

# Flexible Text Recovery and Recognition from Degraded Historical Typewritten Documents

A THESIS SUBMITTED IN ACCORDANCE WITH THE REQUIREMENTS OF THE  
UNIVERSITY OF LIVERPOOL  
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY  
IN THE FACULTY OF SCIENCE

2007

By  
Celia Casado Castilla  
Department of Computer Science

**First Supervisor:** Apostolos Antonacopoulos, University of Salford, UK  
**Second Supervisor:** Michael Fisher, University of Liverpool, UK  
**Internal Examiner:** Bernard Diaz, University of Liverpool, UK  
**External Examiner:** David Elliman, University of Nottingham, UK

# Contents

<b>Acknowledgements</b>	<b>6</b>
<b>Abstract</b>	<b>7</b>
<b>1 Introduction</b>	<b>8</b>
1.1 Overview . . . . .	8
1.2 Background . . . . .	9
1.2.1 Document Image Analysis . . . . .	10
1.2.2 Analysis of historical documents . . . . .	16
1.2.3 Typewritten Documents . . . . .	17
1.3 Key Contributions of the Research . . . . .	18
1.4 Structure of the Thesis . . . . .	20
<b>2 Document Image Analysis</b>	<b>21</b>
2.1 Introduction . . . . .	21
2.2 Image Pre-Processing . . . . .	21
2.2.1 Skew Correction . . . . .	22
2.2.2 Thresholding . . . . .	24
2.2.3 Thresholding of degraded documents . . . . .	28
2.2.4 Image Enhancement and Restoration . . . . .	35
2.2.5 A Note on Measuring Image Quality . . . . .	47
2.3 Summary and Conclusions . . . . .	49
<b>3 Optical Character Recognition (OCR)</b>	<b>51</b>
3.1 Introduction . . . . .	51
3.2 Character Segmentation . . . . .	54
3.3 Feature Extraction . . . . .	54
3.3.1 Template Matching . . . . .	55
3.3.2 The Distribution of Points . . . . .	56
3.3.3 Transformation and Series Expansions . . . . .	56

3.3.4	Structural Analysis . . . . .	57
3.3.5	Shape Description and Recognition . . . . .	60
3.3.6	Feature Extraction from Greyscale images . . . . .	61
3.4	Classification . . . . .	61
3.4.1	Decision-theoretic methods . . . . .	62
3.4.2	Syntactic pattern recognition . . . . .	63
3.4.3	Combination of classifiers . . . . .	64
3.5	OCR errors description . . . . .	64
3.5.1	Imaging Defects . . . . .	64
3.5.2	Similar Symbols . . . . .	66
3.5.3	Punctuation symbols . . . . .	67
3.5.4	Typography . . . . .	67
3.6	Measuring Recognition Accuracy . . . . .	68
3.7	Alternatives to OCR . . . . .	70
3.8	Summary and Conclusions . . . . .	71
<b>4</b>	<b>Flexible Text Recovery</b>	<b>72</b>
4.1	Introduction . . . . .	72
4.2	Overview of the Method . . . . .	73
4.3	Character Position Location . . . . .	74
4.3.1	Projection profiles . . . . .	74
4.3.2	Projection profile analysis . . . . .	79
4.3.3	Fixed-grid segmentation . . . . .	82
4.4	Character Localisation . . . . .	85
4.5	Combination Thresholding . . . . .	86
4.5.1	Cautious Ternarisation (CT) . . . . .	87
4.5.2	Aggressive binarisation (AB) . . . . .	93
4.5.3	Results Combination . . . . .	94
4.5.4	Results and discussion . . . . .	96
4.6	Flexible Stroke Refinement . . . . .	97
4.7	Results and Conclusions . . . . .	105
<b>5</b>	<b>Character Recognition</b>	<b>107</b>
5.1	Introduction . . . . .	107
5.2	Weighted Image Matching (WIM) . . . . .	110
5.2.1	Superimposition . . . . .	111
5.2.2	Measuring the degree of coincidence . . . . .	112
5.2.3	Recognition Confidence . . . . .	117
5.3	Feature Extraction . . . . .	122



5.3.1	Character Size . . . . .	123
5.3.2	Character Weight . . . . .	126
5.3.3	Vertical and Horizontal Maxima . . . . .	128
5.3.4	Other Features . . . . .	129
5.4	Classification . . . . .	132
5.4.1	Results and discussion . . . . .	137
5.5	Use of Uncertain Strokes . . . . .	138
5.6	Results . . . . .	140
5.7	Summary and Conclusions . . . . .	141
<b>6</b>	<b>Results</b>	<b>143</b>
6.1	Introduction . . . . .	143
6.2	The Data Set . . . . .	143
6.3	Measuring Recognition Errors . . . . .	145
6.4	The Original Images . . . . .	148
6.5	Flexible Text Recovery Results . . . . .	150
6.5.1	Comparison to Otsu's method . . . . .	155
6.5.2	Comparison to Sauvola and Pietaksinen's method . . . . .	158
6.5.3	Comparison to Gatos et al. method . . . . .	159
6.6	Comparison of Character Recognition Results . . . . .	161
6.7	Conclusion . . . . .	166
<b>7</b>	<b>Summary and Conclusions</b>	<b>169</b>
7.1	Introduction . . . . .	169
7.2	Conclusions about the research . . . . .	170
7.3	Limitations and Implications . . . . .	172
7.4	Further research . . . . .	173
	<b>Bibliography</b>	<b>175</b>
	<b>List of Tables</b>	<b>186</b>
	<b>List of Figures</b>	<b>191</b>

# Acknowledgements

I would like to thank my supervisors Dr. Apostolos Antonacopoulos for his valuable advice and guidance and Prof. Michael Fisher for his unconditional help. Additionally, I am very grateful for the advise received by my examiners, Prof. Dave Elliman and Dr. Bernard Diaz. I would also like to thank my fellow office mates and friends, David Bridson, Dimosthenis Karatzas, Christos Papadopoulos, Ian Guthrie, George Papoulakis, Maryam Rahnemoonfar, Stefan Pletschacher, Belen Gomez and Carmen Fernandez for the great office talks and the good times. I thank my family for being there for me and support me in my studies. And last, but not least, I would like to thank my boyfriend Dirk Walther for believing in me, for his incredible motivational skills and for his infinite patience and care.

## Abstract

The work presented here proposes a novel method for extracting and recognising text from highly degraded historical typewritten documents. The method is based on the enhancement of each character individually to be able to adapt to its individual features. A combination of methods is used for thresholding. A novel *cautious ternarisation* method is applied to each character separating the background and foreground pixels that are confidently identified from the remaining uncertain pixels. The result of the cautious ternarisation method is then combined with an *aggressive binarisation* thresholding method creating a three level image. The output image maintains grey level pixels in key areas where a character might be broken or wrongly connected. The key areas are then reduced by a novel *refining algorithm* that locates the most conflictive pixels. These key areas are then used during recognition. The recognition process consists of a *template-matching algorithm* that emphasises the size and the distance of the uncertain connected components. To aid the template-matching algorithm, a *flexible classification tree* is used to reduce the number of comparisons by extracting four simple but robust features from the refined characters: the size, the weight and the number of horizontal and vertical maxima in the character profiles. The recognition rates obtained from this customised OCR have improved upon those obtained by two commercial OCRs by up to almost 30% reaching 94% recognition.

# Chapter 1

## Introduction

### 1.1 Overview

The number of documents now stored permanently is immense and continues to increase every day. Millions of handwritten, typewritten, and printed pages are stored worldwide. Historical documents have been preserved through the years, or even centuries, and most are of enormous importance. They contain information about our past, and so, tell our history.

The study of, and general access, to historical documents enriches our knowledge about our culture, past civilizations and even our family paths. Over recent years, the digital conversion of archives and libraries has increased, aiming to promote their study and understanding, distribution and easy access to the public. The main reason to digitise is to improve access and help preservation. By digitizing their collections, cultural heritage institutions, governmental organizations and libraries can make information accessible that was previously only available to a select group of researchers.

Ideally, document images are taken and stored in online databases. Then the text and graphics are extracted and recognized to allow rapid searches by users anywhere and at anytime. Unfortunately, this is not the case for all documents. Due to the age of the ink and paper, storage conditions and/or production quality, some documents fail to be recognized by current technologies. Although off the self Optical Character Recognition (OCR) software can confidently recognise clean typewritten or machine-printed characters (over 99% recognition rate [72]) and clean hand-printed documents (80-90% [72]), when

the documents are exposed to degradation, the recognition rate drops radically. Most of the errors in recognising degraded documents are due to touching or broken character strokes as a result of incorrect binarisation and other factors, such as noise or blur.

The main interest of this work is historical typewritten documents produced during the 20th century. Due to the nature of their creation, like carbon copies and/or a poor preservation condition, up to date systems cannot correctly recognise them. Most of these documents are stored in archives and libraries and contain unique information. Some OCR tools, used to improve recognition rates, such as the contextual use of dictionaries, are of no use in these documents since they contain names of people or places, or are even written in multiple languages.

The main aims of this research are to segment, identify and recover text from highly degraded typewritten historical documents by enhancing and recognising the character images in a novel individual and flexible way. There is a significant need for a system that can deal with the degradation in such documents to extract useful information for further analysis. The result of this research was such a system by which the degraded characters are extracted, thresholded and restored providing with a recognised and improved version of the document. The approach chosen for this purpose is the individual enhancement of the characters, in order to adapt to their different printing intensities and degradation, which is characteristic of typewritten documents. After the character images are extracted, they are refined and recognised. Recognition is performed using a weighted image matching algorithm. To reduce the number of matchings required for the recognition, a probability tree was created by extracting four robust features from the characters.

## 1.2 Background

The following will broadly describe the image analysis process in the context of documents. Document image analysis is a sub-area of image analysis and therefore it inherits all the image analysis techniques and processes but also incorporates new or modified ones in order to accommodate the document processing needs.

## 1.2.1 Document Image Analysis

The objective of document image analysis (DIA) [44, 61], is to extract and recognize text and graphics in a document image. Unlike image analysis where the scope is to process or recognise any kind of image (industrial inspection, medical imaging, etc) document image analysis focuses on the recognition and processing of patterns that appear in text or graphics. The readability of documents can be interpreted in at least two different ways: human and machine readability. A document image is often human readable but not always machine readable, as optical character recognition (OCR) software is much more sensitive to degradation than a human being. If the quality of a document is so low that a human cannot recognize what is written on it, then the document is declared 'unreadable' and the information can be considered lost.

In order to prepare the document and achieve better machine readability the document is subject to pre-processing and different kinds of enhancement before the recognition stage.

Below, a general overview of the steps of document image analysis will be given. In Figure 1.1 a diagram of the typical sequence of steps in document analysis and understanding is shown (from [3]).

### 1.2.1.1 Digitisation

Generally, the first step in document analysis consists of the digitisation of the document page. This stage is also known as 'data capture' or 'image acquisition'.

Data capture of a paper document is usually performed by optical scanning or by using a digital camera. In the case of historical documents, delicate documents might be photographed, while others that are better preserved may be scanned.

Different applications require different scanning approaches. The scanning settings for historical documents have to be satisfactory for the later processes. Due to the nature of the documents, rescanning might be very costly. A low scanning resolution would decrease the level of detail available for subsequent stages. Most character recognition systems require a minimum scanning resolution to properly work. In most applications, a resolution of 300 dpi is sufficient.

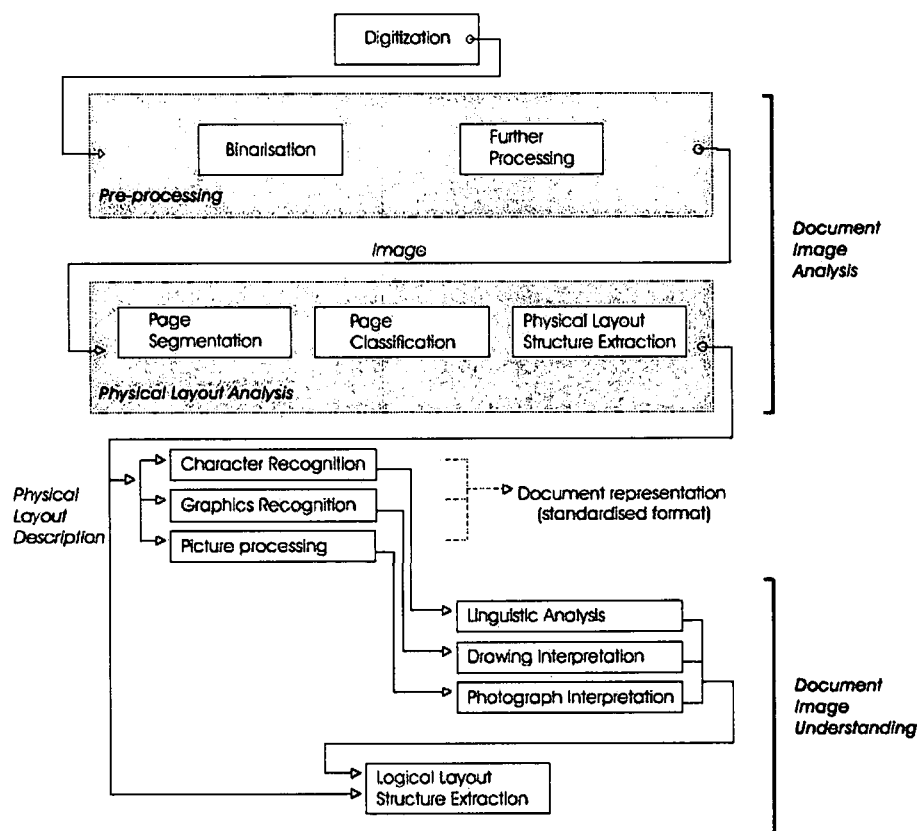


Figure 1.1: Document Image Analysis Diagram [3]

During the digitisation process, noise, skew, shadows or other scanning artifacts may corrupt the image. The noise can appear due to dirt in the scanning glass and skew may be introduced by an incorrect positioning of the paper on the scanner. A lens may deform the image by altering text lines at the edges to a curve or changing disproportionately the size of some characters. Illumination problems during scanning may produce shadows and very bright areas in the image, damaging the text and in some cases rendering it unreadable. The thickness of the characters may be also altered due to irregular illumination during the scanning.

Similar artifacts may appear when obtaining the image with a digital camera. The positioning of the camera over the paper may produce different text sizes between the top and bottom of the page if tilted. The use of a bright flash or an uneven focus might also alter the image quality.

The effects of these artifacts may lower the quality of the documents affecting the performance of subsequent stages such as character recognition.

### 1.2.1.2 Pre-processing

The aim of the pre-processing stage is to prepare the image for further analysis towards understanding and interpretation.

Image pre-processing can be described as taking one array of pixels as input and producing another array of pixels as output, representing, in some way, an improvement or enhancement to the original array. Pixel-level processing is also known as *image enhancement*.

Pre-processing techniques include binarisation plus some enhancement procedures, such as skew correction and denoising. Binarisation, achieved by thresholding, consists of reducing a grey-scale or color image to a binary image.

The output of the thresholding operation is a binary image whose one state will indicate the foreground objects, such as printed text, and the other state will correspond to the background. For that reason it can also be used as a segmentation technique. Thresholding is a crucial step towards recognition since most OCR systems work on binary images. Choosing a correct thresholding technique for a particular image is a difficult task and no one threshold technique has been found to work in every image. An incorrect threshold may worsen the quality of the image and render characters unreadable, for example by being too dark or too faint, or introduce noisy data in the document image.

Noise reduction, or denoising, consists of removing extraneous data from the image. This noisy data may have been introduced to the image during the acquisition process, due to incorrect thresholding or may be in the document before scanning. This noise should be identified and removed as the character recognition system would attempt to recognise it creating recognition errors.

Although documents are created with skew of zero degrees, making text lines perfectly parallel to each other, a skew angle can be introduced to the document image by tilting the document page while scanning or, in typewritten documents, by removing the paper and inserting it again at a different angle. Both page-layout analysis and OCR work on a skew free image, thus, the skew angle needs to be corrected to zero degrees before further analysis can be carried out. The page layout analysis may fail in the presence of skew since



most page segmentation techniques are based on projection profiles which can be affected by skew.

Most pre-processing techniques are problem oriented and in all cases aim to improve the quality of the image. However, choosing an incorrect enhancement technique may degrade the image further. An extensive discussion of the work done on image pre-processing will be given in Chapter 2.

### 1.2.1.3 Layout Analysis

Documents do not only contain information in the form of text, pictures, or graphs but they also convey information expressed in the order, orientation and position of those components.

The organization of the components in a document is denoted as *physical layout*. The physical layout analysis (also known as structural or geometrical) consists of the description of the documents in terms of the geometrical aspects of the printed regions (e.g. position and boundaries) and the type of their contents (e.g. text, line drawing, photograph) [3].

Before any further analysis can take place, the document should be skew free. The *physical layout analysis* comprises three procedures: page segmentation, page classification and physical layout structure extraction.

Page segmentation is concerned with the identification of areas of interest in the document page. It will produce a higher level description of the printed areas in the document ~~by finding~~ the contour of these areas and outlining them. A failure to detect such regions, due to skew in the document, faint image regions or an incorrect page segmentation technique would avoid such regions being sent for further analysis and recognition. The presence of noise in an image could confuse the page segmentation algorithm mistaking the noisy data for the real data, creating false regions or modifying the contours of other regions.

Page classification is concerned with the determination of the type of content of each area previously segmented. Based on an analysis of attributes of the contents of each area of interest it can be classified, or labeled, into different types. These types or classes can vary according to the application. For an OCR application, the only concern is to differentiate between text and non-text regions.

After the document has been segmented and classified, the description obtained can be used by applications to perform further processing within the document image. Text areas can be sent to the OCR for recognition, to be then used as ASCII codes and substituted in the electronic document. Drawings could be converted to vectors, and pictures, or photographic components could be processed further. After recognition and processing it is possible to recreate a compact detailed representation of the document.

Failure to correctly classify the type of region would cause a non-text region to be sent for recognition. A similar case may happen if a region of noise is sent for recognition. If a text region is mistaken for a non-text region, it would not be sent to be recognised, and therefore its contents would be lost.

When analysing historical degraded documents, most of the standard recognition systems do not work so well due to the degradation on the documents or the difficult layouts.

Documents of the same archive usually share a creator, a topic, and therefore they can share a physical similarity or structure. For that reason, several projects for conversion of historical documents [6] can manually define and label their regions of interest by observation of a repeated structure. This document structure can be expressed by using XML templates which are then used by the recognition tools.

And secondly, the *Logical layout analysis* (also known as syntactic or functional) uses domain-dependant information consisting of layout rules of a particular page to perform labeling of the structural blocks, giving some indication of the function of the block.

An example of the result of functional labeling for the first page of a technical article would indicate the title, author block, abstract, keywords, paragraphs of the text body, etc.

