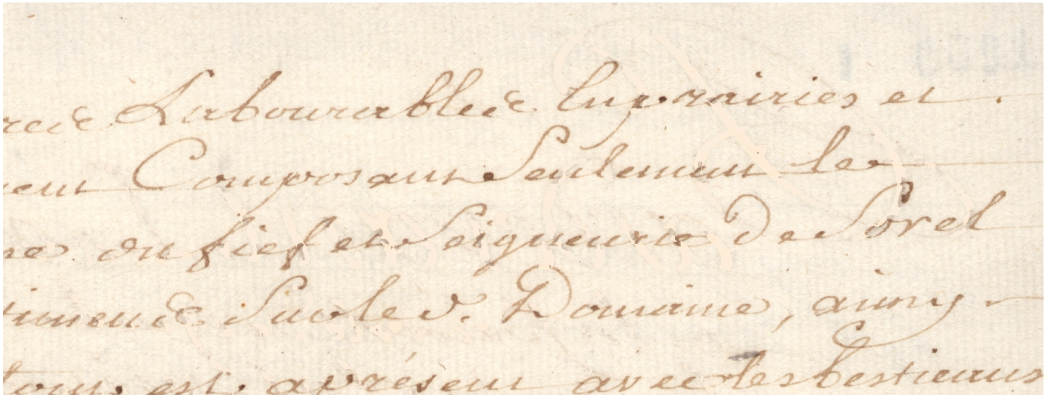

Figure 4.8: The bleed-through corrected version of the image shown in Fig. 4.4

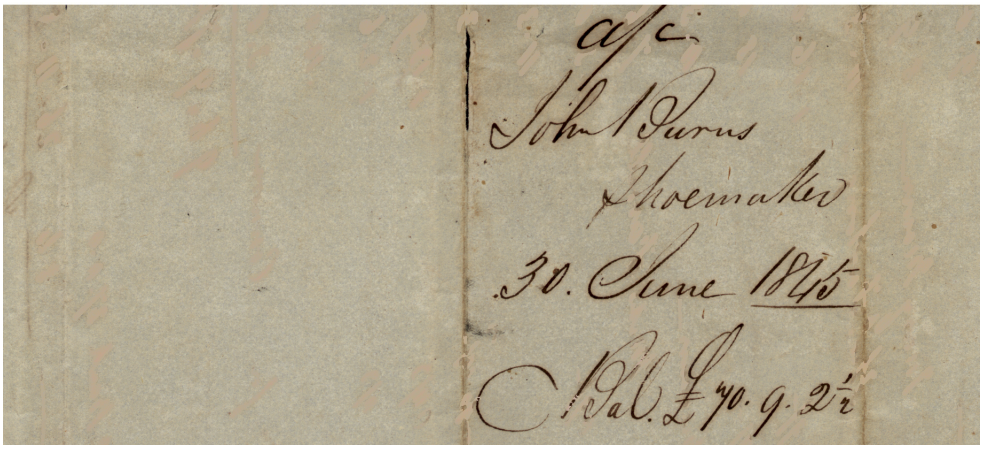

Figure 4.9: The bleed-through corrected version of the image shown in Fig. 4.6

4.4 Summary

To conclude this chapter, the segmentation algorithm developed in this thesis resulted in bleed-through corrected versions of the original documents that were easier to read and understand. Furthermore, the inpainting algorithm also was successful in reducing the amount of bleed-through distortion in the original documents.

Chapter 5

Joint compression of the original and bleed-through corrected images

The next step in the development of an online digital library is making the documents easily accessible across a network. This is where the problem of size comes in. Uncompressed scans of full page documents occupy a large space and pose a problem for quick distribution over networks. The documents we are concerned with have small areas of handwritten text over a homogeneous background and the transition between text and background is usually very sharp. This translates into an image containing high spatial frequencies requiring a high scanning resolution to capture these details.