#### 1.2.1.4 Optical Character Recognition

The regions identified as text by the layout analysis are sent to be recognised. The task of the OCR is to interpret a sequence of characters taken from an alphabet. Characters can be rich in shape and size depending on the type of font, or the handwriting style. Despite variations in the characters, they share a basic abstraction of their shapes that makes them identifiable. In the beginning of OCR, only one type of font could be recognised but nowadays, many

fonts are positively recognized. In the best case scenario, OCR can recognise a clean printed document with up to 99.77% [72] accuracy and for poor-quality<sup>1</sup> printed text from 89%, but still, while a human can recognise 100% handwritten or printed characters, there is no OCR that can match that.

Character segmentation is one of the main concerns for OCR. On printed documents, recognition is easier because usually characters are well segmented. However, when printed characters are touching due to a poor binarisation or noise, it can also lead to recognition errors.

The OCR algorithm can be divided into two components: *Feature Extraction* and *Classification*. The recognition is achieved by assigning a character to a class by using a classification algorithm based on the features extracted and their relationship.

- **Character Feature Extraction.**

Features are attributes that define the characters. To represent a character class we must know either a perfect character or a set of samples from that class. The feature selection must attempt to return the pattern attributes characteristic of each class. The character features chosen must clearly classify each character into its class reducing ambiguity. The classification stage identifies each input character image by considering the detected features. An extensive description of the feature-level analysis applied to this project will be explained in Chapter 5.

- **Classification.**

A classifier partitions the feature space into regions associated with each class. Template matching is the most natural approach as it consists of using pixels directly as features. An extensive description of classification methods applied to this project will be given in Chapter 5.

- **Contextual Processing.**

These techniques utilise knowledge at the word level to correct errors committed by OCR by using the information from other, well recognised characters as well as knowledge about the text in which it occurs.

Contextual information can be very valuable when the language in which the text is written is known. When in doubt of a character in a word, in

<sup>1</sup>Quality is a subjective term. Highly degraded documents perform even lower than 60% accuracy. Quality in documents is discussed in Section 2.2.5

most cases there will be only one possible word spelled in that way, or otherwise the possibilities will be narrowed down considerably.

Unfortunately, the use of contextual knowledge cannot be applied to all documents. This is the case of many historical documents where, for example the spelling of certain words can have changed over time, or as in archival documents, containing the names of people or places that can be spelled in several ways. There are also cases in which spelling mistakes that appear on the original document can be of interest to the historians.

For some of these cases, purpose-built dictionaries are created but this requires the knowledge in advance of the contents, language or terminology in the documents and it is mainly not the case.

### 1.2.2 Analysis of historical documents

When documents are preserved because they contain valuable information about persons, events or places they are known as *historical documents*. Most of these documents consist of laws, accounts of wars, or works of art. Generally, historical documents are very valuable and they are usually preserved in museums, libraries or archives.

Ancient and historical documents pose several challenges to the document image analysis and recognition systems. This is due to high variability in format, layout, creation method and conservation state. These documents may be handwritten, printed, typewritten, contain pre-printed tables or forms or a combination of these. In addition, they may contain stamps, seals, ornamentation, logos and posterior annotation by historians. They may be created on different kinds of paper and inks. They may as well be of varied colours and sizes.

The age and conservation state may also interfere with the recognition process. Many documents are too delicate to be scanned and they must be photographed instead. The quality of the materials and the preservation conditions will affect the quality of the document images.

The effects of degradation through age or repeated use can be reflected in the documents by faded characters, breaks, stains, wrinkles, dirt, coloured paper or see through characters.

Full text recognition is in most cases not yet available, except for dedicated OCR developed for specific documents. The vocabulary they contain is also large and may include unusual names and words and therefore the use of dictionaries to aid OCR cannot be applied to such documents.

### 1.2.3 Typewritten Documents

Of great interest to this project is the digital conversion of typewritten documents. These documents are characterized by being produced by a typewriter.

The typewriter was created at the end of the 19th century and it is described in a dictionary as "a mechanical, electromechanical, or electronic device with a set of *keys* that, when pressed, cause characters to be printed on a document, usually paper. The method by which the typewriter actually marks the paper is by the impact of a metal (or, later, metallised plastic) type element against an "inked" ribbon which caused ink to be deposited on the paper. When multiple copies are necessary, carbon paper is inserted between multiple pieces of paper (also known as "carbon copy" (CC)), so the type impression is transmitted to multiple layers of paper."

Typewritten characters are printed on paper individually. This differs from other documents produced by a printer where the amount of ink used for every character is the same throughout the document. Printed documents are characterised by all the text in the document being printed at the same time, for example newspaper print. As explained before, each typewritten character is produced individually by pressing a character key against a ribbon of ink. The force applied to each key will result in a print on the paper. If the key is stroked harder the character will appear darker, more intense. On the other hand, if the character key is pressed softly the character might be very faint. In addition, the amount of ink left in the ribbon will influence the intensity of the printed character. These possible differences on the intensity of each printed character in the document will pose a challenge to subsequent processing tools.

Another important feature of typewritten documents is that the space between them is regular. This is due to the equal size of the character keys in the typewriter. This is also known as *monospace*. Monospaced typefaces qualify by each character fitting in a "box" of the same size. This means that some characters that are small or narrow by nature, such as "i" or "l", will be stretched to fill

in this box and others will be made narrower to fit it, like "W", or "M". Therefore, characters like "i" and "m" will have a similar width in a monospaced typeface. On the other hand, proportional typefaces allow wider characters and thinner ones to keep their differences. If all characters fit in the same box size, hypothetically, a grid can be created based on the "box" size separating characters perfectly. This would be the case if the text is typed consecutively without removing the paper from the machine, otherwise, the regular spacing could be altered.

It is worth highlighting that monospace typefaces are not exactly equally wide. Even after stretching or widening, some characters still appear slightly wider than others. This does not affect the typewriter grid since the regular space separating characters is to be measured from the center of each character and not edge to edge.

The paper in a typewriter is placed and removed manually. As mentioned before this can frequently provoke the text to be skewed compared to the paper. Since documents might be removed and placed on the typewriter again for additional writing, a number of different skew angles might appear in the same document page altering the regular spacing between previously typed characters.

### 1.3 Key Contributions of the Research

The main objective of this research was to improve the quality of degraded historical typewritten documents to achieve better recognition results.

The text needed to be separated from the background and cleaned of noise and other degradations. These objectives included enhancing the characters by reducing the number of broken and touching characters, since broken and touching characters are responsible for most OCR errors.

To achieve the objectives set, a number of novel algorithms and techniques have been developed.

1. A novel **flexible character recovery technique** has been proposed, that is applied to each of the individually extracted characters. The recovery technique proposes a combination of local and global thresholding approaches. Both thresholds are calculated based on the individual values of each character, such as intensity, contrast or size. By applying the

thresholds just to each individual character the main weaknesses of both the global and the local thresholds in highly textured and low contrasted degraded characters are reduced. The final combination of both thresholds provides a ternary character image where the background and foreground are separated and a third layer of uncertainty (in grey-level) is kept to avoid binarising the image in an uninformed way. The method here proposed improves the recognition performance of two OCR systems in comparison to other thresholding methods for noisy and degraded historical documents.

It has been tried in a range of historical typewritten documents including carbon copies. It increases the recognition rates of two OCR systems on all tested images in comparison to other thresholding methods for noisy and degraded historical documents. The improvement is greater in the more difficult carbon copies.

2. A novel **character image refinement technique** has been created to enhance the ternary character image obtained from the flexible recovery. This technique studies the grey-level region in the image and determines, depending on the location of the grey pixel in the image, if it belongs to the character stroke, the background or belongs to a key uncertain area. The decisions for the refinement are based on the evaluation of the non-grey adjacent pixels. The result is a smoother and noise free character where only key areas, such as broken or touching regions are kept. These key regions can only be identified after the character is recognised.
3. A **character recognition system** based on weighted image matching has been implemented. The matching uses the proximity and the size of non-matching areas to weight the matching distance. The image matching results are combined with a classifier based on several character features. The implemented recognition system has obtained better results than professional OCR software systems.

Using the weighted image matching recognition system, the character images are restored based on the recognition results obtained from matching all the possible candidates created from all the combinations of the two states of the key uncertain areas identified in the previous refinement.

The results are restored clean characters free of noise and preserving, in most cases, connectivity and shape. The recognition rates obtained when using two different OCRs have improved by up to 35% compared to the results obtained by the original images, and several thresholding techniques used for comparison.

## 1.4 Structure of the Thesis

The thesis is structured as follows. Chapter 2 gives an introduction to Document Image Analysis in the literature emphasising on the work on historical and typewritten documents. This is followed by Chapter 3 with an introduction of OCR in the literature. Chapter 4 presents the *flexible text recovery (FTR)* method proposed here. The recovery is composed of *segmentation*, a new *flexible multi-thresholding* technique, and a novel *stroke refinement* algorithm. In Chapter 5, the *character recognition* proposed in this thesis is described. An intelligent *weighted image matching* algorithm was used together with a classifying probability tree. In this chapter a character restoration method is also proposed that exploits the information obtained from the recognition to restore the characters. In Chapter 6 a summary of the methods here used and their recognition performance are given. The recognition results of the methods proposed here will be presented by using two commercial OCR to evaluate their performance. As well, three well known thresholding methods will be used for comparison. Conclusions about the results will also be given in this chapter.

And finally, Chapter 7 will provide conclusions about the research problem, discussing its limitations and implications for further research.



## **Chapter 2**

# **Document Image Analysis**

### **2.1 Introduction**

In this chapter an extensive survey of typical Document Image Analysis (DIA) techniques will be given. As discussed in Chapter 1, the objective of DIA is to recognise the text and graphics in an image and extract its information. The digitally obtained image, represented by a set of pixels, will be pre-processed and ultimately recognised and stored as ASCII characters in words and paragraphs.

The main steps in DIA are described next, as are their applications and significant existing research in the area. Pre-processing methods will be described in detail in Section 2.2. Among the pre-processing techniques, skew correction 2.2.1, and thresholding 2.2.2 will be discussed. They will be followed by a description of thresholding methods for degraded documents in Section 2.2.3. Section 2.2.4 describes the image enhancement and restoration processes and systems.

### **2.2 Image Pre-Processing**

Image pre-processing aims to prepare the image for further analysis. In the case of DIA, the document image needs to be prepared to be recognised by the OCR system. After the document has been digitised, a number of pixel level processing techniques will take place. Any skew should be corrected and grey-scale, or colour images are usually binarised for the OCR. As well, during the

pre-processing step, the image may be enhanced if necessary. Image restoration methods aim to suppress degradation effects using knowledge about their nature.

As discussed in Chapter 1, degradation in images can have many causes. During digitisation (i.e. scanning, photocopying, photographing), the visual quality and the information content of the image can suffer. Poor illumination during the acquisition process can cause low contrast in the image while the camera or scanner might as well introduce signal noise. Other factors that might affect the quality are the type of paper, the ink, and the way the document is produced. The quality of a document can also decrease as it ages. Old documents tend to become yellow and, due to continuous use, ink tends to blur decreasing the contrast. Some historical documents can be already degraded when they are chosen to be stored.

The objective of the image pre-processing step is to improve the image's quality towards recognition.

In this section, an overview of the different pre-processing methods will be given, and the work done in the area will be discussed. In Section 2.2.1 different approaches to the detection and correction of skew in documents are presented. An extensive overview on thresholding techniques and binarisation is presented in Section 2.2.2, describing the different thresholding categories and methods. Their applications to degraded documents will be presented in Section 2.2.3. Thresholding algorithms created for, or applied to, degraded historical documents will also be of particular interest and are discussed there.

Image enhancement techniques will be presented in Section 2.2.4, such as noise removal, contrast enhancement, character enhancement, typewritten documents enhancement and image enhancement systems.

### 2.2.1 Skew Correction

For completeness, skew correction will be discussed here although it is not part of this thesis. The dominant orientation of the text lines in a document page determines the *skew angle* of that page, or region. Generally, a document has zero skew when its lines are oriented, either horizontally or vertically, parallel to the edges of the paper.

When manually scanning, photocopying, photographing, or even typewriting or handwriting, non-zero skew may be introduced.

Since OCR or other image analysis processes assume a zero skew document it is important to correct it beforehand. There are three main techniques for skew estimation [61]: using projection profiles; fitting baselines by the Hough transform; and nearest-neighbour clustering.

Other relevant publications in this area are [8], [84] and [4].

Some documents may be created in a different orientation angle for ornamentation, as in logos or advertisements. This different orientation must not be mistaken for skew.

A brief description of the three main deskewing techniques will be shown next. The easiest way to measure the skew using the projections of the image is to project the image in a range of angles and the correct skew angle will be the one described by the highest peaks in the histograms. It is often taken horizontally along rows or vertically along columns. These are called vertical projection profiles and horizontal projection profiles respectively [61]. Most commercial OCR systems use projection profiles, limiting the amount of skew they can handle to  $\pm 10$  degrees.

The *Hough transform* identifies alignments along lines by mapping each point in the original  $(x, y)$  plane to a point in the  $(r, \theta)$  Hough plane of possible lines through  $(x, y)$  with slope  $\theta$  and distance from origin  $r$ . It can be used on pixels, connected components or their centroids to reduce computation. It has also been applied to the base of the character regions. This method is usually higher in accuracy but slower than non-iterative methods [25] [61]. The above methods have some limitations in the maximum amount of skew they can handle. The *nearest-neighbour clustering method* does not have these limitations. The first step is to determine the different connected components in the image and then find the nearest neighbour for each component and the angle between their centres. These angles are accumulated in a histogram and the highest peak indicates the dominant skew angle. The accuracy of this method depends on the number of connected components in the image and the presence of noise. An extension to this method was made by O'Gorman [61] where, instead of just obtaining one neighbour, *k-neighbours* were obtained for each component (where *k* is typically 4 or 5.) In this case, connections between characters and between text lines can occur. The histogram is used to eliminate connections that do not correspond to the average. Finally, a least-squares fit is made to the centres of components that are still connected, in order to extract the text lines.

### 2.2.2 Thresholding

Among other pre-processing techniques, most DIA systems need to binarise the grey-level or colour input image obtained from scanning, separating in this way, text and graphics from the unwanted background, prior to further processing. This segmentation operation generates an output where there are only two possible modes, the background and the foreground. These are usually represented by the binary values 0 and 1 respectively. Binarisation can be achieved by *thresholding*. To binarise the image, a threshold  $T$  needs to be found where:

$$\bar{B}(i, j) = \begin{cases} B_f & \text{if } B(i, j) < T(i, j) \\ B_b & \text{otherwise} \end{cases}$$

Here  $B(i, j)$  is the intensity of image  $B$  at position  $(i, j)$ , while  $B_f$  and  $B_b$  are the intensity levels for the foreground and background.  $B_f$  is usually represented by 1 and  $B_b$  by 0 in a binary notation or 0 and 255 in a grey scale. Finally, the value  $T(i, j)$  is a fixed or spatially varying threshold that is applied to each position  $(i, j)$ .

The performance of the thresholding operation depends obviously on the choice of the threshold function  $T$  and this depends on the application. The document conditions, such as correlated noise, ambient illumination and busyness of grey levels within the object and its background. An inadequate contrast and the object size can also complicate the thresholding operation.

Classically, thresholding techniques were divided into *global* or *local* depending on the region it is applied to, and *adaptive* or *fixed* depending on how the threshold value is calculated.

Global thresholds are applied to the whole image and local thresholds are applied in small neighbourhoods. Fixed thresholds are decided without any analysis of the image and adaptive thresholds are based on data obtained from the image in a global or local area. A fixed threshold may be applied globally to an image or a variable threshold may be applied locally depending upon the local characteristics of the image.

To better present the different range of applications that have been created in the area of thresholding, the author will adopt the classification made by

Sezgin and Sankur in [79] to discuss thresholding methods. Thresholding algorithms are divided in six detailed categories depending on the kind of information exploited. A description of each category plus some of the best representatives of each are given next.

### 2.2.2.1 Histogram shape information

The first category is based on an analysis of the peaks, valleys, and curvature of the histogram.

*Convex Hull Thresholding* is based on the analysis of the concavities of the histogram by selecting the deepest concavity points as candidates for threshold. The method developed by Rosenfeld [75] also included object attributes, such as busyness, to aid the final decision. Other methods were based on an *analysis of the peaks and valleys* of the histogram. The method presented by Sezgan [78] convolves the histogram with a smoothing and differencing kernel reducing the histogram to a two-lobe function. The optimal threshold is found in the valley between the two lobes. These methods are globally adaptive. In historical documents, due to a noisy background or blurred characters, the valleys in the histogram of the document are generally not clearly separated. As well, global thresholding methods fail to correctly binarise degraded documents since there may be variations in the intensity throughout the document.

### 2.2.2.2 Clustering-Based Thresholding methods

In this class of algorithms, the grey-level data undergoes a clustering analysis, with the number of clusters being always set to two. Since the two clusters correspond to the two lobes of a histogram (which are assumed to be distinct), some methods search for the midpoint of the peaks [73]. These methods are classified as *iterative thresholding methods*. Among the *clustering thresholding methods* is the well-known method created by Otsu [63]. This method suggested minimizing the weighted sum of within-class variances of the foreground and background pixels to establish an optimum threshold. The Otsu method remains one of the most referenced thresholding methods.

$$T_{opt} = \operatorname{argmax} \left\{ \frac{P(T)[1 - P(T)][m_f(T) - m_b(T)]^2}{P(T)\sigma_f^2(T) + [1 - P(T)]\sigma_b^2(T)} \right\}$$

Other methods in this category are based on a *minimum error thresholding* [47], or a *fuzzy clustering thresholding* [39]. This last one assigns a fuzzy clustering membership to pixels depending on their differences from the two class means. On the other hand, the minimum error threshold assumes that the image can be characterised by a mixture of the distribution of foreground and background pixels. These methods are applied globally and may fail if applied to degraded documents with varying intensities and noisy background.

### 2.2.2.3 Thresholding methods based on Attribute Similarity

These algorithms select the threshold value based on some attribute quality or similarity measure between the original image and the binarised version of the image. These attributes can take the form of edge matching [94], shape compactness [65], connectivity [60], or texture [55]. The method by O’Gorman [60] for printed documents is based on connectivity rather than intensity, aiming to preserve this within regions. Since connectivity is a local measure, and since it is measured throughout the entire image, it is a global thresholding method based on a local feature. This multi-threshold method can be described as performing a global threshold at every intensity level and then choosing the thresholds where the number of connected regions is constant. As globally applied, these methods would also fail in degraded historical documents where there are intensity variations among characters and noisy background.