In this section I describe different methods of compressing the original document, and a method to transmit both the bleed-through corrected image along with the original. A typical document will be scanned at 300 dpi and assuming a page size of 8.5 x 11 inches, we expect a document that is 2550x3300 pixels in size. This translates to an uncompressed document size of around 30 MB. We are interested in sending both the original and corrected images; if we do not apply any compression then we could expect to transmit around 60 MB of information for every page. This problem of joint compression has never been addressed before for this type of application. This section deals mainly with testing mainstream image compression algorithms in order

to compress the original image, and defining a method to jointly compress the bleed-through corrected version of the image along with the original. It is the goal of this chapter to introduce a method of joint compression that will yield better results than individually compressing the two images. Once again, we are interested in providing both the original document and its bleed-through corrected version in case any scholar studying a bleed-through corrected document feels that some important details are missing and could be found on the original. Alternatively, the idea of transmitting the original document alone and performing the bleed-through removal operation at the user's end is presently not an option as the bleed-through removal operation is a complex algorithm that requires a large amount of time to work. Even though the transmitted data would be less than that of the method proposed here, the extra time required to generate the bleed-through corrected version at the user's end would not be worth the bandwidth saved.

I have considered the following recent image formats which are adapted to coding image documents. The first is the JPEG 2000 format, which will eventually be the next release of the widely used JPEG format. I also consider DjVu and LDF, which are two image formats related to JPEG 2000 that were aimed specifically towards document compression. The current JPEG format was also considered.

In order to achieve better compression for the set of the original document and its bleed-through corrected version, another method of transmitting the two images must be found. In this scheme, the original image is compressed using techniques that are widely in use, and enough information about the corrected version is sent along with the original so that the bleed-through corrected version can be assembled in a stand-alone decoder at the user's end. This information consists of the bleed-through mask and a data stream which is an estimate of the background.

The bleed-through mask is a bilevel image in which a high pixel value (the colour white) denotes a pixel in the original document that is bleed-through. This mask image is to be compressed using the JBIG standard, which is a bilevel image compression standard.

The data stream that contains information on the backgound information is a set of values that should provide an estimate of the background for every pixel that is

white in the mask image. This data stream does not have to be compressed, since it is very small in size to begin with; it did not exceed 1 kB in the tests run during this project. However in future revisions, the inpainting portion may be changed and this data may require a compression technique.

The main issue that remains is to choose a suitable compression format for the original image.

5.1 Compression Schemes

In this section, I briefly describe the compression schemes that were evaluated for this project. I only describe the basic ideas of each compression scheme, and outline the basic differences between each of them. For a general overview of data compression and its applications to images, please refer to [10].

5.1.1 JPEG 2000

JPEG 2000 is a new image compression algorithm being developed by the ISO. It supports both lossless and lossy coding of both greyscale and colour images.

The purpose of JPEG 2000 was to address weaknesses in the original JPEG standard (which is in wide use today), and to include new features that were not available in the old JPEG. These are

1. to provide for both lossy and lossless coding;
2. to provide better image quality at lower bit rates;
3. to support a more flexible file format;
4. to lessen the computation and memory requirements of systems it runs on.

The JPEG 2000 codec is based on wavelet coding techniques; a full discussion is provided in [11]. Basically, to code an image, the codec first passes the image to be coded through a preprocessor, which makes the pixels within the image be centered

about zero, (for example pixels may range from -128 to 127 after this process, as opposed to 0 to 255.)

This data is then passed to a forward inter-component transform. The inter-component transforms operate across the components of the image (components being RGB for example). There are only two inter-component transforms defined for the baseline JPEG 2000 codec, and both of them essentially map image data from the RGB colour space to the YCrCb colour space.

Next the data is passed to an intra-component transform. Here transforms are applied to each component of the image data. This is where the wavelet transform takes place.

After the intra-component wavelet transforms have been applied, the resulting data is sent to a quantizer. This allows greater compression to be achieved and is one source of information loss. Two types of quantizers are available here, real and integer. In integer mode the quantizer step size is set to 1, and therefore no quantization is done, and the compression is lossless. Real mode quantization involves step sizes that are chosen according to rate control, this is where the coding loses information in lossy coding. The quantized data then undergoes a lossless binary code assignment procedure prior to storage and/or transmission.