### 2.2.2.4 Spatial Thresholding methods

This class of algorithms uses not only the grey value distribution but also dependency of pixels in a neighbourhood, in the form of context probabilities, 2D-entropy, etc. Kirby and Rosenfeld [46] who considered local average grey levels for thresholding were one of the first to explore spatial thresholding features. Co-occurrence probabilities have been used as indicators of spatial dependence by Chanda and Majumder [17]. Higher-order entropy thresholding considers the joint entropy of two related random variables: the grey value  $g$  at a pixel and the average grey value  $\bar{g}$  of a neighbourhood centred at that pixel. It uses a 2D histogram with all pairs  $(T, \bar{T})$ , calculates the cumulative distribution  $P(T, \bar{T})$  and defines the foreground entropy [1]. Spatial thresholding methods are locally adaptive as they calculate the threshold for each neighbourhood based on the statistics of that region. Depending on the size

of such neighbourhood, these methods can be more or less sensitive to noisy backgrounds and blur characters, such as the ones found in degraded historical documents. The larger the size of the characters the less sensitive to noise it is. However, if too large, faint characters may be under-thresholded. Choosing the right neighbourhood size is very important for local methods.

### 2.2.2.5 Locally Adaptive Thresholding methods

In this class of algorithms, a threshold is calculated at each pixel, which depends on some local statistics such as range, variance, or surface fitting parameters of the pixel neighbourhood. The method from Niblack [59] adapts the threshold according to the local mean  $m(i, j)$  and standard deviation  $\sigma(i, j)$  and calculated with a window size of  $b \times b$  (see (2.1)), where the suggested value for  $b$  is 15 and  $k = -1.2$ .

$$T(i, j) = m(i, j) + k \cdot \sigma(i, j) \quad (2.1)$$

Sauvola and Pietaksinen method [76] is an improvement on the Niblack method, especially for stained and badly illuminated documents. It adapts the contribution of the standard deviation. For example, in the case of text on a dirty or stained paper, the threshold is lowered.

$$T(i, j) = m(i, j) + \left\{ 1 + k \cdot \left[ \frac{\sigma(i, j)}{R} - 1 \right] \right\} \text{ where } k = 0.5 \text{ and } R = 128 \quad (2.2)$$

Niblack, and Sauvola and Pietaksinen, used the *local variance* while others used *local contrast* methods (see (2.2)). In their method, White and Rohrer [98] compare the grey value of each pixel with the average of the grey values in their neighbourhood. The use of a  $15 \times 15$  window around each pixel was suggested to reflect the character size. If the pixel was significantly darker than the average, then it will be set as a character, otherwise it will be classified as background.

Another local method is the one of Bernsen [9]. In his method the threshold is set at the midrange value, which is the mean of the minimum and maximum grey values in a local window of suggested size  $w = 31$ .

### 2.2.2.6 Miscellaneous methods

Liu and Srihari [55] proposed a binarisation method based on texture features. Their algorithm consisted of a combination of three steps. Histogram analysis, in which a limited number of candidate thresholds are generated from a grey-scale histogram through iterative use of Otsu's algorithm [63]. Texture feature extraction, where texture features associated with each candidate threshold are extracted from the run-length histogram of the appropriately binarised image. Third, threshold selection, in which the optimal threshold is determined resulting in the most desirable document texture features. This method utilises two fundamental characteristics of most document images. First, the characters normally occupy a separable grey-level range in the grey-scale histogram. Second, text images, in general contain highly structured-stroke units. Their method achieved an 8% improvement compared to the classic Otsu method when tried on a set of 500 difficult handwritten mail address blocks.

As discussed above, different thresholding techniques have been developed for different applications. Postal mail recognition [55], bank cheque image analysis and recognition [42, 20], car plate recognition, etc. No one threshold can be applied successfully to all images and in general, new "problem oriented" thresholding techniques are created.

### 2.2.3 Thresholding of degraded documents

In their survey, Sezgin and Sankur [79] compared and ranked a number of thresholding techniques by their results on degraded documents. The dataset used consisted of 40 document images created with different fonts and type-faces. To simulate documents that are more realistic they used a set of degradation models that introduced blur and speckle noise similar to that produced by repeated photocopying and faxing. The thresholding methods that performed best in this evaluation were the clustering method by Kittler and Illingworth [47], followed by the local methods of Sauvola and Pietaksinen [76], of White and Rohrer [98] and finally of Bernsen [9].

Trier and Taxt [23] presented another evaluation of eleven locally adaptive thresholding methods on grey scale images of maps with low print quality. In their paper, they compared the results of the chosen methods before and after a post-processing step designed by them. The post-processing step they



implemented was aimed at the removal of edge pixels where the average gradient value at the edge of a printed object fell below a given threshold  $T_p$ . This threshold value was calculated by them on a trial and error basis.

They concluded that, overall, the best performing adaptive methods, prior to the post-processing, were the ones by Yanowitz and Bruckstein [103], White and Rohrer [98] and Bernsen [9] and after the post-processing step, the method by Niblack [59] performed significantly better, followed by Eikvil et al. [49] and Bernsen [9].

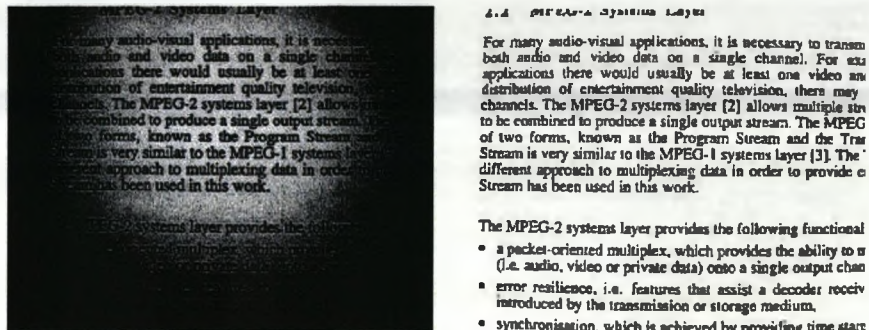
The conclusion obtained from the study and comparison of thresholding algorithms [79, 23, 52], indicated that no one single method performed better than the others in all images. Selection of an appropriate binarisation method, given different input images with different features or degradations, has been a difficult problem and it is still under extensive research nowadays.

As discussed in Section 2.2.5, degradation can appear in many different situations, due to various reasons. In some cases, documents may be degraded artificially for testing purposes [104] as were the ones used by Sezgin and Sankur. Trier and Taxt used map images with low print quality. It is important to highlight that, due to the different nature of the degradation in the evaluated documents, the results of these surveys cannot be used to predict the performance of such methods in documents with other kind of degradations. Some methods created for different degradations are given below.

Wu and Manmatha [99] proposed a method for the thresholding of images with highly textured background. Their algorithm followed two simple steps. The first smoothed the input image using a low-pass (Gaussian) filter enhancing the text relative to any background texture and reducing speckle noise. The second step identifies the first peak in the intensity histogram as the foreground and selects the threshold of the value at the valley between the first and second peak of the histogram. This is smoothed by a low-pass filter before thresholding is completed to aid the identification of the valley. The authors compared their results with the ones of Otsu [63], Kamel and Zhao [42] and Tsai [92] showing their superiority. This global method performs quite well on images where the background is textured but the quality of the printed text is good and not degraded. Accuracy is lower when the noise affects the characters.

The method by Yang and Yan [102] aims to threshold grey scale images with complex signal-dependent noise, variable background intensity caused

by nonuniform illumination, shadow or very low contrast. Their method first analyses the background by partitioning it until a quasi-bimodal histogram is found. When a bimodal histogram is found, a stroke width and noise analysis is performed by analysing the run-length histogram. This analysis is followed by a local analysis applied to local regions, determined by the stroke width and finished by a post-processing step to reduce noise. An example of the results of their method is shown in Figure 2.1.



(a) Document under bad illumination (b) Enhanced Document Image by condition and signal-dependent noise. Yang and Yan Method.

Figure 2.1: Image examples on the results of the method by Yang and Yan [102]

Another method for the binarisation of badly illuminated documents is proposed in [66]. Papamarkos proposed a clever technique for fuzzy binarisation of mixed-type documents based on a hybrid neuro-fuzzy system. The feature vector consists of the image grey-scale values and additional spatial features that emphasize text components. Their proposed method is simple and unsupervised and in most of the cases, led to desirable binarisation results. However, in cases of very poor quality documents the technique could only be used just as a pre-processing procedure.

Cameras present an alternative to scanning devices. However, since they are not specifically designed for document image capture they can introduce severe image variations and degradations that make it especially hard to obtain reliable OCR images. When binarising images digitised using a camera, the lower resolution can hinder the binarisation process. Seeger and Dance [77] proposed a new method for the binarisation of such low quality images. Their Background Surface Thresholding (BST) method computes a surface of background intensities at every point in the image and performs adaptive thresholding based on this result. The surface is estimated by identifying regions of

low resolution text and interpolating neighbouring background intensities into these regions. They compared their performance to the one of Niblack's [59] regarding OCR performance, noticing the improvement with fewer merged or split character strokes. Niblack's method tends to produce background noise when using small windows and tends to break the characters when using large ones.

### 2.2.3.1 Thresholding Degraded Historical Documents

Thresholding of historical degraded documents poses a challenge to the field. Most of the available enhancement methods do not perform so well in these documents due to the different nature of the degradation from one document to another. There are many projects attempting to enhance different historical documents, creating their own techniques and enhancing them individually to adapt to their particular degradations. Historical documents have been introduced in Chapter 1 and differ from average documents in their value, their contents, their authors and in some cases the need for their digitisation and their potential target user group (i.e. museum curators, historians and researchers). The kind of degradation and the origin of the documents are, in some cases analysed before applying any technique. Noise can be modelled, the document structure can be extracted, and enhancement is usually reduced to the identified regions. Archival documents tend to share a common layout. This allows in some cases, to focus on certain parts of the image to aid the enhancement and posterior processing. In other cases, different thresholding methods can be tried and combined to best suit the particular degraded document. No one thresholding method has proved to be good in all degraded images and, frequently, an inadequate thresholding can degrade the image even further.

Leedham [82, 51, 52, 100, 19] has been working in the historical handwritten documents area extensively. He proposes, in several of his publications, the advantages of a multi-stage thresholding, namely a threshold is performed in  $n$  stages, producing different binarisations in each stage through different methods. Multi-stage thresholding can be seen as a process of reducing the search-space of threshold candidates stage by stage where each step uses different information from the image until the final stage dictates the final threshold value.

In [51], Leedham et al. compare a set of global and local thresholding methods. They identify that the method, known as *Integral Ratio (IR)*, described in [82] is the best performing global method for badly contrasted images with non uniform background and foreground. This method is a two-stage thresholding where the first stage consists of dividing the pixels into three classes instead of two: the *foreground*, the *background* and a *fuzzy class* where it is hard to determine whether the pixel belongs to either of the first two.

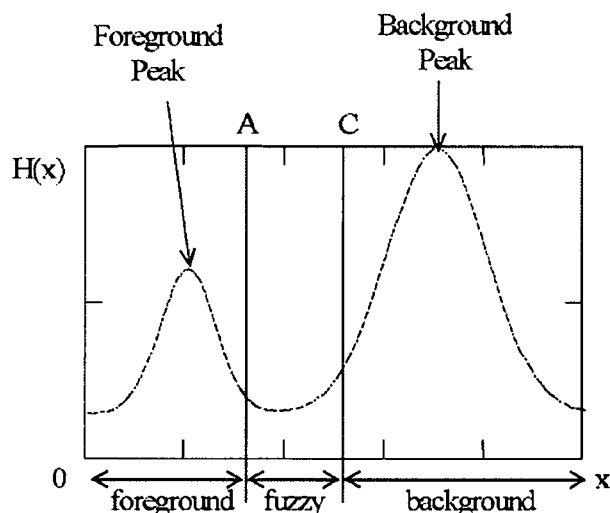


Figure 2.2: Three pixel classes in the approach by Solihin and Leedham [82]

The histogram in Figure 2.2 shows the three pixel classes where  $x$  denotes the pixel gray-scale intensity value and  $H(x)$  denotes the number of pixels in the image which have intensity  $x$ . Two important parameters that separate these classes are  $A$ , which separates the foreground and the fuzzy classes, and  $C$ , which separates the fuzzy and the background classes. If, however, a pixel has an intensity greater than  $A$  but less than  $C$ , it belongs to the fuzzy class and more information from the image is needed to decide whether it belongs to the foreground or the background. The first stage of the technique produces a range of threshold values delimited by the parameters  $A$  and  $C$  ( $T1 = [A, C]$ ). During the second stage, a final threshold value is chosen between  $A$  and  $C$ . They use the help of the *Quadratic Integral Ratio (QIR)* to estimate the best position of the points  $A$  and  $C$ .

Later Yan and Leedham [100] proposed another multistage thresholding

approach. Similar to the method proposed by Yang and Yan [102], their algorithm recursively decomposes a document image into sub regions until an appropriate existing global or local thresholding algorithm can be applied. The first stage analyses the curvature of the peaks and valleys of the smoothed grey-scale histogram and, if it is bimodal, then an existing global threshold can be applied to the whole image. If not, the image is recursively decomposed using quad tree decomposition into smaller regions, and each one is examined to see if existing techniques can be applied. This continues until the whole image has been thresholded. In order to decide the thresholding method several features are extracted from each sub region. The *edge strength*, the *variance*, and the *mean-gradient* which finds information about the stroke direction and sensitivity to noise. Based on these three features they characterise the sub-blocks into three classes: *background*, *heavy strokes* and *faint strokes*.

Quantitative analysis using word recall on more than 300 historical handwritten greylevel documents from the Library of Congress show the above technique has superior performance to six of the best performing methods. Their results showed good performance in all their testing images, but due to the nature of the features used such as the orientation of the strokes, it is only applicable to handwritten documents.

Noisy backgrounds are frequently a reason for thresholding failure in historical documents and background subtraction has been studied on several occasions [27] [31].

The method by Gatos et al. [31] has achieved good results in the thresholding of low quality historical documents. It consists of 5 steps. The first step is a de-noising procedure by using a low-pass Wiener filter. On the second step, they use Niblack's approach to estimate roughly the foreground regions. In the third step, the background surface is computed by interpolating neighbouring background intensities into the foreground areas that resulted from Niblack's method. The fourth step combines the obtained background surface with the original image to obtain a final thresholding. The last step is a post processing technique that reduces noise in the image. This method obtained the best results on low quality historical images among the current state-of-the-art adaptive thresholding techniques. The methodology proposed by Gatos et al. is shown in Figure 2.3

Background light intensity normalisation has also been proposed [81], as well as some other global thresholds [45] based on the relationship between

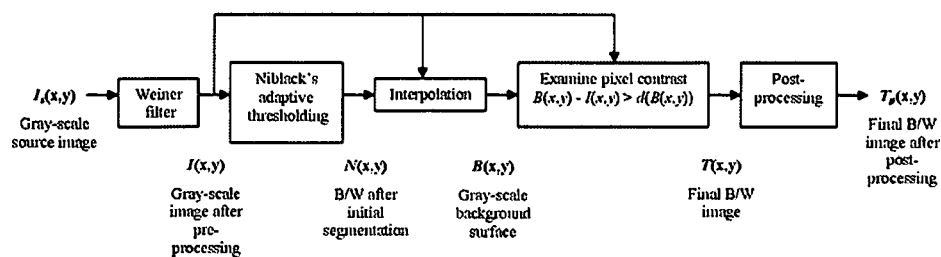


Figure 2.3: Block diagram of the method proposed by Gatos et al. [31]

the amount of foreground and background to choose a threshold.

A recent article on the separation of foreground and background in low quality historical document images was published by Garain et al. [27]. Their method is primarily applied to the thresholding of colour images but can also be applied to grey scale. The data used in their study consisted of a hundred images selected from handwritten notebooks of famous writers who used to write with quill or pencil generating very low contrast between the foreground and the background. Figure 2.4 shows the diagram of the steps followed by their algorithm.

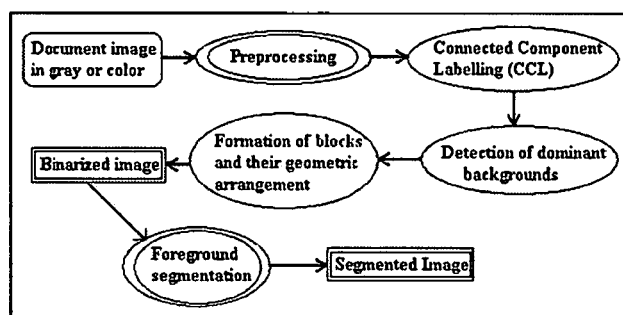


Figure 2.4: Schematic diagram of the approach by Garain et al. [27]

In conclusion, their method performed quite well on printed or handwritten documents under uneven illumination conditions or other degradations such as the effects of aging. The authors do not mention any results on type-written documents or whether the method can perform positively on documents where the foreground is damaged by degradation or its intensity is variable.

### 2.2.4 Image Enhancement and Restoration

Image enhancement techniques are processes for improving the quality of a degraded image. This is necessary since subsequent analysis stages, such as page layout analysis or OCR, require a pre-specific level of quality.

The factors that can contribute to the degradation in documents have been divided in two groups. The factors due to a wrong preservation condition, such as humidity, rust, tears, punch holes, breaks or stains. Alternatively, other degrading factors derived from the creation technique used, such as typewritten documents, carbon copies, photocopies and faxes. The way that those factors are reflected in the documents can be seen as variable background and foreground intensity, shadows, smear or smudge, skew, low local contrast, additive noise, broken and touching characters, blurred or faded character strokes.

Common office document problems are concerned with the digitisation of multi-generation photocopies where, with each generation, character strokes are progressively thinned and begin to break at weak points. When dealing with archived or old material, the original hardcopy may have been subject to aging or weathering, lightening the density of the text. Digitising and thresholding these images with an inappropriate method may result in thinned, or thickened text and broken or connected strokes. Connecting broken characters, and disconnecting the touching ones, is key to enabling accurate machine OCR on degraded documents, and restoring stroke width is of prime importance to achieving better document appearance.

Note that, although thresholding techniques may be especially designed to deal with the degradation found in degraded documents, they do not perform an enhancement but only binarise the image more consistently. The aim of document image enhancement techniques is to correct and reduce the effects on degradation towards recognition.

#### 2.2.4.1 Noise Removal

During the pre-processing step, commonly after binarisation, documents are usually filtered to reduce noise. Filter design and application [56] typically occurs by one of several scenarios. For common degradations, like the ones occurring in office scanner scenarios, one or more common filters can be pre-designed. The filters can be either user selected or automatically applied based

upon intelligence. Stored hardcopies can suffer from degradations such as "salt and pepper" noise in both foreground and background areas. Knowing the nature of these degradations allows for pre-design and application of global restoration filters. Usually the type of degradation remains constant for a collection of images, but the degree of degradation may vary from image to image, and even within each image's regions. In these cases, pre-designed adaptive filters are sometimes employed. Filter design and application for less common degradations and applications, may require specialised modelling for each given image. Possible degradations are classified as *antiextensive* (or subtractive), as thinness or holes, and in the opposite sense *extensive* (or additive), thickened strokes, connected characters and random patterned noise. This can be created by multigeneration copying, aging, weathering or improper thresholding techniques. They can be modelled respectively as:

$$S = (S_0 \ominus B) - N$$

$$S = (S_0 \oplus B) \cup N$$

where  $S$  is the degraded form of ideal image  $S_0$ ,  $B$  is a thinning or thickening element and  $N$  is random patterned noise. Other document degradation that is not *extensive* or *antiextensive* typically involves *ragged edges*.