5.1.2 DjVu and LDF

DjVu is a new method of image compression that was designed specifically with documents in mind. DjVu was initially a creation within AT&T labs, but has since been purchased and brought to market by a company named LizardTech (www.lizardtech.com).

The main idea behind DjVu is to separate the document into 3 different images: a background image, a foreground image, and a mask. The original image can then be reconstructed from these images.

The 3 images that compose a full image file are:

1. Mask: This is a bitmap which indicates whether the corresponding pixel in the image is a foreground or background pixel. This layer identifies all the text and drawings, and is a bi-level image coded at 300 dpi, it is encoded using the

JBIG2 standard, which is a compression codec for bi-level images.

2. Background: This contains all the pictures in the document. It is sampled at 100 dpi with 24 bits/pixel, and is encoded using a wavelet based compression algorithm.

3. Foreground: This layer contains all the color information present in the document. This layer is sampled at 25 dpi with 24 bits/pixel and is also encoded using the same wavelet based algorithm as the background layer.

After these steps, all the layers are compressed using an entropy coder called the ZP-coder. The ZP coder is a type of arithmetic coder; arithmetic coding is a method for compressing a string of symbols so as to approach the information theory limit.

LDF is another compression method for scanned documents only. LDF is short for LuraDocument Format, and is provided by Algo Vision Luratech. The main ideas in LDF are similar to DjVu, the only differences being their bi-level coders.

The 1-bit coder in DjVu is a common implementation of JBIG2 (done by AT&T), whereas LDF uses it's own 1-bit coder.

The Wavelet coder in DjVu is the IW44 coder that was designed by Lizardtech, whereas the one used in LDF is called LWF (for LuraWave Format) and is the wavelet implementation used in the JPEG2000 codec.

Lastly, DjVu and LDF also differ in their segmentation algorithms. These segmentation algorithms are used to segment the image into the three parts described above.

A complete description of the DjVu compression method is found in [12], while a complete description of the LDF method is found in [13].

5.1.3 PNG

The original reason for the development of PNG was for the replacement of GIF (GIF is a legally licensed compression algorithm that uses the LZW compression standard. Unisys now holds the LZW patent), but now provides more features than GIF does. In addition to providing the features that GIF does, PNG also provides the following:

1. Can represent true colour images up to 48 bits / pixel
2. Greyscale images up to 16 bits / pixel
3. Full alpha channels

As PNG is a replacement for the GIF standard, the compression algorithm that PNG uses is a variant of LZ77 that is used in many ZIP applications. LZ77, unlike LZW is not under any patent, which makes the implementation and distribution of applications that use the PNG coder much more attractive to the programmer (and the end user) since the intellectual property used for the codec is without any royalty.

The LZ (named for Lempel and Ziv, the people who developed the algorithm) family of coders are dictionary based coders. What that means is that they replace portions of the input with a symbol that characterizes it. Dictionary coders start with a blank dictionary, and grow in size (and data) as they progress through the input. The repetitive occurrences of a string in the input stream are replaced by a pointer to the former occurrences of the same string. Usually the pointer is smaller (in length) than the string that it replaces therefore data compression occurs, though sometimes for non-repetitive data streams, an expansion can occur.

LZ coders can get very good compression on ASCII messages (due to the repetitive nature of languages) and are also good for the compression of text or program files (variants of LZ77 are used in gzip, zip and other file archiving formats). LZ performance on compressing images is generally not up to par with lossy coders such as JPEG (as is seen in GIF images), though this project will evaluate this new variant of LZ coding against the new lossless JPEG provided in JPEG 2000.

A full description of PNG can be found in [14].

5.2 Performance

During the benchmarking of these codecs, the most important criterion for performance of the codecs was the quality of the resulting output. Visually, the output of any codec must be very near to the original image. For the tests, I used the following software:

1. JPEG 2000: JasPer codec, a freeware reference implementation of the JPEG2000 part 1 (ISO/IEC 15444-1). It resulted from a collaborative effort between Image Power, Inc. and the University of British Columbia.