Morphological filters have also been proposed for the restoration of binary images to clean noise, additive or subtractive, having the ability to preserve character's details. Mathematical morphology has proven useful in many image processing applications. The main reason for this success is that morphological methods can take into account the geometrical shape of the objects analysed. Soft morphological filters have proved to have better performance than standard morphological filters in noisy conditions. These filters are based on maximum and minimum operations. Soft morphological filters include the weighted median filter, which behaves robustly. Koskinen et al. [48] propose a method for the enhancement of binary images using soft morphological filters. Their method seems to perform well on bi-level images corrupted by salt and pepper noise and edge noise. Another method for document image enhancement based on morphological filters was proposed by Liang et al. [54]. In their paper, they discuss a method for binary morphological filters with a



size restriction, where each pixel in the input depends only on its neighbourhood, for example  $3 \times 3$ . In this way, it is possible to construct a look-up table between input and output. In their paper, they provide a methodology for knowledge based look-up table design. This methodology can be applied iteratively so that the final output image is the input image after being transformed through successive  $3 \times 3$  operations. In conclusion, they describe an increase in OCR performance of around 15% but do not show the resulting visual improvement.

Other degradations on bi-level images can be created by Fax machines. Faxed images have lower quality than average scanned images and can add to the image degradations such as salt and pepper noise. The thickening or partial omission of strokes or figures caused by the inappropriate thresholding by the sensor can add random noise. The method by Oguro et al. [62] addressed this problem by following three steps. First, they produce a greylevel image from the faxed one. Second, they perform a restoration based on the local pixel distribution, and finally they correct the image by comparing it to the input image.

Another method for the enhancement of faxed or degraded bi-level images is the one proposed by Hobby and Ho [38]. The essence of their method is to find and average, the bitmaps of the same symbol around the page and replace all occurrences of that symbol with it. After the initial skew correction, the characters are segmented and each character is then fed to the clustering procedure. From an initial set of clusters, bitmap averaging, by smooth shading, is then applied to obtain an outline character that represents the cluster. The characters are individually compared to the cluster average and assigned or rejected accordingly. Finally, the outline characters are rasterised and inserted in the corresponding places in the image.

Image restoration using resolution expansion is important in many areas of image processing. Techniques for improving the definition of low-resolution video images have been proposed by Thouin and Chang [88].

#### 2.2.4.2 Contrast Enhancement

The purpose of image contrast enhancement is to increase the visibility of images. Poorly contrasted images can occur from poor illumination settings,

faded ink, or dark textured paper, plus noise. Most of the contrast enhancement methods can be classified into two main categories: *intensity-based* and *feature-based* techniques.

In the *intensity-based* methods a transformation of the grey levels is applied to the whole image. In other words, pixels with the same grey level value at different places of the original image are still kept the same in the processed image. These methods try to rearrange the grey levels so that the image appears more distinct. They do not consider the contents of an image but its grey level distribution. To maintain the appearance of the enhanced image similar to the original one, the order of the grey levels in the original image must be kept invariant, that is the transformation function must be monotonically increasing. *Contrast stretching* is the most representative and the simplest method in this category [32]. Linear and non-linear functions such as *square*, *exponential*, and *logarithmic* functions can be used to enhance an interesting grey level range in the image. Another very popular method is *histogram equalisation*. This method assumes the information carried by an image is related to the probability of the occurrence of each grey level. To maximise the information, the transformation should redistribute the probabilities of grey levels in a uniform way [105]. The *histogram matching or specification* method works in the same way as the histogram equalisation, but in this case a specific histogram is used to specify the shape of the desired histogram.

*Feature based* contrast enhancement methods emphasize the contents in the image. Unsharp masks, or statistical methods, such as local means or variances, are representatives of feature based contrast enhancement in the spatial domain. In the frequency domain methods, the Fourier transform is first applied to the image, and then a range of frequency components is selected to be enhanced. The final image is obtained by the inverse *Fourier transform*. These methods can be applied as well locally for local contrast enhancement.

#### 2.2.4.3 Character Image Enhancement

A study by Stubberud et al. [85, 86] showed that OCR technology could recognise over 99% of the characters in a good quality page image but only around 85% could be recognised in low-quality images. To identify causes of OCR errors, common text image distortions which cause OCR errors were identified and classified concluding that approximately 77% of all the OCR errors are

caused by broken characters (52%), touching characters (20%) or both (5%). These distortions cause OCR to incorrectly segment the characters leading to misrecognition. To improve the OCR systems ability to correctly recognise touching characters, segmentation algorithms, such as [93], and segmentation free OCR techniques such as [18] have been proposed. Most algorithms address the touching character problem but only a few attempt to solve the fragmented character problem. Adaptive binarisation techniques can be used to improve OCR results, but they cannot improve bad originals, electronic binary documents or faxed documents.

In general, degradation does not affect all characters equally. Characters, which have neither a loop nor a gap, are less susceptible to distortion.

Shi and Govindaraju [80] proposed a method for character image enhancement by selective and adaptive stroke filling with a neighbourhood operator that emphasizes connectivity. This method was oriented to handwritten address block images. Using this method, the recognition of post codes went up by 7%. Another character enhancement algorithm was proposed by Hobby and Baird [37]. The idea behind their *Image Averaging* algorithm is, given an input set of degraded bi-level images of a single unknown character, to superimpose these images, add up the intensities at each point and threshold the result to obtain a new binary image. A post-processing step smoothes out the high-frequency "wobbles" in the outlines produced by the thresholding process.

The reconstruction of broken characters is one of the most difficult tasks in image enhancement. Reconstruction prior to recognition is usually not possible and records have not been found in the literature. However, reconstruction of characters after recognition has been proposed by Allier and Emptoz [2]. They attempted to reconstruct degraded character image by the use of active contours. This is done using two original kinds of energies, the first one is based on the use of a punctual attraction forces plot on degraded areas, and the second one is based on the use of an external "ideal" image. However promising, their method did not achieve good results.

#### 2.2.4.4 Typewritten Document Image Enhancement

Cannon et al. [14] are the main contributors to this area. Their method was aimed at degraded bi-level typewritten images and is among the few enhancing methods found in the literature aimed at degraded typewritten documents. Their method automatically selects a restoration technique for optimal restoration depending on the evaluation of five quality measures.

1. Small Speckle Factor (SSF). This measures the amount of black background speckle in the document.
2. White Speckle Factor (WSF). This measures the degree to which fattened character strokes have shrunken existing white connected components and created new ones.
3. Touching Character Factor (TCF). This measures the degree to which neighbouring characters touch.
4. Broken Character Factor (BCF). This measures the degree to which individual characters are broken.
5. Font Size Factor (FSF). There is a correlation between accuracy and size due to increasing and decreasing the font size.

Based on these five measurements they designed several restoration methods to repair the degradation.

- Do Nothing. Sometimes, the use of a non-appropriate enhancement technique can lead to worse results than non enhancement at all.
- Cut on typewriter grid. To separate touching characters using the fixed-width typewritten characters feature.
- Global fill holes and breaks. By filling out breaks and fractures using morphological kernels.
- Global Despeckle. To suppress black speckle while preserving shape.

The best restoration method is picked up by a trained classifier after taking into account the 5 quality measurements.

The method proposed by Cannon et al. is the only method in the literature that was intended for the enhancement of the quality of typewritten characters.

The application of a typewriter grid can be very effective in the segmentation of touching characters and it is still used confidently nowadays. However, segmentation errors in difficult images are still common in standard OCR software. These measures and enhancement methods were intended for black and white documents where the kind of degradations differ greatly from the highly degraded typewritten historical grey scale document images or carbon copies. Black and white speckle are common when the wrong binarisation technique is applied to a degraded document or when photocopying or faxing.

#### 2.2.4.5 Enhancement Systems

Standard enhancement or recognition systems are known to fail when dealing with unexpected degradation. For commercial reasons, most OCR systems are aimed at office and home documents where the expected quality is usually high. When confronted with degraded documents their standard methods fail to extract valuable information. As described in Section 2.2.4, many different methods have been developed to clean and restore the effects of degradation in a range of documents and applications. Sometimes, a simple method is needed to pre-process the document, but other times a combination of methods needs to be created. In real world applications, it would be helpful to *automatically* apply the best image processing methods to enhance the quality of text images before using OCR. Unfortunately, there is no single image processing transformation that always improves document image quality. In general, such transformations improve some images and degrade others.

The future of OCR research is to create a clever application that can decide automatically if enhancement is needed, what restoration technique is more convenient, and how or where to use it in the given image. Some general applications are being created with the help of intelligence to automatically choose the best restoration or thresholding technique for a given image. In 2004, Yang et al. [101] proposed an enhancement method based on segmentation and classification methods. The segmentation methods divide the images into statistically homogeneous segments and a neural network selects the best transformation for every individual segment based on the image statistical properties of that segment. The main reason to segment the image is that in different areas of the image, different methods were the best performers. In this way, each region can be enhanced by the best performing transformation,

chosen by a trained neural network. The transformations considered are de-speckle, fill hole, de-pebble, morphological open and close, and kFill transformations. The results showed significant improvement in reducing OCR error rate (around 35%).

A number of projects for historical documents that customise enhancement and recognition methods into systems to their particular problems will be discussed in the next section.

#### 2.2.4.6 Enhancement systems for degraded historical documents

The digitisation of historical documents, books and archives is one of the applications where standard OCR systems often fail to segment and correctly recognise text. An overview of some of the proposed systems for image enhancement will be given next.

Document analysis of ancient documents is mainly oriented to text extraction, since most ancient documents consist only of written text. In some cases, special characters, stamps or logos may need to be recognised or classified.

Damaged documents need special thresholding techniques that can deal with their particular degradation to create decent binary images. Standard office OCR systems are not prepared with algorithms that can accommodate degradation and can end up producing low quality, mis-segmented, broken or merged strokes in the output binary characters. The randomness of the noise location and its effects cannot be predicted by the OCR and leads to poor performances due to the variability in the characters morphology.

Many approaches have been proposed based on the use on Neural Networks (NN)(NN will be described further in Section 3.4), being more robust in the recognition of broken characters. Other approaches were based on Hidden Markov Models (HMM) [21], having the advantage of recognising sequences of characters without the need for previous segmentation using contextual knowledge. However, for very degraded documents, HMM does not perform satisfactorily.

Preliminary enhancement techniques have also been proposed to improve the quality of the images. As we discussed earlier in this chapter, there are filters and denoising techniques, although their use is derived from a trial-and-error supervised procedure making the task to chose the best method prohibitively expensive. Automatic restoration algorithms have been designed

such as [13] and methods based on Markov Random Fields (MRF) which are also suitable to enhance binary images. Such techniques are referred to as *joint blind restoration* and *labeling* and rely only on the given degraded image, not needing knowledge of the degradation operator. However, these techniques still leave a number of errors in the character morphology preventing satisfactory recognition.

For all these reasons, the development of efficient OCR systems for historical and very degraded documents must be considered an open issue. A common opinion is that the recognition of degraded documents requires intelligent systems and that it should *integrate* three fundamental modules:

- A **pre-processing** module for the joint blind restoration and segmentation of images;
- A **character recognition** module; and
- A **linguistic analysis** module.

The integration should be accomplished by, for example, the forwarding of the character images that fail to be recognised to the first module for a more refined restoration and segmentation, as well as a recognised character that fails the linguistic analysis.

The D-SCRIBE project [67], a system for digitization, processing and recognition of old Greek manuscripts, aims to support and facilitate current and future efforts in manuscript digitization and processing. This project created a software system to assist in converting an archive of manuscripts into a digital collection using automated methods. Among those methods they developed a binarisation technique for low quality documents and a recognition procedure based on the identification of characters with closed cavities [68, 31]. An image of the documents used in their project is shown in Figure 2.5

Two research projects were supported by the Italian National Research Council (CNR) with the objective of designing and developing computerised tools to retrieve and restore textual information contained in ancient documents, accessed as digital images. The first, *L'AperLA*, concerns old printed books, while the second, *Safeguard of Cultural Heritage*, carried out in the framework of a special project that aims at preserving cultural heritage, regards old manuscripts. Both projects are aimed at implementing an *integrated system* that improves the

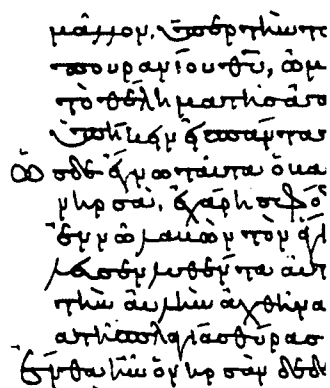


Figure 2.5: Sample image from a Greek manuscript from the D\_SCRIBE project [67]

quality of the images and, at the same time, carries out the optical character recognition (OCR) functions.

Tonazzini et al. [90, 95], collaborators within the second mentioned project proposed a complete method for the digital processing and analysis of highly degraded printed documents. The systems diagram is shown in Figure 2.6. The document is first cleaned, then the background noise is reduced by wave-

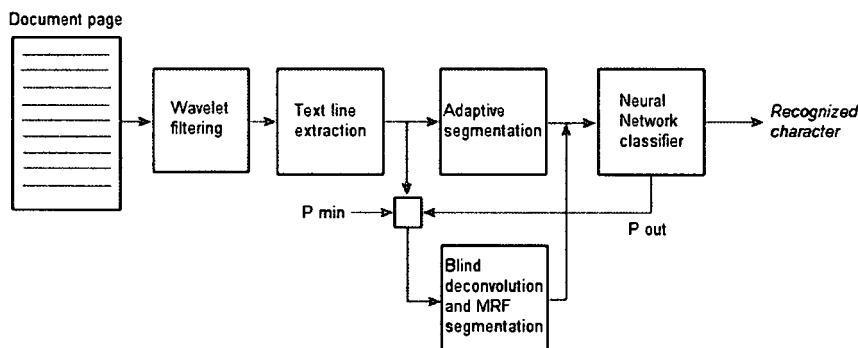


Figure 2.6: Integrated System implemented by Tonazzini et al [90, 95]

let-based decomposition and filtering. The next step is the text line detection and extraction, by a simple adaptive threshold separating the characters. The various components are analysed by a feed-forward multilayer neural network trained with a back-propagation algorithm. If the probability obtained from the recognition is lower than a determined threshold, the feed-back process is activated, sending the image back for refined segmentation. The refined segmentation of this small region is done by blind convolution and MRF-based



segmentation techniques, whose high complexity is reduced when applied to a small portion of the image. Their results show precise segmentation and effective recognition. In the sample image in Figure 2.7 can be seen the quality of the originals analysed by the system. The sample belongs to the First Book of the "Opera Omnia" by Girolamo Cardano, in the 16th century. This work

The image shows a close-up of a Latin phrase 'in pectore gratia exempli' from a document. The text is rendered in a dark, high-contrast, serif font, appearing as if it has been scanned from an old document. The background is a light, textured grey, suggesting the paper of the original document.

Figure 2.7: A sample image from the documents used by Tonazzini et al [90, 95]

was carried out by the Cultural Heritage Language Technologies; a collaborative project to create computational tools for the study of ancient Greek, early modern Latin, and old Norse in a network of affiliated digital libraries.

The MEMORIAL project [5, 6], on the other hand dealt with the complete life cycle in the digital conversion of a set of thousands of typewritten documents relating to prisoners in World-War II concentration camps. Not much work had been done previously on the conversion of typewritten documents into a logically indexed, searchable form. Some characteristics of the documents from the MEMORIAL project are described next:

- Most documents (as in most archives) are fragile and mass scanning cannot be used;
- The paper is frequently damaged by use and decay as well as heavily stained;
- The characters typed on the paper may be produced as carbon copies being frequently blurred and joined together;
- A logical structure may be missing in the text and position of documents; and
- An off-the-shelf OCR package must be used since the creation of a restricted dictionary for a purpose-built OCR is not possible.

The MEMORIAL project developed a Digital Document Life Cycle model. It is divided in two parts. The left arm of the model, as can be seen in Figure 2.8, represents analysis of information aided by the user. Digitisation, manual and automatic, qualification, where documents of similar structure, purpose and

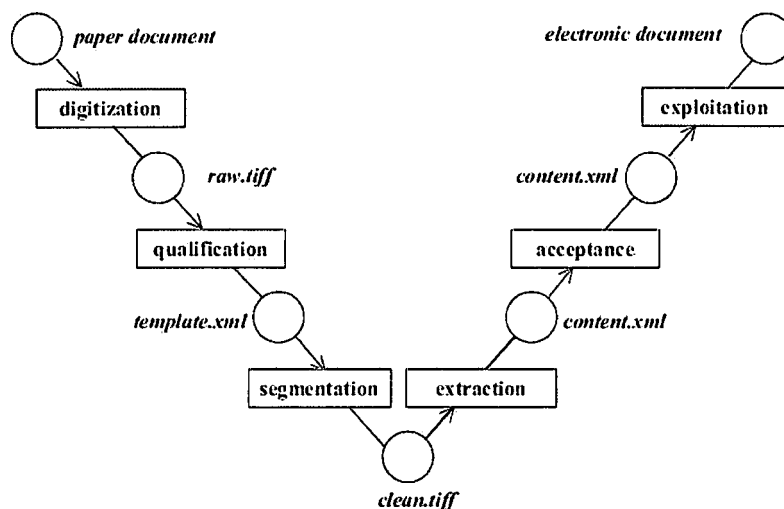


Figure 2.8: Digital historical document life-cycle [5, 6].

meaning are grouped into semantic classes. This was made possible by the use of two concepts: an XML document template and phase tuning.

Afterwards, the document template is consulted by the segmentation and extraction phases, where textual regions are segmented and improved by an Image Processing Tool, producing a clean bi-level image that is later processed by an OCR and a document content XML file is created. Following the right hand side of the diagram in Figure 2.8, the user intervenes in the acceptance phase to detect incorrectly recognised characters and supervise results.

The *content extraction* phase in the digital historical life cycle was of great interest to this project. The segmentation phase consulted the document template to get specific information on the region to be segmented. Segmentation was performed by exploiting the fixed-pitch of typewritten documents and an adaptive method was applied to the segmented areas aiming to minimise the existence of filled in holes and broken characters. They used the methods by Gatos et al. (GPP) [31], Weszka and Rosenfeld (WR) [97] and Niblack [59] for the thresholding. Their output results are evaluated by comparison of the application of the enhancement techniques at different segmentation levels, such as region level, text line, word and character level. The project dealt with mainly two kinds of documents, transport lists and personal cards (shown in Figure 2.2.4.6 respectively). The transport lists are carbon copies of prisoners



(a) Carbon Copy of a transport list (b) Personal card sample image. document image sample.