2. JPEG: IrfanView v3.80, freely downloadable at http://irfanview.tuwien.ac.at

3. DjVu: LizardTech Solo v3.1. Available for download for demonstration purposes.

4. LDF: Irfanview v3.80

5. PNG: Irfanview v3.80

6. GIF: Irfanview v3.80

The following two images were used to benchmark the compression schemes listed above.

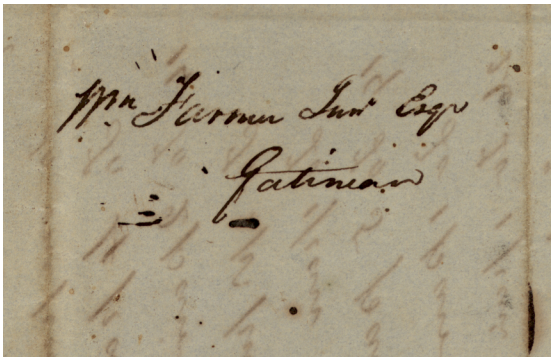

Figure 5.1: Recto of the shoemaker image before any compression (1380 kB)

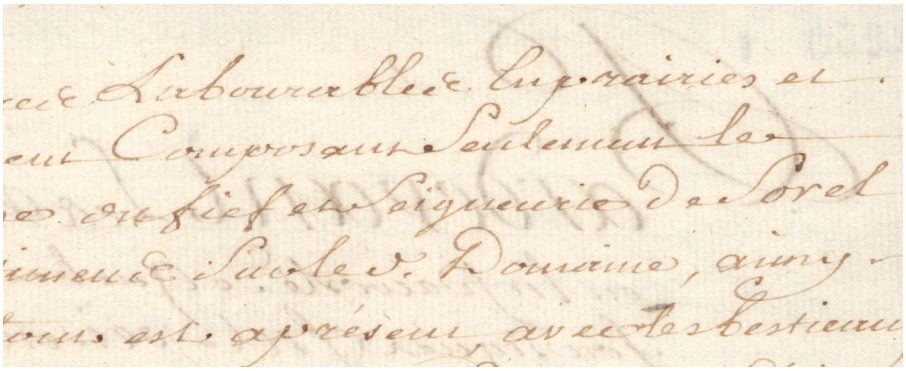

Figure 5.2: Recto of the bail image before any compression (2325 kB)

The image shown in Fig. 5.1 was put through the different compressors, and the following table of data was compiled. Original file size was 1380 kB.

Codec Type	Size (kB)	Compression ratio	PSNR (against original)
Original	1380	1:1	Infinite
GIF	284	4.86:1	42.70
JPEG 2000 Lossy 10:1	138	10:1	42.58
JPEG 2000 Lossy 30:1	46	30:1	38.44
JPEG 2000 Lossy 50:1	28	50:1	36.73
JPEG 2000 Lossless	662	2.1:1	Infinite
JPEG (Q=40)	25	55.2:1	36.16
JPEG (Q=65)	42	32.9:1	37.75
JPEG (Q=80)	64	21.6:1	39.21
PNG	724	1.91:1	43.87
LDF	14	98.6:1	30.13
DjVu	4	345:1	25.90

Table 5.1: Results of compressing the shoemaker image

The image shown in Fig. (5.2) was put through the different compressors, and the following table of data was compiled. Original file size was 2325 kB.

Codec Type	Size (kB)	Compression ratio	PSNR (against original)
Original	2325	1:1	infinite
GIF	446	5.2:1	44.69
JPEG 2000 Lossy 10:1	232	10:1	43.79
JPEG 2000 Lossy 30:1	77	30:1	38.58
JPEG 2000 Lossy 50:1	47	50:1	36.74
JPEG 2000 Lossless	881	2.64:1	Infinity
JPEG (Q=40)	46	50:1	36.60
JPEG (Q=65)	75	31:1	38.22
JPEG (Q=80)	111	20.9:1	39.65
PNG	1164	2:1	44.10
LDF Low	25	93:1	33.62
LDG High	112	20.75:1	38.41
DjVu	8	390:1	29.89

Table 5.2: Results of compressing the Bail image

The above data is not meaningful unless accompanied by the images that it is associated to. However, the above tables do help to classify the compression schemes relative to themselves. Solely from the data shown in the above tables we can safely say that the LDF (Low) and DjVu formats result in the highest compression rates. Unfortunately, they also have the lowest PSNR's.

The images shown in Fig. 5.3 and Fig. 5.4 show the results of compressing the image shown in Fig. 5.2 using the LDF (Low) and DjVu codecs.

The images shown in Fig. 5.5 and Fig. 5.6 show the results of compressing the image shown in Fig. 5.1 using the LDF (Low) and DjVu codecs.

These images are not very close in visual detail to the originals even though the DjVu and LDF image compression formats are designed with documents in mind. The documents compressed using the LDF (Low) codec are not strikingly different than the originals, but if viewed closely, one can see that the compressed document is blurrier than the original. Both the LDF (Low) and DjVu codecs seem to compress the documents far too much for our purposes. This problem may be attributed to the demonstration versions of the software that we have available for use.

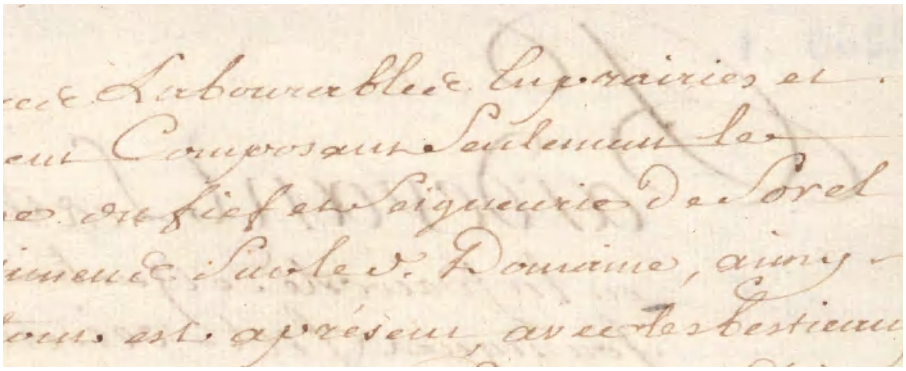

Figure 5.3: Bail image compressed with the LDF (Low) codec

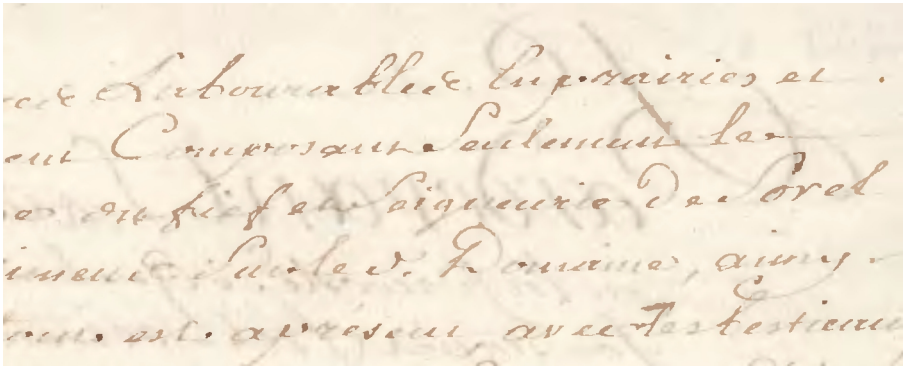

Figure 5.4: Bail image compressed with the DjVu codec

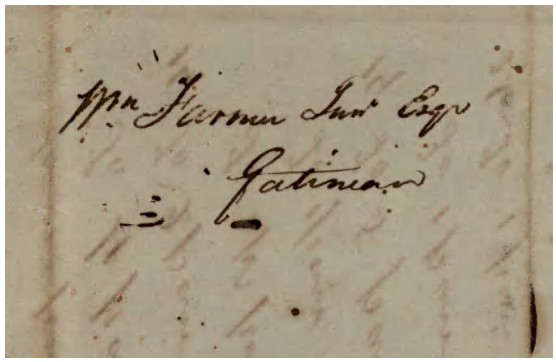

Figure 5.5: Shoemaker image compressed with the LDF (Low) codec

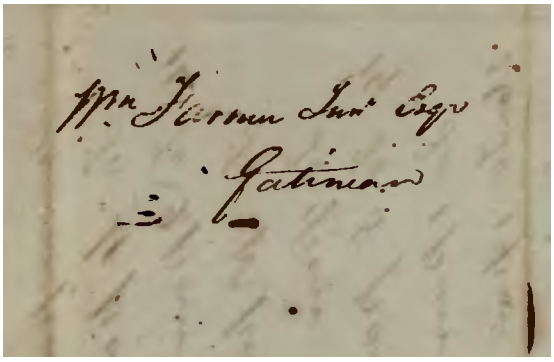

Figure 5.6: Shoemaker image compressed with the DjVu codec

In addition to PSNR, visual image quality must also be considered. The ideal codec for this application must result in an image that is visually indistinguishable from the original, with the highest compression ratio. Upon close examination of the images produced in obtaining the values for Table 5.1 and 5.2, it was found that very slight distortions began to appear in the images when the PSNR was in the vicinity of 35. Eliminating all codecs that resulted with PSNR of less than 35, all that remained was to choose the codec with the highest compression ratio.