Figure 2.9: Image samples from the MEMORIAL project.

list names and they are generally highly degraded. Carbon copies show usually faint characters and blur edges. The other documents, the personal and index cards, were in some cases created by historians or museums and their quality is not as heavily damaged as the transport lists. In the case of the car-

	Transport List Sample		Catalogue Card Sample	
	Region	Character	Region	Character
WR	15.8.25 8.8.15 20.8.20	15.8.25 8.8.15 20.8.20	nalne wi p.Lipno	nalne wi p.Lipno
Niblack	15.8.25 8.8.15 20.8.20	15.8.25 8.8.15 20.8.20	nalne wi p.Lipno	nalne wi p.Lipno
GPP	15.8.25 8.8.15 20.8.20	15.8.25 8.8.15 20.8.20	nalne wi p.Lipno	nalne wi p.Lipno

Figure 2.10: Results of applying the different processing methods to different region levels. [5, 6]

bon copies, the results of the GPP method at region level were better and for the cards, with cleaner background, the results of the method by Niblack were visibly better at the character region level. Figure 2.10 shows the results mentioned.

Finally, a system for the conversion of archive documents from the Natural History Museum of London, has been researched by He and Downton [36, 35] to help convert more than 500,000 cards of insect species to a searchable digital archive.

## 2.2.5 A Note on Measuring Image Quality

As mentioned earlier, image enhancement techniques are processes for improving the quality of a degraded image. The required level of quality varies

with the kind of application that the document is being processed for. In text documents, the quality has to satisfy the recognition tools.

A document image of 'good quality' will present the right conditions to be perfectly recognized. For example, a good quality printed document will be one with perfectly clean background (usually white), without any noise and whose foreground (i.e. important information that we want to be read) is in a contrasting colour (usually black) whose characters are perfectly sharp, regular, well spaced and skew free.

Measuring the quality beforehand could help to predict the accuracy of the subsequent processing stages (OCR) or could help to decide on a specific restoration technique.

In order to measure the quality of a document, one needs to know exactly what is defined as an ideal document image and how much a degraded one differs from it. In the paper [43] the authors describe the ideal image as "a synthesized page image at 300 dpi without any artificial noise". There have been several studies to determine the quality and readability of document images since the beginning of OCR systems back in the seventies [89] [11]. Later in the 90's, [10] aimed to predict character accuracy achieved by any OCR system. Their work was limited to black and white printed pages where text areas had been already identified. The prediction system implemented simply classified the images as either *good* (i.e., when high accuracy was expected, over 90%), or *bad*, (i.e., when low accuracy was expected, below or equal to 90%). Their prediction method was based on three simple rules, based on three observations. The observations were that, thickened characters cause closed inner loops and touching characters. Broken characters are fragmented into smaller pieces that can be of any size. In addition, pages with inverse colours (i.e. white text on black background) tend to produce more OCR errors.

Cannon et al. [15] also development an algorithm to predict the OCR error rate that can be expected from a particular document page. Prediction of the OCR accuracy helped to choose whether a document page was fit for OCR or not.

In 1999 another paper about the measurement of a document's quality was published [34]. In this paper, they concentrate on measuring the quality of the black and white original documents ignoring the digitisation process and other OCR settings. Their aim was to answer the question: "-How low can accuracy be allowed to go before the process becomes unable to produce usable

documents?''.

They describe that the quality of the original can be a problem for the following reasons: it is old (physical degradation), it is a typewritten document, and so characters show variation in pressure and position; it is a carbon copy produced by a typewriter; or it is a low quality photocopy.

They make a number of observations. Fixing OCR errors is expensive since a human operator would be necessary to read and compare both versions. Recognition errors can be false positives, words incorrectly identified by OCR, and false negatives, words readable by the operator but not identified by OCR. The remaining errors can cause mistakes in using the information of the document. The possibility of using the image of the word instead when OCR fails helps users reading but it is unable to use these words in searches. Error rates in degraded documents can be so high that not even after manual editing the results are acceptable.

However, all the methods described above base their predictions on black and white document images. For historical documents, it is rarely the case that the document is bi-level. For this reason, none of the prediction of OCR error rate or the evaluation of the quality as previously described can be applied directly to historical document images. The quality of a historical document can be categorised objectively by its performance by an OCR system. It can also be categorised by its creation method, such as carbon copies, or handwritten documents or visually by common degradation, such as the amount of blur, see through paper or background noise.

## 2.3 Summary and Conclusions

In this chapter an extensive overview of the work performed in the DIA area has been given, focusing mainly on the work on documents affected by aging and other degradations. A number of enhancement systems aimed at historical documents proposed in the literature have been surveyed. Only the MEMORIAL project [5, 6] and Cannon et al. [14] approached degraded typewritten documents. The MEMORIAL project selected the thresholding methods by Gatos et al. and Niblack as the best thresholding methods for their images, were the method by Gatos performed better on the carbon copies at region level, and Niblack's method obtained better results on the file card documents

at character level (see Figure 2.10 in Section 2.2.4.5). Cannon et al. restricted their method to binary images where the kind of degradations differ greatly from the highly degraded typewritten historical grey scale document images or carbon copies.

## Chapter 3

# Optical Character Recognition (OCR)

### 3.1 Introduction

Optical character recognition deals with the problem of recognising character images. The recognition is carried out off-line and both hand-printed and machine printed characters may be recognised. Although character recognition is a subset of the pattern recognition area, it was this that gave the incentives for making pattern recognition and image analysis mature fields of science [24].

The history of research in OCR follows three main paths: template matching, structural analysis and neural networks. Template matching has been mostly used for the recognition of machine-printed characters, while structural analysis and neural networks have also been employed for hand-printed data. Template matching and structural analysis have been employed and studied for many years, but the use of neural networks is a relatively new technique that has been popularised in the last 30 years or so.

The first OCRs were based on *template matching*. The template matching process can be divided into two processes: superimposing an input shape on a template; and measuring the degree of coincidence between the input shape and the template. The start and end points in both the template and the input need to be found during the matching. In Figure 3.1 a sample of character template matching is shown. In this figure, two pairs of characters are superimposed to the template 'E', in orange. The resulting image shows each matching pixel in green and the non-matching pixels in their original colours.



The degree of coincidence between the images 'E' and 'E', is 100% since all the pixels match. On the other hand, the second pair, 'E' and 'B', shows a large number of non-matching pixels increasing the matching distance.

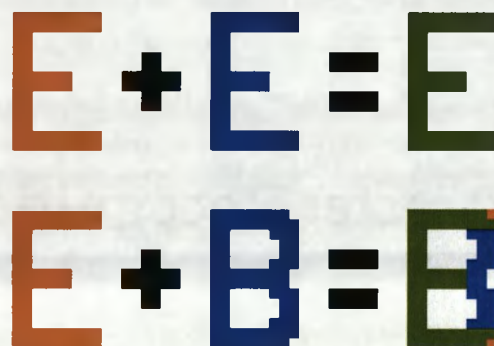


Figure 3.1: Template matching example

Naturally, the arrival of computers influenced the design of OCR with respect to hardware and algorithms. The initial step for any template matching method was to assume a binary character as input. Ideally, an input character has two levels of density: black and white, commonly represented by '1' and '0', respectively. However, real data are not always so distinct and binarisation is an important pre-process in OCR technology.

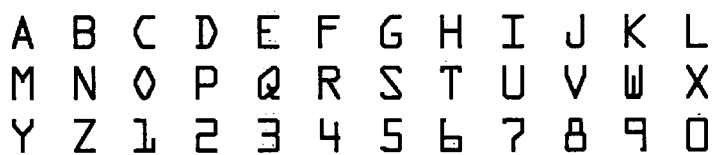
From *template matching*, the research orientation wave moved towards *Structure Analysis* methods. The principle underlying template matching is only appropriate for the recognition of a reduced set of printed characters. However, the variation in the shape of handwritten characters and the increasing amount of new fonts and varied sizes for digitally created prints is so large that it is difficult, or almost impossible, to create templates for all of them. A structure analysis method has been studied and applied to handwritten character recognition and is used nowadays in most OCR systems. In the case of structural analysis, there is no underlying mathematical principle and it is rather an open problem. Since a structure can be broken into parts, it can be described by the features of these parts and by the relationships between them. Then, the problem becomes how to choose features and relationships between them so that the description gives each character a clear identification. Feature extraction, therefore, has become the *key* in pattern recognition research.

The earliest commercial OCR page reader was developed in the late 50's



being able to read only one font in one point size. After that, *multifont* machines were developed, reading a set of fonts, but limited by the bitmap image library contained in the reader. Accuracy was quite good depending mainly on the quality of the original fonts. Two standardised fonts OCR-A and OCR-B were created in America and Europe respectively to provide a font where no (or lower) confusion between characters could occur. OCR-A is shown in Figure 3.2. These fonts were designed in a way that just by combining the horizontal and vertical profiles each character could be perfectly recognised.

Their aim was applications where exceptional accuracy was required, such as the Post Office, the Department of Defence or credit card companies. In the late 70's a system was introduced [57] that could be trained by the user to read any font. Training did still take less time than typing. Further systems replaced user training providing prior training in the lab and substituting the manual parsing with the automatic location of columns and pictures.



A	B	C	D	E	F	G	H	I	J	K	L
M	N	O	P	Q	R	S	T	U	V	W	X
Y	Z	1	2	3	4	5	6	7	8	9	0

Figure 3.2: OCR-A font

Recognition systems that are able to recognise all kinds of documents and fonts are called *Omni document*. The design goals of such systems include an *Omni text* engine, able to read the text in every page and to be able to correctly segment a page with complex layout and analyse and preserve its structure.

The *Omni text* engine has two main tasks, one is to be *Omni font* and the second is to be able to recognise degraded characters. The recognition of degraded characters is still the biggest challenge to every OCR engine. There are two different problems raised by degradation. Filled holes, blur, and incorrect pre-processing methods can lead to difficulties in the segmentation process and can break the characters strokes leading to segmentation and recognition errors [12].

Commercial OCRs perform the image pre-processing and image enhancement processes, as shown in Chapter 1, and then perform the recognition. The image is pre-processed, the skew is corrected and it is thresholded. The pre-processing is followed by the segmentation of the page and the layout analysis.

Once the character regions are identified, they are sent for recognition to the OCR. OCR segments the character images for then recognising them individually by extracting features and classifying them initially without taking into account contextuality. Contextual information is used in the following step to classify ambiguously recognised characters.

A good overview on OCR can be found in [87, 57, 58, 26, 12].

The next sections will describe the OCR steps in detail. The pre-processing step and the layout analysis will not be presented again as they were described in previous chapters. Section 3.2 presents *character segmentation* methods, followed by *feature extraction* in Section 3.3 and *classification* methods in Section 3.4. In Section 3.7 alternatives to OCR will be given. Finally, a description of common OCR errors, what causes them and how to measure them, will be presented in Sections 3.5 and 3.6.

## 3.2 Character Segmentation

The extracted image regions labelled as text are sent to be recognised. Before OCR, they must be perfectly segmented into characters. Among the techniques used for character segmentation there are *projection profile cut*, *run-length smearing*, *connected component analysis* and *segmentation by white streams* [61, 16]. The output image should be perfectly segmented into binary character images. Ideally non-broken, non-touching and noise free.

## 3.3 Feature Extraction

The objective of *feature extraction* is to extract the essential characteristics of the character symbols and it is generally accepted that this is one of the most difficult problems of pattern recognition. The features extracted should be sufficient to classify the character in its class and therefore robustness is necessary. The decision of which features shall be studied, which would describe the characters in a consistent way, which ones are computationally useful, how to combine them or how to strengthen their sensitivity to noise has been studied since the early development of OCR systems and remains under extensive research. One of the most studied techniques in feature extraction has been the study of the physical structure of the characters.

The ideal description of a good set of features [29] contains certain characteristics such as:

- **Discrimination.** Features should take different values for characters that belong to different classes.
- **Reliability.** Features should take similar values for characters belonging to the same class.
- **Independence.** Features should be uncorrelated with each other.
- **Small feature space.** The number of features should be small enough to make classification both simple and fast.

In addition, low computational cost and low complexity of the feature extraction techniques are also required.

Features can be of continuous values, such as real values like size, or can be binary, for example the appearance of holes. Binary features can have high discrimination ability and are less sensitive to noise. To evaluate the feature performance we can look at some other criteria such as:

**Robustness:** Sensitivity to noise, to local variations, to style variations, such as serifs or slants. Sensitivity to translation and rotation.

**Practical use:** Speed, complexity and independence.

Below are descriptions of some techniques for the extraction of features [91] together with a brief discussion of the results of these when applied to OCR systems outlining both their weaknesses and strengths. The techniques for extraction of such features are often divided into three main groups: *the distribution of points, transformations and series expansions* and *structural analysis*. Template matching can also be considered as a feature extraction technique. It will be discussed first.

### 3.3.1 Template Matching

The first recognition systems were based on template image matching. A set of "ideal" images was used to recognise the given segment by superimposing them and measuring the degree of coincidence between them. In image

matching each of the pixels evaluated can be interpreted as a feature. Template matching seems like a sensible and reliable method of character recognition, but it has some inherent weaknesses. First, as can be clearly inferred, template matching involves comparison with all possible candidates which is time-consuming. Second, shift variance, rotation variance and other issues of normalisation degrade the recognition rate. Two methods that were most studied, were *autocorrelation* and *moment based methods*; the second method is both shift and rotation invariant [91].

### 3.3.2 The Distribution of Points

This category covers techniques that extract features based on the statistical distribution of points. These techniques are not too sensitive to distortions or style variations. Some typical techniques on this area are *zoning*, where the rectangle surrounding the character image is divided into overlapping or non-overlapping regions and densities of black in each of these regions is computed and used as a feature. Another technique is the use of *moments*, where the moments of black points about a chosen centre, i.e. centre of gravity, can be used as features. *Crossings and distances* have also been used, as well as *n-tuples*. N-tuples have been researched in detail by Nagy et al. [40] and are described as a collection of  $n$  pixels with distinct locations and values that are superimposed on to characters and checked for transparency, meaning that each of the  $n$  values of the tuple needs to be of the same colour as the corresponding pixel in the character. *Character loci* requires that vectors be generated from each point in the background and the number of times these vectors intersect the character body being counted.

### 3.3.3 Transformation and Series Expansions

This techniques help to reduce the dimensionality of the feature vector. The extracted features can be made invariant to global deformations, such as translation and rotation. Many of these transformations are based on the contour curves of the character and can be affected by noise on the character's edges. *Fourier Transform*, *Walsh*, or *Hough transformations* may be also used.

### 3.3.4 Structural Analysis

It is very difficult to construct a template for each specific character because the variation of shape and style can be very large and, for that reason, structural analysis of characters started being researched.

*Stroke analysis methods* are amongst the easiest of the *structural analysis* approaches. A common feature of this method is that it looks at the character only partially, identifying character segments and simple relationships within them. This approach has to take into account the full structure of a character as, otherwise, valuable information such as orientation can be lost.

There are several ways to systematically see the complete structure of a character: thinning line analysis, bulk decomposition, stream following, contour following analysis and vectorisation.

**Thinning.** *Thinning* line analysis has been most intensively investigated, and many OCR systems have been made based on this. The thinning step consists of eroding a line from both its sides while keeping some constraints so that the line is not broken or shortened and connectivity is kept. This centre line can be also called the *skeleton* and the process is also known as *skeletonisation*. The aim of thinning is to reduce the image components to their essential information in order to facilitate further analysis. Since the 1960's, more than 30 different algorithms have been proposed for thinning. The basic iterative thinning operation is to examine each pixel in an image within the context of its neighbourhood region, of at least  $3 \times 3$  pixels and to remove the region boundaries, one pixel layer at a time, until the regions have been reduced to thin lines [61]. Another way to obtain the thinned version of an image is by *distance transformation*. This binary operation labels each pixel by the shortest distance from it to the boundary of the region within which it is contained and selects the pixels with farthest distance.

**Chain coding.** *Chain coding* and *vectorisation* are used to represent the skeletons or contours more efficiently than just by ON and OFF in the image. In *chain coding*, the ON pixels are represented as a sequence of connected neighbours and the direction of each previously coded pixel is stored. An advantage of chain coding is that it can facilitate further processing such as smoothing of continuous curves and analysis of the location of straight lines. An illustration of chain coding can be seen in Figure 3.3.

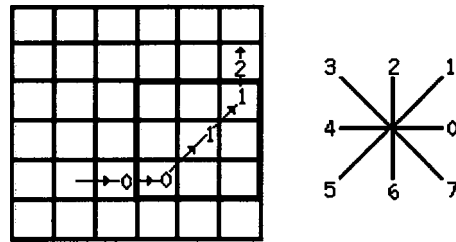


Figure 3.3: Illustration of chain coding and a coding rule.

**Vectorisation.** An alternative to *thinning* or *chain coding* is *Vectorisation*. Image lines are represented by the straight line segment that can be drawn within the original thick lines. It is intermediate in purpose between the *thinning*, *chain coding* and *poligonalisation* approaches. It is widely used for the conversion of engineering drawings. It might require user assistance if the quality of the original is not good.

**Bulk decomposition.** Among the structural analysis methods *Bulk decomposition* has also been studied. Consecutive black runs are analysed and connected if certain conditions are satisfied, in this way finding the different segments that form a character. The segments are used to describe the characters by extracting attributes from them, such as length, slope, or curvature, and then describing the relationship within them. One method by Spinrad [83], regarded as the slit method, consisted of eight directed slits on a frame which was superimposed on a character allowing long strokes, horizontal, vertical or diagonal to be found, through intersection. This method was based on feature matching.

**Stream following analysis.** The idea is basically to trace the curve of the input character by emulating the current of a stream. We can assume a coordinate system based on top to bottom, or right to left. Then we start from a given side and use the slit method explained previously to extract information from the input. For example, if we move from right to left, we track the black regions found in each slit. Therefore, if we find a black region that is not connected to any other black region, we establish a principle or a point to track. Then if the stream splits, we note the split and if parts of the stream reform, we note the union. At the end of the scanning process, a simple structure or shape of the segment has been determined using a vocabulary of principles, splits and unions. The description of this method is fairly simple and because of this

simplicity different shapes such as '+' and 'x' or 'U' and '∩' can be classified as the same. Thus the stream-following method is generally complemented with some analysis of the character's geometric attributes. Basically, all structural analysis methods use the overall structure of the character's bitmap to determine features. There are many different methods that have been formulated such as *contour following analysis* which basically follows the same methodology. These methods tolerate slight differences in style and some deformation in character structure. However, as it has been shown to be quite difficult to develop a guaranteed algorithm for recognition, heuristics are typically used. While these heuristics are somewhat tolerant, structural feature extraction has proved to be difficult in character recognition because of the significant variations that can occur due to degradation.

**Polygonalisation.** Polygonal approximation is one common approach to obtaining features from curves. Two approaches have been researched. The first approach is based on iterative and sequential methods. This method takes the distance between the original curve and the polygonal approximation as their error measurement. It starts by connecting the end points of the data with a straight line and then measures the perpendicular distance from the segment to each point in the curve measured. If this distance is larger than a chosen threshold, then the segment is replaced by two segments from the origin to that point and from that point to the end. This approach can vary with the error measure. Other methods used the area between the segment and the curve as a measure of error. Another variation is the use of the measure of the radius of the circle between the curve and the approximation segment. Other approaches to polygonalisation for closer approximation require more computation and more complexity. One method is to use a minimax measurement of error where the line segment approximations are chosen so that they minimize the maximum distance between the data points and the approximating line segment.