With the elimination of DjVu and LDF (Low) as viable formats, we are left with JPG, JPG2000, PNG, GIF, and LDF (Good). Upon further consideration, it was deemed that either the JPG2000 Lossy codec with a compression factor of 30:1 or JPEG with a quality factor of 65 were appropriate codecs for our needs. These two codecs both produced images with a PSNR around the value of 38, and had the highest compression ratios.

Table 5.3 shows details of compressing the original and bleed-through corrected version of the Bail image, including the segmentation masks and the background estimates.

We see that our method of joint compression reduces the amount of data down to 519 kB from the theoretical 1004 kB (if both images were compressed individually) yielding a compression ratio of 1.93:1.

Data	Original Size (kB)	Compressed Size (kB)	Compressor
Original Image	16971	502	JPEG (Q=65)
Segmentation Map	161	16	JBIG
background estimate	1	1	None
Total Size	17133	519	

Table 5.3: Comparison of sizes before and after compression of the Bail Image

Table 5.4 shows details of compressing the original and bleed-through corrected version of the shoemaker image, including the segmentation masks and the background estimates.

Data	Original Size (kB)	Compressed Size (kB)	Compressor
Shoemaker Image	20252	624	JPEG (Q=65)
Segmentation Map	145	18	JBIG
background estimate	1	1	None
Total Size	20398	643	

Table 5.4: Comparison of sizes before and after compression of the shoemaker Image

We see that our method of joint compression reduces the amount of data down to 643 kB from the theoretical 1248 kB (if both images were compressed individually) yielding a compression ratio of 1.94:1.

The compression used for the bi-level mask is JBIG, which is a lossless codec for the compression of bi-level images. JBIG2 will eventually replace this standard, but unfortunately a freeware implementation of JBIG2 was not available for use in this project. The image shown in Fig. 5.7 shows a the bi-level segmentation map of the image in Fig. 5.2, which is compressed using the JBIG standard.

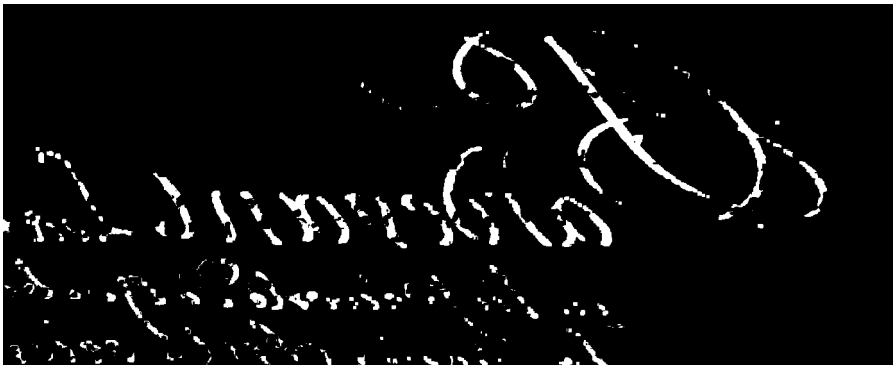

Figure 5.7: Segmentation map of the image shown in Fig. 5.2

The compressed sizes resulting from the joint compression method presented here show that this method is more efficient at transferring both the original document and its bleed-through corrected version than to just compress and transmit both images independently. The user can reconstruct the bleed-through corrected document using a stand alone decoder at their end, this can be done within seconds.

The decoded outputs of the full images are shown in Appendix B.

Chapter 6

Conclusion

6.1 Summary of Work

This thesis has presented an approach for the restoration of documents that are degraded due to bleed-through. The technique developed has been shown to be successful in restoring such documents.

The overall subject of restoration was divided into three parts:

1. Registration.
2. Segmentation.
3. Inpainting.

In order to test the efficiency of the registration of the two images, the difference test (total squared error) was applied across the images, and a performance metric was evaluated. The six point affine transformation technique is able to adjust for any translation, rotation and slight distortion that may have occurred during the scanning of the document.

The quality of segmentation of the images was left up to the eye. Visual inspection of the segmented images was a good enough performance metric in judging the ability of the algorithm in distinguishing areas of bleed-through from other areas. Overall

the algorithm provided in this thesis delivered results that were significantly better than previous results in this field.

The quality of the inpainting of the segmented images was also left up to the eye. In this case, it makes better sense to judge if a restored image is better than the original just by looking at it, since what we're really interested in is to make the document image easier to read. The inpainting technique shown in this thesis is a crude method that can definitely be improved upon in further work.

In addition to document restoration, this thesis also examined different methods of compressing the document for distribution over a network. Although we found that the JPEG 2000 codec was the ideal candidate for compressing the images, the task that we were more interested in was to find a way in which both the restored image and the original could be transmitted together. The simplest method found was to send the following three components:

1. The complete original image, compressed using either the JPEG 2000 or JPEG codecs;

2. a bilevel map image that shows which pixels are bleed-through, compressed using the JBIG format;

3. a data stream that contains information on how to generate an estimate of the background. This information is used to replace any pixels that are bleed-through in the original image.

The code developed in this project can work on images that are full colour (24-bit) or greyscale (8-bit). Most of the images tested were colour images.

All the programming done in this thesis was done using MATLAB, so that the actual processing time for performing a restoration could be drastically reduced once the algorithm is ported over to C/C++.

6.2 Contributions of this Thesis

Even though we mainly used only two images (the 'bail' document and the 'shoemaker' document), we can assume with a certain sense of confidence that the conclusions we arrived at in this thesis can extend to any documents that are in the same class as these.

The main contributions of the thesis are as follows.

6.2.1 Registration

I modified the registration algorithm discussed in [4] and [15]. The algorithm presented in this thesis produced a better alignment between the recto and flipped verso of a document image, in a lower amount of time than the previous registration algorithms.

6.2.2 Segmentation

I modified the segmentation algorithm discussed in [3] so that it took into consideration the correlation between the recto and flipped verso images of a document affected by bleed-through. Previous segmentation algorithms relied only on pixel intensities in order to classify pixels within a document image degraded with bleed-through.

6.2.3 Inpainting

I introduced two simple methods for replacing pixels affected by bleed-through. The proposed methods were effective in obtaining a good estimate of the background, but were not good enough so that the images looked non manipulated. It was beyond the scope of this thesis to produce a new inpainting algorithm, or to adapt current more complex inpainting schemes for this application.