**Critical Point Detection.** Its objective is to locate the critical points in the shape outline of the image and represent the shape in more detail for each of the linear segments between the critical points.

**Line and Curve Fitting.** Approximation or fits of lines and curves can be used to identify features in objects. This representation by parameters instead of pixels can provide useful description, for example the number of straight lines, the angle between them or a particular type of curve. There are several

methods for straight and curve fitting. A popular way to achieve the lowest average error is to perform a *least-squares* fit of a line to the points of a curve.

Curve fitting features are described by the radius of the curve, the centre location of the curve and either its ends or the point where it makes a transition. Splines can be used to perform piecewise polynomial interpolations among data points. B-splines are piecewise polynomial curves that are specified by a guiding polygon, such as the one generated by polygonalisation.

The Hough transform is another famous approach to line and curve fitting. This approach involves a transformation from the image coordinate plane to the parameter space. One of its main applications is the determination of the text lines in a document and help with the skew correction.

### 3.3.5 Shape Description and Recognition

After the features have been extracted they will be analysed and used for classification. Depending on the application, different features will be chosen to describe the different classes. Handwritten and printed character recognition use different sets of features due to their different natures. A set of features has to be chosen that can maximise the difference between classes, in this case of characters. One of the best ways to describe shapes is by shape metrics. For example, *area measurement*, which is the number of ON pixels of the connected components in the image, can be used to discern between large and small characters, or even noise. The *length of its contour* can also be used, also the measurement of *compactness* which is the ratio of the square of the contour length over the area. Another way to describe a shape is by its *moments*. The first moment of a region describes its average location. Moments can be used to indicate roundness, eccentricity or elongation. A combination of moments that are invariant to geometric transformations are called *Moment invariants*. Moments are not useful for describing complicated shapes such as those with concavities or holes. Topological features are used instead to describe shapes with holes or branches. The number of holes can be determined by connected component analysis, although it can also be done by thinning and then finding the loops and branches. The number of branches, loops, endlines and their directions and lengths are used as descriptors of the shape.

Contour can be also used to describe a given shape. *Fourier descriptors* are used for this purpose, measuring the smoothness or sharpness in the frequency



of the contours. The drawback of this approach is that it only provides global information and cannot distinguish between shapes of the same frequency but different class, i.e. "d" and "p". Local features along the contour are also used and can be used as a sequence to describe a shape.

Finally, *projection profiles* are also used to describe shape. This is the accumulation of the number of ON pixels along all rows or columns into a one-dimensional histogram. The shape of the resulting histogram can be used as a shape descriptor. In the case of the previous example of the characters "d" and "p", the horizontal projection profile, which is the accumulation of pixels along all rows, could resolve the ambiguity by pointing out a larger number of ON pixels in the upper part of the character "p", opposite to the character "d".

Combinations of features are used in feature vectors where a set of features is combined and each character class should map separate points in the feature space.

### 3.3.6 Feature Extraction from Greyscale images

In order to avoid some of the information loss or difficulty related to binarisation of greyscale images, several methods have been proposed for the extraction of features from greyscale images [96, 50]. The best resulting approaches were based on the location of the character edges and then extraction of structural features from these and then the use of topographic features.

## 3.4 Classification

Classification is the process of identifying each character pattern and assigning it to the correct character class [21, 41, 57]. In the beginning of work on OCR, the character image was used as a matrix of features in template matching. A single feature is rarely enough to discriminate between classes and instead a set of features is necessary. The chosen set of features can be represented as a feature vector, or as the structural relation between them.

To classify a pattern, or set of features, into a category requires a learning process. A pattern recognition system learns through iterative adjustment of weights and/or other system parameters. Classifiers can be trained by repeatedly getting as input a set of patterns/features and the category where they belong until the number of prototypes is enough for a future similar pattern to

be classified correctly. These classifiers are called *supervised*. In *unsupervised* learning (also called clustering), patterns are associated by themselves into clusters based on some common features. Below, decision-theoretic pattern classification methods will be discussed, as will be structural and syntactical methods.

### 3.4.1 Decision-theoretic methods

These methods are used when the description of the character can be numerically represented in a feature vector. *Statistical Classifiers* use a classification scheme that is optimal, meaning, that it gives the lowest probability of making classification errors.

**Bayesian decision theory** is a fundamental statistical approach to pattern classification.

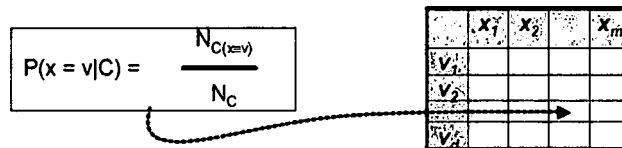


Figure 3.4: Contingency table

The probabilities calculated during this training are stored in a feature distribution matrix known as the *contingency table* (i.e. a matrix of frequency counts of each observation). This is illustrated in Figure 3.4. The goal of the Bayes classifier is to classify any new sample character minimizing the probability of misclassification. Two common learning strategies are *probability density estimation* and *direct loss minimization*.

**Tree classifiers or decision trees.** A classification tree breaks the original classification task into multiple classification steps. In other words, instead of assigning the correct classification in a single step, the objects are initially classified into subfamilies. Then, in a second step of classification, each of the subfamilies is classified into another number of subfamilies. This process is repeated for each subfamily until a full classification is performed. Given an image and a classification tree whose set of objects in the root node is the set of all objects present in the image, then, in order to classify those objects, one must traverse the entire tree (except for the leaf nodes) applying the operators.

For the optimal construction of a tree a criterion is needed in order to select, at every node, the feature with the maximum discrimination ability, plus a criterion in order to decide whether an end node has been reached, or if there is a need to extend it further. An excellent review on decision trees can be found in [70].

There have been different kinds of trees used in the literature for both character classification and pre-classification [7]. Binary trees have been used by Gatos et al. [29], and hybrid combination between trees and other classification techniques such as neural networks [64], or Bayesian probability [41] are also proposed in the literature.

The use of trees in classification relies on robust features since one wrong feature could result on misclassification. Fuzzy trees have been proposed as have been probabilistic fuzzy trees.

**Neural Networks (NN).** Recently the use of neural networks to recognise characters has resurfaced. A back-propagation network, is composed of several layers of interconnected elements. A feature vector enters the network at the input layer. Each element of the layer computes a weighted sum of its input and transforms it into an output by a nonlinear function. During training the weights at each connection are adjusted until a desired output is obtained. A problem with NN in OCR may be their limited predictability, while an advantage is their adaptive nature.

### 3.4.2 Syntactic pattern recognition

For syntactic pattern recognition, qualitative description of objects is characteristic. Measures of similarity based on relationships between structural components may be formulated by using grammatical concepts. The idea is that each class has its own grammar defining the composition of the character. A grammar may be represented as a string or as trees and the structural components extracted from an unknown character is matched against the grammars of each class. The elementary properties of the syntactically described objects are called *primitives*. Relational structures are used to describe relations between the object *primitives*. The set of all primitives is called the alphabet. The set of all words in the *alphabet* that can describe objects from one class is named the *description language*. A grammar represents a set of rules.

### 3.4.3 Combination of classifiers

It is well-known that classifier combination [33] provides increased accuracy over individual classifiers. One can boost the performance of a single classification paradigm by training multiple classifiers on different data sets. Alternatively one can combine classifiers that use different features, training samples, and decision-making methodologies. The most general way to combine outcomes from disparate classifiers is to aggregate their ranked outputs. The correct setting for analysis in this case rests in the statistical theory of groups. In the literature, classification in several stages has been proposed. Gatos et al. [30] proposed a multiclassifier that used curvature features of the characters combining the results at each sequential stage. A two-stage classifier for broken and blur digits was proposed by Rodriguez et al. [74] which in the first step classifies characters by size based on a global feature and, secondly, uses a specialised classifier based on structural features.

## 3.5 OCR errors description

OCR technology has advanced to the point where it has become reliable and useful for processing a large variety of machine printed documents. Accuracy is now over 99.8% in clean printed documents. However, when these documents are degraded the recognition rate is lowered in some cases reaching a state where the recognition can become useless.

In this section a description of the imaging defects, noise or any other degradation that can prove difficult for the recognition process will be discussed. The publications by Nartker, Nagy and Rice [87, 26] examine in detail the reasons for OCR failure, classify them and try to find a possible solution. In their book [87], the authors divide the errors into four major classes and several subclasses. The main classes are: *imaging defects, similar symbols, punctuation and typography*.

### 3.5.1 Imaging Defects

The class *imaging defects* is characterised by defects that are introduced to the document somewhere between the printing process and the acquisition of the image. Defects in the document may arise from the moment of printing, such

as rough, porous or glossy papers, a poorly maintained typewriter, with worn ribbons and dirty character keys, fast newspaper presses or later in its life, like multiple generation photocopy or by aging.

These defects are common among historical documents such as coloured paper, faded ink and wrinkles and breaks. Scanning can also introduce imperfections, such as incorrect illumination settings or low acquisition resolutions.

The first imaging defects subclass they describe is the *heavy print*. Thickened characters may occur due to multi-photocopying, too high threshold selection, or just by inappropriate enhancement techniques. Heavy print can distort the characters shapes, closing inner holes and closing opened gaps, making them unidentifiable. Some thickened characters may touch each other misleading the segmentation algorithm and therefore the resulting recognised character.

The x-height letters, a c e m n o r s u v w x z have no ascenders or descenders and are especially prone to confusion due to heavy print. Particularly the filled in s. The characters a and e are also easily filled up due to the small area covered and the proximity between strokes. A touching rn is easily mistaken for an m. Similarly, rm resembles nn and rrn. When the dot in i is touching the body of the character it can be confused with the character l.

*Light print* is discussed next. In OCR systems, fragmented characters cause more errors than heavily printed ones, due to the reduction in information. Light print may be caused by lightly pressed typewriter keys, worn ribbons, nearly empty printer cartridges, on an inadequate low threshold. When the characters are broken, the segmentation algorithm can be confused and may join fragments that do not belong together or separate others that do. Italics are easily broken due to their fine strokes. Examples of misinterpreted broken characters are, for example, a broken H interpreted as a II, an e into c, a broken D into a I), a d into cl or a k into lc. In general, horizontal strokes tend to be thinner than vertical ones and tend to break more often. The same thing happens with diagonal strokes where the ones in NW-SE direction are generally heavier than the NE-SW strokes.

Heavy and light print may occur within the same word or even character. The combination of broken and touching characters in a word or text can difficult the recognition further than separately.

Documents are rarely immune to the appearance of large blobs of noise or *stray marks*. Low quality print or coarse paper can cause them, as can external

degradation or problems during image acquisition. Sometimes, printed documents have hand written annotations, signatures or underlining that can be confused with noise. Stray marks can disorient the OCR system and are often accompanied by heavy or light print. Even a very small speck can confuse the OCR.

The last subclass is the one caused by *curved baselines*. This results from the photocopying or scanning of a bounded book creating a slight angle on the text that is closer to the binding or gutter and so distorting it. This distortion has to do with the acquisition mechanism and not with any external degradation and will not be discussed further.

In historical documents, imaging defects may appear mainly during the creation and preservation stages. The image acquisition of historical documents is a delicate process and extreme care and precision is taken during digitisation. Historical typewritten documents possess a combination of most of the defects here mentioned. Heavy and light print are very common. Strongly pressed characters can be found next to lightly pressed or blurred characters. Highly noise backgrounds that may include paper wrinkle marks, stains, breaks, etc. An incorrect thresholding method may worsen the quality of such documents. In general, typical enhancement methods do not manage to improve the image quality of highly degraded documents.

### 3.5.2 Similar Symbols

The second class of OCR errors described in [87] are *similar symbols*. Even on clean images, certain symbols are difficult to distinguish. Contextual information is used nowadays to resolve ambiguity. Very similar symbols are character O and numeral 0, or 1, 1 and I. For example, in some old typewriters, character l and 1 where printed using the same key, or in some fonts, such as *sans serif Helvetica*, capital I and 1 are indistinguishable. A common rule is to believe that digits appear with other digits and letters appear with other letters, except in some post codes, or address lines. Capitals C and G, R and B, 5 and S can also be confused due to similarity, or lower cases and upper cases of the same characters.

In typewritten historical documents confusion among similar characters

occurs more frequently than in better preserved documents. Degraded characters that are broken or touching other strokes may modify their shape to resemble other characters. In some historical documents, such as archives, contextual information cannot be used since these documents may contain words that may be spelled in many ways, such as names or be written in several languages. A challenge to OCR is to distinguish between a broken or touching character from a clean character without using contextuality.

### 3.5.3 Punctuation symbols

*Punctuation symbols* can also create confusion for OCR. The physical similarity of commas and periods, plus their reduced size, makes them difficult to recognise. They are followed by hyphens and dashes, quotation marks, exclamation and question marks and other special symbols.

In highly noisy backgrounds, like those found in historical documents, OCR tends to fail to separate the foreground and the background confusing some background noise with punctuation symbols. In many cases, such noise regions resemble punctuation symbols, like dots, commas or dashes and are easily confused by the OCR.

### 3.5.4 Typography

The last class is for *typography*. The way the document is laid out, the typefaces used, the variations in size and colour are not OCR friendly. OCR-A and OCR-B are typesettings specially designed for OCR, have uniform stroke widths and all are clearly distinguishable. However, the number of different typesettings in use increases with time resulting on a continuous challenge for OCR. Typefaces can be grouped by the appearance of *serifs* or not, *sans-serifs*, the horizontal space each character takes, constant space known as *fixed pitch*, or monospace, or variable, known as *proportional pitch*. Some typefaces may belong to one or more of these categories. A typeface can have as well several styles, such as *normal*, *bold*, *italic* or *bold italic*. A typeface-style-size combination is known as a *font*, i.e. **12-point Arial Italic**. Other typographic features that may complicate the recognition are underlining, the segmentation problems of the slanted italics, shaded backgrounds, and either very large or very small prints.

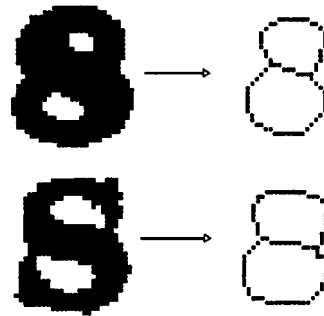


Figure 3.5: Illustration of the limitation of skeletonisation [12]

Some OCR systems may use stroke thickness variations for the recognition of similar characters. As seen in Figure 3.5, after skeletonisation, these two images can no longer be distinguished. The strokes in the character S have been thickened and now they are closing the gap creating inner holes, similar to numeral 8. This figure shows one of the limitations of skeletonisation. The second limitation of skeletonisation is its sensitivity to noisy characters where no obvious strokes are distinguishable due to thickening or breaking.

Topological and geometrical features extracted from the thinned images have been used in OCR systems as a way to filter out font-specific attributes, such as the exact position or angle of a crossbar. However, similar limitations to skeletonisation were found in heavy characters, showing touching edges or deformations. Figure 3.6 illustrates the limitations of topological and geometrical features due to touching edges. Degraded characters require more features than good quality characters and a robust classifier.

Typewritten documents do not present a varied typography. Each typewriter has just one fixed pitch font with a limited number of symbols. Some OCR systems include an optional typewriter settings to aid the recognition.

### 3.6 Measuring Recognition Accuracy

In order to evaluate the performance of the recognised text, a measure of the accuracy can be made. However, this is a complicated task. First, to measure the accuracy of a given page we need to have the correct, ground-truth version for comparison. The cost of producing the correct version of the document is very high, since it must be manually retyped and each character must be



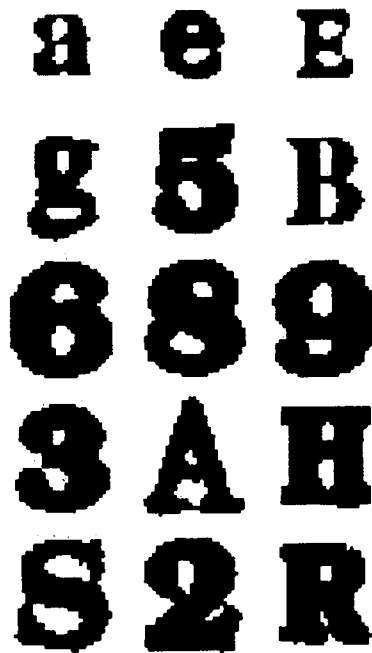


Figure 3.6: Illustration of the limitations found by topological and geometric features [12]

checked and rechecked for correctness. Second, it is necessary to conduct such tests with large amounts of pages. In general, tests of hundreds of pages are preferred in order to insure statistically significant results. And, third, it is not easy to determine which accuracy measure is more appropriate for a given application. The conventional measure measures the correctness of every ASCII character on each page. Correctness is given by the following formula:

$$\text{Character Accuracy} = \frac{\text{Total Characters} - \text{Character Errors}}{\text{Total Characters}}$$

Character errors are considered the sum of all the insertions, deletions, and substitutions needed to convert and output character string into the exact ground-truth string.

Word accuracy can also be measured and is described as the following:

$$\text{Word Accuracy} = \frac{\text{Total Words} - \text{Number of Incorrect Words}}{\text{Total Words}}$$

Word accuracy is considered to be a better accuracy measure for information retrieval systems.

To compare the groundtruth and the recognised version of the image, a string comparison method could be used, such as [53].

### 3.7 Alternatives to OCR

Although not the main focus of this thesis, several alternatives to OCR will be discussed for completeness. Indexing documents is a useful way to retrieve information faster. Lately more and more libraries and institutions are making their collections available online or in a digital format. Many of these documents are historical, handwritten or just too degraded to be successfully recognised by OCR. Rath and Manmatha proposed a set of features [71] for word spotting in historical handwritten documents, to aid text indexing. All the words in a collection are matched as images and grouped into clusters which contain instances of the same word. The selection of the best features for matching words is a crucial step in word spotting. They use a variety of projection profiles of each word; a complete profile, an upper and lower profile of the word and upper, centre and lower profile. The upper and lower profile scored the best results. Other features were the *ink to background transactions*, and vice versa, the aspect ratio of the image and the greyscale variance.

Then, for each word image to be indexed, the first comparison is to rule out the set of images that have a different aspect ratio and after this reduction, comparing the remaining possible images based on the extracted features.

A more recent approach was proposed by Gatos et al. [28] for segmentation-free keyword search in historical typewritten Greek documents. In this paper, they describe the pre-process they used to prepare the image plus the features they used to match the words. One of these features is based on the analysis of the density at some regions in the word-image, proposed in [12], and as a second feature they use area of the outlines of the upper and lower profile of the word as seen in [71]. Their method is aided by some user feedback, and the character degradation seems to be low, with no broken strokes, and little additive noise.