6.2.4 Joint Compression

I introduced an efficient method of transmitting both the original document and its bleed-through corrected version. The topic of joint compression has not been addressed before in terms of document bleed-through removal. The approach taken in this thesis was shown to produce an efficient method of compression as compared to the individual compression of both the original document and its bleed-through corrected version. It was beyond the scope of this thesis to define a new document compression scheme, though the comparison of several current compression schemes showed that JPEG2000 and JPEG were viable codecs for this application.

6.3 Future Work

The work presented in this thesis is a step in the right direction for the restoration of document suffering bleed-through. However, there is much room for further improvement. Two specific aspects requiring additional work are segmentation and inpainting.

6.3.1 Segmentation

Better segmentation can be achieved if we develop better models of the relationship between the recto and flipped verso of a document affected with bleed-through. The approach described in this thesis did not have a mathematical model of a document in order to perform the segmentation, and instead relied on pixel intensities and correlations in regions affected by bleed-through.

6.3.2 Inpainting

The inpainting methods described in this thesis are very basic. Future work may involve adopting methods for local inpainting which will produce final documents that are easier to read. These should be based on more sophisticated models of the background in the vicinity of the bleed-through being eliminated.

Bibliography

[1] G. Sharma, "Show-through cancellation in scans of duplex printed documents," *IEEE Transactions on Image Processing*, vol. 10, no. 5, pp. 736–754, 2001.

[2] K. Knox, "Show-through correction for two-sided documents." United States Patent 5,832,137, Nov. 1998.

[3] E. Dubois and A. Pathak, "Reduction of bleed-through in scanned manuscript documents," *Proc. IS&T Image Processing, Image Quality, Image Capture Systems Conference (PICS2001)*, pp. 177–180, 2001.

[4] Y. Bienvenue, "Registration of two sided documents suffering from bleed-through." CSI4900 Project Report presented to Dr. Eric Dubois, University of Ottawa, Canada.

[5] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Siggraph 2000, Computer Graphics Proceedings* (K. Akeley, ed.), pp. 417–424, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[6] T. F. Chan and J. Shen, "Mathematical models for local nontexture inpaintings," *SIAM Journal on Applied Mathematics*, vol. 62, no. 3, pp. 1019–1043, 2002.

[7] L. G. Brown, "A survey of image registration techniques," *ACM Computing Surveys*, vol. 24, no. 4, pp. 325–376, 1992.

[8] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ, USA: Prentice Hall, 2nd ed., 1991.

[9] The Mathworks, Inc., "Image processing toolbox user's guide,"

[10] K. Sayood, *Introduction to Data Compression*. San Fransisco, CA: Morgan Kaufmann, 2nd ed., 2000.

[11] M. D. Adams, "The JPEG 2000 still image compression standard." Dept. of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada, V6T 1Z4.

[12] L. Bottou, P. Haffner, P. G. Howard, P. Simard, Y. Bengio, and Y. LeCun, "High quality document image compression with DjVu," *Journal of Electronic Imaging*, 1998.

[13] K.U. Barthel, S. McPartlin and M. Thierschmann, "New technology for raster document image compression," *Proc. SPIE*, vol. 3967, pp. 286–290, December 1999.

[14] G. Randers-Pehrson and et. al., "PNG (portable network graphics) specification." http://www.w3.org/TR/REC-png, October 1996.

[15] A. Pathak, "Restoration of documents with bleed-through distortion," 2001. Master's Thesis, University of Ottawa, Canada.

Appendix A

Documents used in this thesis

Two documents were used while researching during this project. The first I call 'bail', which is a farm lease for the domain of the Sorel Seigneury. I call the document 'bail' because it is the french word for lease. This document was written on January 29, 1759 and is accessible through the National Archives of Canada Call number MG18-H54.

The second document is called 'shoemaker', simply because the word shoemaker is visible on one side of the document. This document was also provided by the National Archives of Canada. Unfortunately I was unable to find any more information on this document.

A third document was introduced late in the writing of this thesis, it is a sixteenth century manuscript of *Le Chevalier au Lion* written around 1520. This document was provided by the Bibliotheque National de France.