In Figure 3.7, a sample image containing non conventional characters is shown. This document contains non-text characters that are not suitable for



Figure 3.7: Example of a text-like block non-suitable for conventional OCR [69]

OCR. As a solution, Pletschacher [69] proposed a way to store and process a digital version of the image on various digital platforms by extracting a document immanent alphabet, preserving the graphical representations by means of vectorisation to encode the original document. Like this it is possible to gather benefits of encoded text without the effort and the possible mistakes that arise from recognition methods. The use of the Extensible Mark-up Language (XML) for structural descriptions and Scalable Vector Graphics (SVG) for graphical representations enables a seamless integration into style sheet based output workflows for producing system specific layouts.

### 3.8 Summary and Conclusions

In this chapter an overview of OCR with common techniques for feature extraction and classification has been given. As well, in this chapter, the most common OCR errors and their main causes were discussed. A noisy background, broken and touching characters are the main contributors to OCR errors and are commonly found in historical typewritten documents. Heavily pressed characters in typewritten documents, and carbon copies pose a clear challenge to up-to-date recognition tools where the currently used methods cannot deal with such levels of noise and blur and fail to produce useful recognised outputs. In conclusion, there is a need for a specialised method to successfully extract and recognise text in historical typewritten documents and their carbon copies.

# Chapter 4

## Flexible Text Recovery

### 4.1 Introduction

There are several reasons why most thresholding methods cannot be applied successfully to degraded typewritten documents. As mentioned in previous chapters, typewritten characters are printed individually and therefore their stroke's intensities can vary according to the strength applied to each character key and the amount of ink left in the ribbon, plus posterior degradation. Due to these variations among the foreground intensities, global methods almost always fail.

Besides the intensity variations among characters, global methods also fail to adapt to the variable blur surrounding each character. This failure results in closed character inner holes, or very thick strokes when enhanced globally. Historical documents might also be affected by other degradations, like wrinkles, or stains, altering the intensity of the and colour of character strokes. Globally applied methods would again fail to adapt to these degradations, producing breaks or dark patches.

In addition, due to the poor preservation conditions of old documents, the background can be highly textured due to the accumulation of smudged ink and dirt. The character strokes can be faded and blurred, disappearing into the background. These degradations affect the performance of local adaptive techniques, which use neighbouring pixel values for their measurements, outputting large amounts of noise, deforming the character's shape and making strokes touch other edges, creating false inner loops.

The aim of the method here presented is to be able to effectively extract and

enhance the text from degraded typewritten historical documents. The desired output is well segmented characters that maintain structure features and can be correctly recognised.

## 4.2 Overview of the Method

A novel approach has been developed to overcome the errors produced by the already known, global and local techniques. The novelty consists in enhancing the characters individually. Prior segmentation is necessary. The individual improvement of each character solves the lack of adaptation problem of the global methods.

The *segmentation* of the characters will be presented in Section 4.3. Two character segmentation techniques will be discussed. A standard segmentation technique, based on the image projection profiles, is contrasted with a fixed-width segmentation using the fixed-pitch of characters in typewritten documents and its broad position identified.

After each character is segmented, a closer *localisation* within the segmentation area follows to exclude as much background as possible and to reduce sensitivity for the adaptive methods.

Once each character is precisely localised a box is tightly placed around it. This will allow studying the intensity values of each character and permitting its individual enhancement. A combination of two thresholding methods has been studied towards the extraction of the characters body. First, the *Cautious Ternarisation* method applies a threshold that divides the localised character image into three categories. The three categories are: the pixels that can be confidently considered as part of the strokes; the ones that can be considered as background; and those that cannot be confidently classified as either. This multi-threshold was inspired in the Integral Ratio method by Solihin and Leedham [82]. The second text extraction method is the *Aggressive Binarisation*. This adaptive method performs very well on the faded areas of the characters but is very sensitive to noise, as in the textured background and character strokes. Both methods, the *Cautious Ternarisation* and the *Aggressive Thresholding* are finally combined, drawing from the advantages of each method to produce a final image. Finally, the background is cleaned and the image only contains the segmented clean characters. The whole approach is summarised in Figure 4.1.

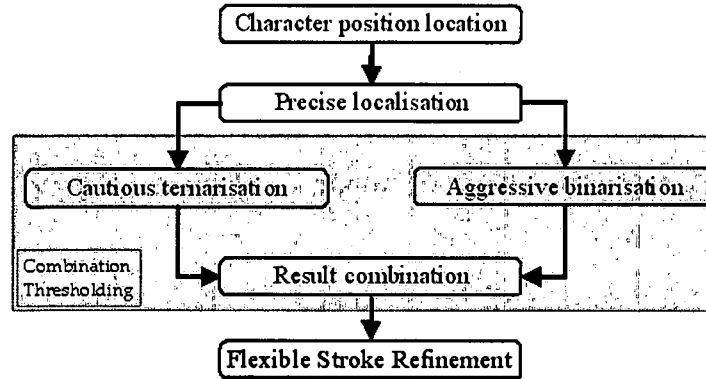


Figure 4.1: Outline of the proposed method

The text recovery method proposed here, flexibly extracts and enhances each individual character in the image. In this way, the character extraction method can adapt to the particular intensity values of each character and at the same time provides a document image segmentation that is free from errors such as joined characters. This is of great importance to subsequent recognition stages since great part of the recognition accuracy is affected by segmentation errors.

### 4.3 Character Position Location

There are a number of situations where wrongly segmented characters lead to recognition errors. In particular characters in blurred degraded documents might be mis-segmented due to broken, or blurred wrongly connected strokes. Most OCR packages are not prepared for poor quality documents with such noisy background and characters. In this section a description of the segmentation techniques used here will be given. Two segmentation approaches will be discussed and compared. The first one is based on a *peak-valley* analysis of the projected image. The second method uses the *regularly spaced* feature of typewritten characters to aid finding the segmentation positions.

#### 4.3.1 Projection profiles

The projection profile of a given image consists of a histogram created by accumulating the grey-level values of the pixels for each row or column depending on whether it is a vertical or a horizontal projection.

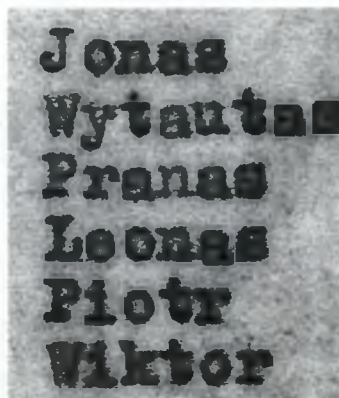


Figure 4.2: Sample image we want to segment

The vertical projection profile of an image is computed by adding all the pixels in each row of the given image, that is, projecting horizontally. In the same way, but projecting through the images columns, horizontal projection profiles are created.

In order to ease the visualization, the accumulated values of the pixels in the profiles have been inverted to show the page contents (lines and characters) as peaks and the background as the valleys between them. The profile histograms are then analysed to find suitable valleys to determine the text line segmentation points. These points should be significant local minima to avoid segmenting through text lines or characters.

Once the image projection profiles are obtained, the segmentation algorithm can look for suitable segmentation points. Both Vertical and Horizontal projection profiles of the sample image in Figure 4.2 can be seen in Figure 4.3.

The peak heights in the profiles can vary according to the number of characters in each line, for the vertical projection, or the number of text lines in the document, for the horizontal projection. In addition, they can also vary with additive noise or the character's intensity strength.

Projection profile-based segmentation for highly degraded documents does not always perform as expected. These images have generally poor contrast, blurred edges and large amounts of noise that can make the peaks indistinct and confuse the segmentation algorithm. In the Figures 4.4(a) and 4.4(b) an example of a degraded image, with faded characters (on the left hand side of the image) and its horizontal projection profile can be seen. In this case,

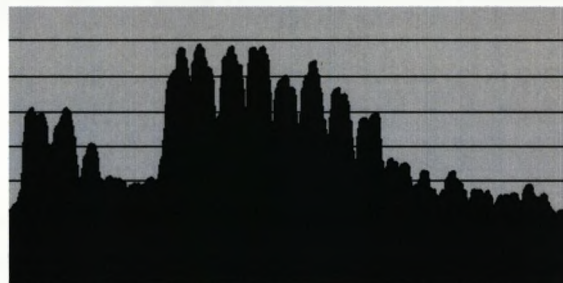




Figure 4.3: Vertical and Horizontal Projection Profiles of the image in Figure 4.2



(a) Difficult image with irregular profile



(b) Horizontal Profile of the image in Figure 4.4(a)

Figure 4.4: Degraded image and its horizontal projection affected by fade characters

the profile is affected by the faded characters and the noisy background creating peaks in the histogram that are as high as the peaks of some of the areas containing characters (on the right hand side of the profile). These variations



are difficult to predict, and are mainly due to degradation. Although the valleys representing characters can be identified visually in the profile, the image could be enhanced to simplify the identification process. A solution to it is to pre-process the image to reduce the noise, and increase the contrast to aid the segmentation algorithm. Possible image pre-processing techniques to improve segmentation are discussed next.

#### 4.3.1.1 Segmentation-oriented image pre-processing

The aim of this pre-processing step is to increase the contrast, discarding as much background as possible, maintaining character dimensions.

It aims to create a clearer projection profile to aid the segmentation process. The resulting image will be discarded afterwards. As discussed in Section 4.1 and Chapter 2, most thresholding methods perform poorly in these documents. Due to the requirements of this enhancement whose only goal is the improvement of the contrast while, at the same time, maintaining the character size, a histogram transformation was found appropriate. The aim of the transformation is to stretch the histogram to produce a bimodal, balanced histogram, increasing the image contrast. To transform the input histogram into the desired one, an Sigmoid function was applied. This "S" shaped function stretches the values of the histogram towards the lighter and darker areas increasing like this the image contrast. Low contrasted images present a non balanced histogram containing most values in their centre region. A Sigmoid function was chosen due to the nature of its shape; the values at its edges are stretched more than the ones in its centre region. By being gentler in the stretching of its centre region the resulting image will increase its contrast but those values in the centre will not be forced to either side avoiding like that over or under thresholded pixels as much as possible. The Sigmoid function has the formula:

$$\text{Sigmoid}(\sigma) = \frac{1}{1 + e^{-\sigma}}$$

where  $\sigma = 1.2$ . The resulting curve can be seen in Figure 4.5. The value of  $\sigma$  was selected after experimentation. A larger  $\sigma$  value would create a curve with a steeper gradient, stretching the histogram too much and in the same way, a lower  $\sigma$  value would output a smooth curve that would not stretch the histogram enough.

In Figure 4.6 an example of an image and its histogram are shown before

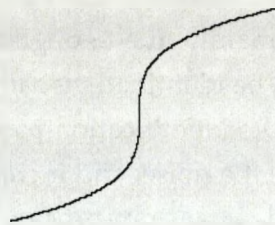
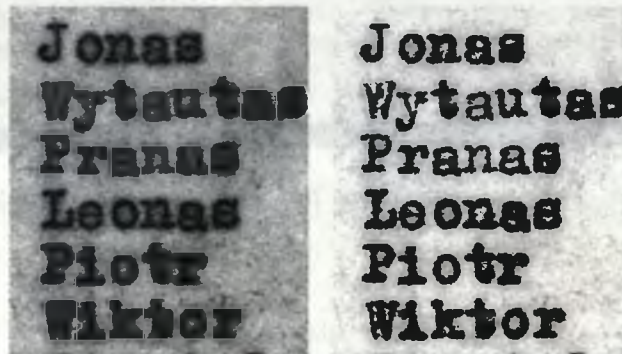
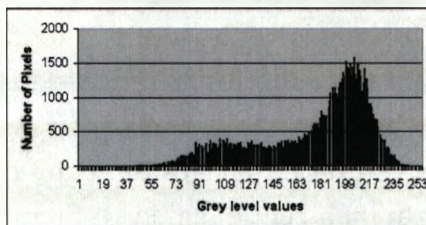


Figure 4.5: Sigmoid curve used for the stretching.  $\sigma = 1.2$

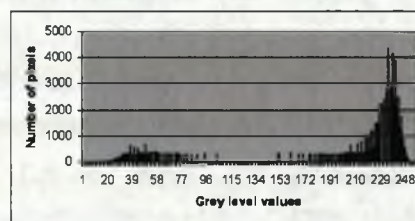


(a) Low contrast image

(b) Stretched Image resulting after histogram transformation



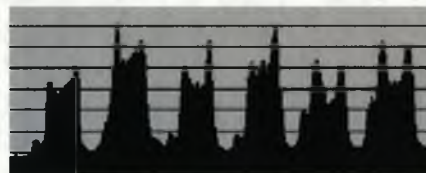
(c) Greylevel histogram.



(d) Greylevel histogram after histogram transformation.



(e) Vertical Profile before transformation



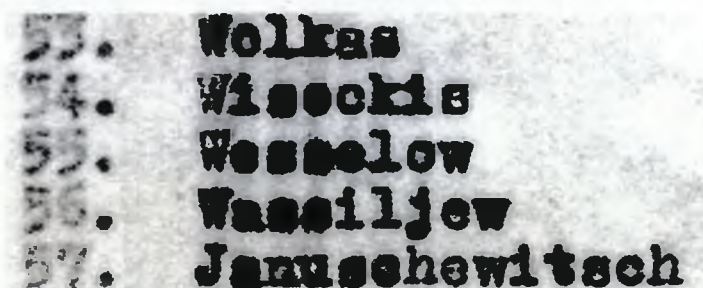
(f) Vertical Profile after transformation

Figure 4.6: Image with its greylevel histograms and vertical projection profiles before and after histogram transformation

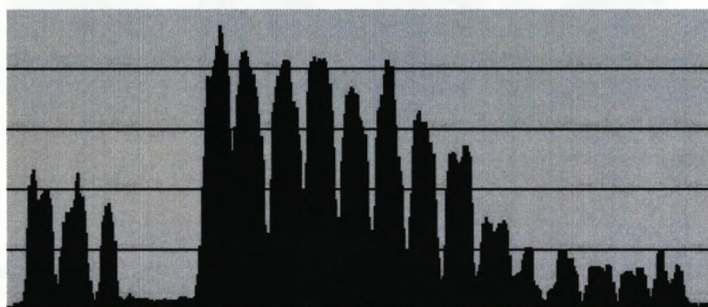
and after the histogram stretching. A clearer bimodal histogram results from the stretching.

Figure 4.7 shows the image previously discussed in Figure 4.4 as a result

of the histogram stretching. The projection profile now shows much clearer peaks that can be segmented in a more straightforward way.



(a) Difficult image with irregular profile after stretching



(b) Horizontal Profile after transformation

Figure 4.7: Comparison of horizontal profiles before and after histogram transformation

### 4.3.2 Projection profile analysis

Projection profile based segmentation consists in analysing the projection profiles in search for suitable minimal points. A peak-valley segmentation algorithm compares the minima points found, their position, their value and the distance to the next minimum and decides the best segmentation points.

#### 4.3.2.1 Text Lines Segmentation

The location of text lines is the first step in the segmentation process. Text lines can be easily identified in the vertical profile of the image. The generous spacing between text lines in the document aids the segmentation, since in general, the valleys in the profile are wide, and peaks are high due to the accumulation of the characters that belong to the text line. The text line segmentation algorithm finds suitable minima in the vertical projection profile of the image and



selects them as segmentation points. The resulting image segmented into text lines is shown in Figure 4.8

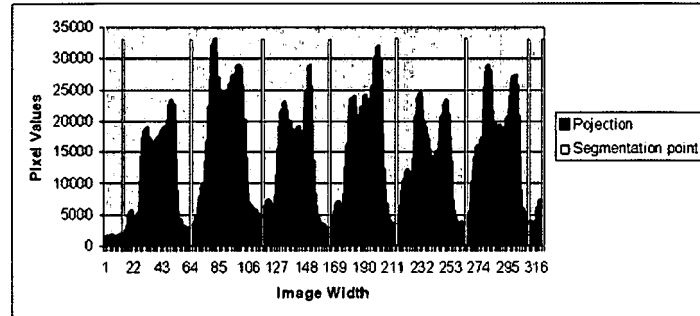


Figure 4.8: Vertical projection of image in Figure 4.2 with segmentation points.

#### 4.3.2.2 Character Segmentation

Once the text lines are located they need to be segmented into characters. The horizontal profile of the image can be analysed for minimal points in the same way. Ideally, in a document where all the text lines are skew free and parallel to each other, all characters should be perfectly aligned in columns. Then segmentation points could be found for the whole document by segmenting the image based only on its horizontal profile. This could be affected by taking the paper out of the typewriter machine and reinserting again at an angle.

Due to the proximity between characters, the blurred edges and the noisy background, the character segmentation from the horizontal profile of the whole image is not always accurate. In addition, the presence of skew could affect the projection profile. In most cases, the minimal segmentation point would step on some character edges leading to possible loss of character information.

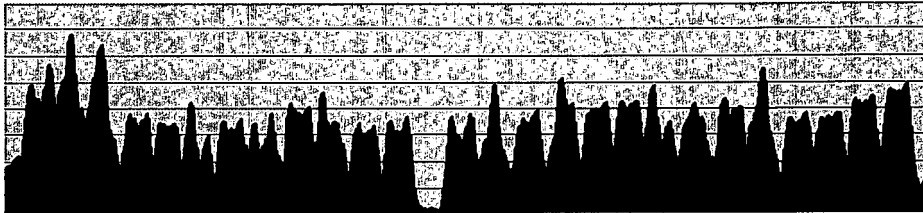
In order to avoid such kind of errors, the character segmentation is performed for each text line individually. When projecting only one text line at a time, the amount of noise and the skew between characters are both reduced considerably.

Although the extracted text lines are likely to contain less background than the whole image, and therefore have clearer profiles, due to the close proximity between characters, the text lines should also be enhanced. The histogram stretching transformation used before is equally effective for the segmented text lines, as well. The same operation is then repeated for each segmented

text line image. As mentioned earlier, the enhancement of the whole image previously performed is discarded. Now, each text line image is enhanced individually to allow for better adaptation to each text line intensity values. Figure 4.9 shows the extracted text line from a document image and its horizontal profile.



(a) A segmented text line



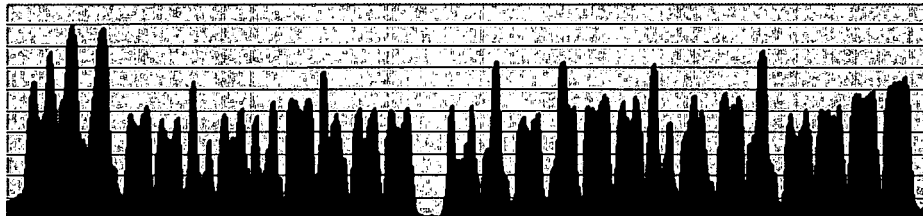
(b) Horizontal profile of text line in Figure 4.9(a)

Figure 4.9: A segmented line and its horizontal profile.

In Figure 4.10 the same example is shown after the histogram transformation.



(a) A segmented text line after histogram transformation pre-processing



(b) Horizontal profile of text line in Subfigure 4.10(a)

Figure 4.10: A segmented line and its horizontal profile after pre-processing.

To aid the segmentation algorithm, other information such as minimum and maximum character sizes are used. These values can be estimated considering the reduced number of fonts from different typewriters and the scanning resolution. Minimum and maximum character sizes are used to discard erroneous, very small or very large segmentation points. After performing several tests on documents created with different typewriters, but scanned at the

same resolution, 300 ppi, the average character size was estimated to be of 12 pt. Around 25 by 25 pixels for a small lower-case character, such as 'a' or 'n' and around 25 by 35 pixels for all capitals and for lower-case characters with ascenders or descenders, as 'l' or 'g' for example. Other sizes are also found such as those for punctuation symbols. The smallest one, the dot '.', is usually 15 by 15 pixels.

Experiments have been performed with and without the use of this information. The results show that the use of this information can help the segmentation algorithm, although there are still some cases where the segmentation fails breaking characters in two. Mis-segmented characters are generally due to faded strokes that can appear in the profile as a wide valley misleading the algorithm. Other segmentation errors are cut character edges. An example of the image in Figure 4.9(a) is shown in Figure 4.11. In this image, segmentation errors can be found in characters 's' in the first word and 'M' in the second word, among others.



Figure 4.11: Image in Figure 4.9(a) with segmentation points.

The results of the segmentation algorithm based only on the projection profiles were insufficient and another method was required.

### 4.3.3 Fixed-grid segmentation

Typewritten font, and the way the typewriter operates, by moving horizontally and vertically along the paper are also known as monospace. That is, *ideally*, all characters in the document should fit within a fixed grid. The grid size can be calculated given the character dimensions, and it can be fitted into the document by convolving the given grid through the projection profile and finding the position with minimum cost. Unfortunately the actual typewriter grid is often irregular. This irregularity is generally a result of the paper being removed and reinserted during document creation or due to adjustments of the paper position by the user (e.g. to erase a character or to add more space between lines). Skew can also affect the grid calculation and fitting. The use for segmentation of a fixed typewriter grid for the whole document would probably not lead to a perfect fit.

Although a fixed grid might not be the best approach to segmenting the whole document, when applied to each segmented line individually, it does reduce the chances of failure.

In order to compute the grid, information on the characters' size (in pixels or points) can be used. As explained in the previous section, this can be estimated from the resolution used during the acquisition and the typewriter font.

To calculate the grid size and fit it in the right places a simple algorithm was implemented. The algorithm takes a range of possible character sizes and for each possible grid size tries to fit it into the text line horizontal profile calculating the cost at each segmentation point. For each size, the minimal cost is selected and compared to the rest. The minimal cost is then selected among all the possible sizes. In this case, a range of 25-35 pixels for the character width was given to the algorithm with the average value being 30 pixels.

The implementation of this method and its performance are known to be very reliable in segmenting typewritten characters. However, due to irregularities in the way the documents were produced; such as the presence of skew, the highly textured background or blurred characters, could lead to a poor grid fit and the segmentation points would step on the character edges. Figure 4.12 shows an example of a segmented text line, where there are a number of characters where the segmentation points fall very nearby or on its edge.



Figure 4.12: Image in Figure 4.9(a) with fixed segmentation points.

To avoid that, a flexible grid segmentation is proposed, where, after the optimal typewriter grid is found and placed, the segmentation points are one by one evaluated and shifted left or right towards their local minimum if it is not on it. The shifting will never be more than a couple of pixels in either direction. Figure 4.13 shows an example the segmented text line image using the fixed but flexible segmentation algorithm. A more detailed example is shown in Figure 4.14, where the fixed segmentation is shown in red and the new flexible one is shown in blue.



Figure 4.13: Image in Figure 4.9(a) with fixed segmentation points that have been flexibly adjusted to their local minimum to avoid overlapping character edges.

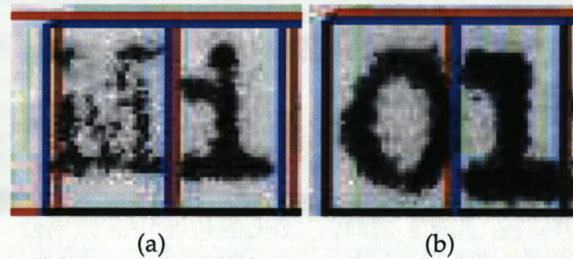


Figure 4.14: Examples of improved segmentation.

#### 4.3.3.1 Character Position Validation

At this point, the image has been segmented into individual characters. The segmented areas are then analysed and discarded when they only contain background. This is done again, by analysing the histogram of the segmented area. The range of values in the segmented area needs to contain a minimum of dark values. In Figure 4.15 two different segmented areas are shown, one containing a character and one containing only background.

When the segmentation is finished, the pre-processed image is discarded and only the segmented areas containing characters are stored. When the histogram transformation approach, as used earlier to pre-process the image to aid the segmentation, was applied to the whole document, it failed to stretch the intensity values of differently pressed characters. Now that each character has been extracted from the document, each region's histogram contains only the intensity of one character. The results from the application of the histogram stretching to each segmented area do not change much when the characters show good contrast (see Figure 4.16). However, when individually stretching a lightly pressed character, histogram stretching can adapt to the low intensity range of values and highlight the strokes much better than when carried out globally for the whole image.

The result of the histogram stretching applied to the whole image and the result of the same applied to each segmented character can be seen in Figure 4.17. Note the faded characters on the left side of the image in Figure 4.17(a)



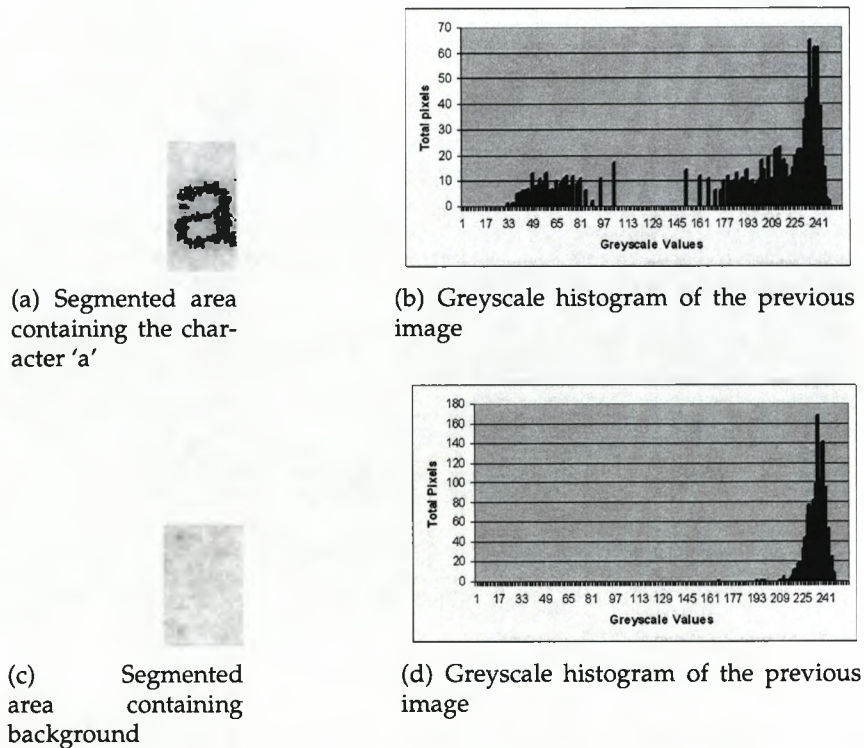


Figure 4.15: Two segmented areas

are now equally dark as the rest.

## 4.4 Character Localisation

At this stage where the character strokes are brought to a similar intensity level, after the individual histogram stretching, and maintain their real dimensions, a tighter localization within the location (segmentation) box will be performed. This will allow reducing the background around each character.

To localise the characters a simple approach was followed. This involves closing down the outer box until an edge of the character is found. A sensitivity to noise factor has been chosen to discard noise connected components smaller than 3 by 3 pixels. The localisation algorithm also uses a threshold value to distinguish a character stroke from the background. Since the image has been stretched, and its histogram is almost bimodal, an easy threshold value of '100' has been used leading to good results.

We can now confidently confirm the number of characters, their location

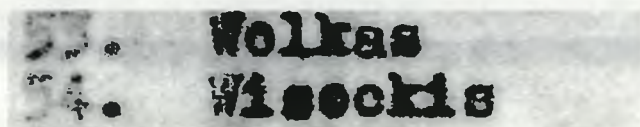


(a) Character 'a' after Histogram Stretching was applied only to the whole area.

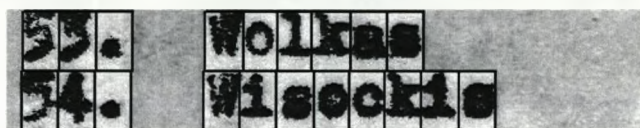


(b) Character 'a' after Histogram Stretching applied to the segmented area.

Figure 4.16: Histogram Stretched image 'a'



(a) Globally stretched image.



(b) Stretched image after segmentation.

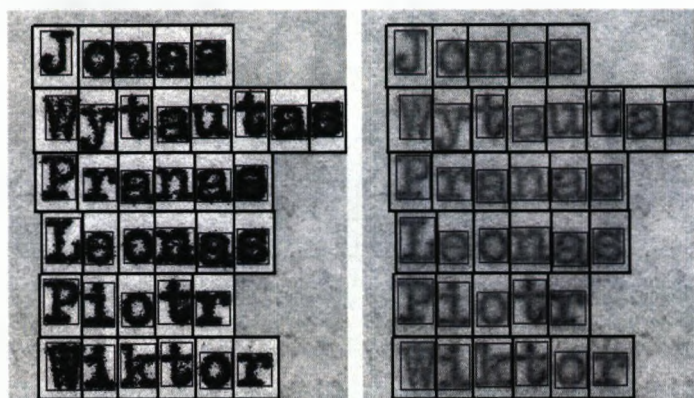
Figure 4.17: Stretched image

and their dimensions. All the previous enhancements used to aid to reach this stage can be now discarded and the extraction of each character can take place from the original image. See Figure 4.18.

## 4.5 Combination Thresholding

At this point each character has been segmented and localised and is ready to be thresholded. The thresholding applied to these characters must ensure that neither any faint strokes are lost nor any character holes and cavities are filled in. To maximise the flexibility this stage applies two methods in parallel to each localised bounding box and adaptively combines their results. The first objective is to repair broken characters by identifying those parts of the character strokes that can be confidently determined as foreground and to reliably indicate those parts of the strokes that could be merged to complete each character. The second objective is to avoid merging any background pixels with the character strokes as that will damage further the dark and blurred characters.





(a) Stretched localised image. (b) Localised image after discarding the pre-processed image.

Figure 4.18: Localised images

#### 4.5.1 Cautious Ternarisation (CT)

This method was developed to classify pixels within the localised bounding box into three categories: pixels that can be confidently considered as part of the strokes, those that can be confidently considered as background and those that cannot be confidently classified outright.

To determine the pixels that belong to the background category, the image will be analysed based on the individual areas extracted. In Figure 4.19 each of the differentiated areas is shown and named. In the image, the background that is outside of the segmentation boxes (in blue) will be referred to as the *Segmentation Background*. The contents of the segmentation boxes will be referred to as the *Segmentation Foreground*. Inside the *Segmentation Foreground*, the contents of the character localisation boxes will be referred to as the *Localised Foreground* (in red), and the smaller area of background outside of it as the *Complement Background*.

By localising the characters, most of the background areas can be analysed and afterwards excluded.

The *Segmented Background* is comprised of the pixels that do not belong to any of the characters, therefore contain only background values. The *Segmented Foreground* is composed of the pixels inside the segmentation areas. In Figure 4.20 the intensity histogram of such areas are plotted. To create the histograms 5 different images containing approximately 600 characters have been

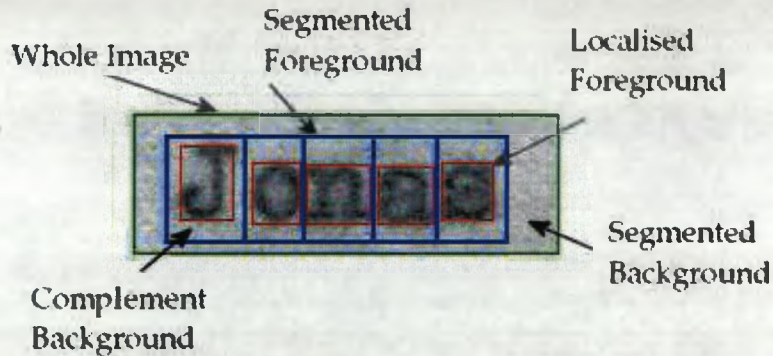


Figure 4.19: Localised character image

used. The given histogram therefore represents an average representation of their *Segmented Background* and *Segmented Foreground*.

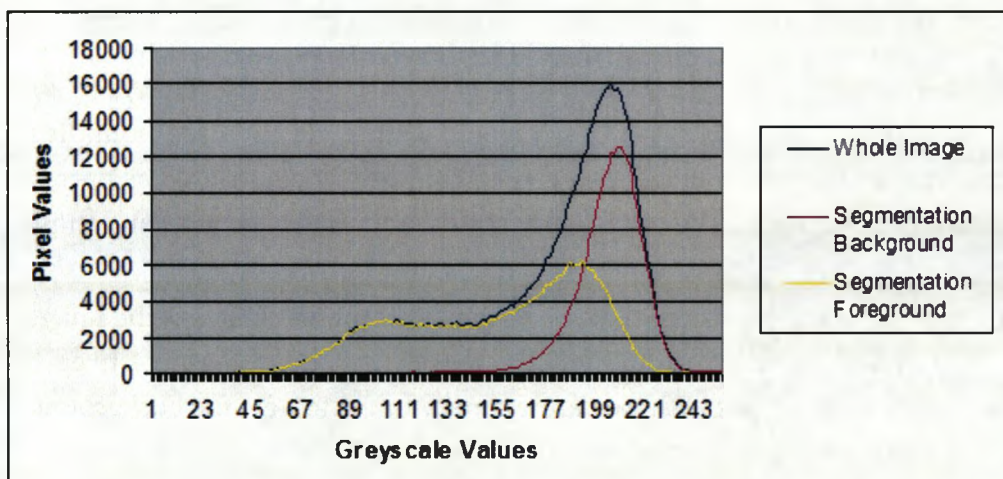


Figure 4.20: Greyscale histogram of the Segmentation Foreground and Background.

The *Segmented Foreground* contains the characters, or *Localised Foreground*, but also some background, the *Complement Background*.

The greyscale histogram of those regions is shown in Figure 4.21. Although, the characters have been localised to exclude as much background as possible, the *Localised Foreground* still contains background pixels that surround the character. This can be seen in the histogram in Figure 4.21 and an overlapping.

After studying several characters it was determined that the amount of background contained in the *Localised Foreground* area depends entirely on the



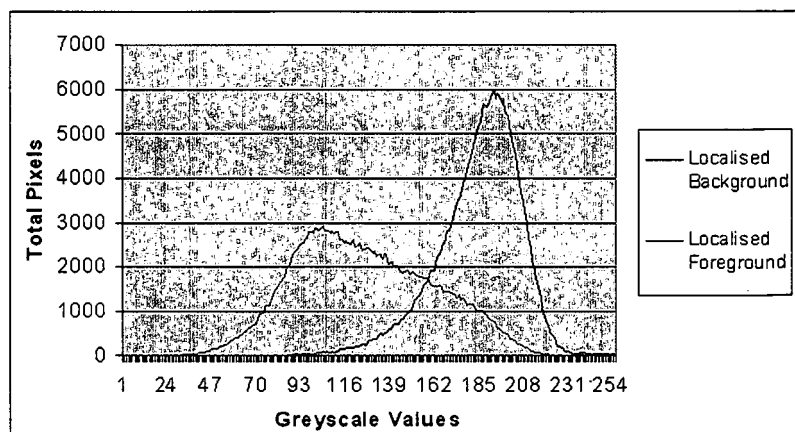


Figure 4.21: Histograms of *Localised foreground* and background regions

character shape and the size of inner holes. When inner holes are small, the effect of edge blur is more prominent. A very good example of this are characters 'a' or 's', where their outer edges and inner holes are very close to each other. The localised character image in these two instances contains, in general, more blur than that of other characters like 'i' or 'o', for example. In Figure 4.22 several characters are compared by the amount of background contained in their localised character image.

Given the information obtained from the histograms of the different segmented areas, we can estimate three ranges of gray values: values that generally belong to the *Complement Background*, (i.e. out of the localisation box), values that are only present inside the *Localised Foreground*, and some values that are in both areas. The ternarisation will only be applied to the *Localised Foreground* area.

The first step involves the extraction of the background pixels. This is done by analysing the pixel values that are present in both the *Complement Background* and the *Localised Foreground* areas. That is, the common area in the histogram. When the common pixel values are found in the *Localised Foreground* area will be turned to white. This is calculated individually for each character to accommodate differences between them. In Figure 4.23 a sample character 'i' is analysed. A histogram displaying the values of its *Localised Foreground* (in pink) area and its *Complement Background* (in blue) is shown.

The area in black corresponds to the number of common pixels for both regions.

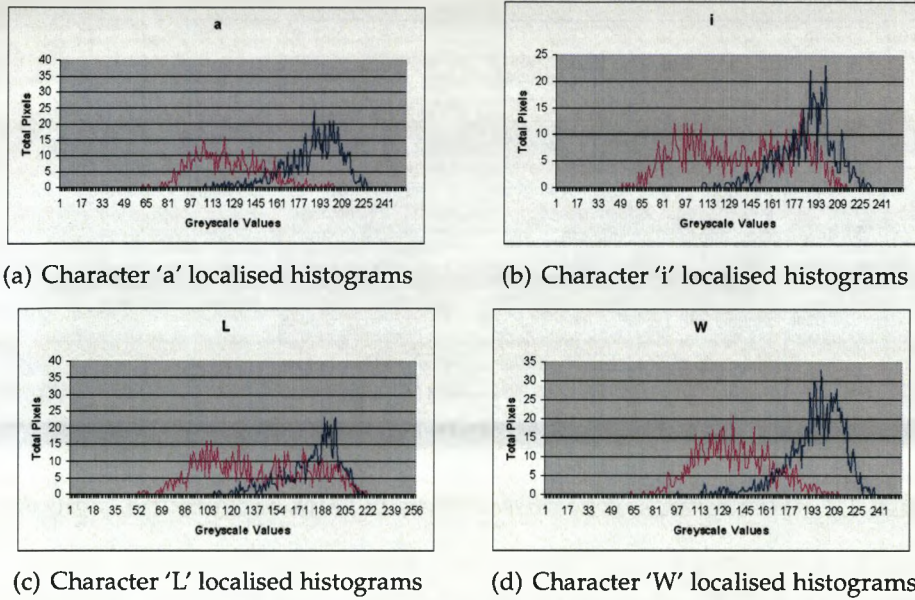


Figure 4.22: Histograms of several characters representing foreground and background

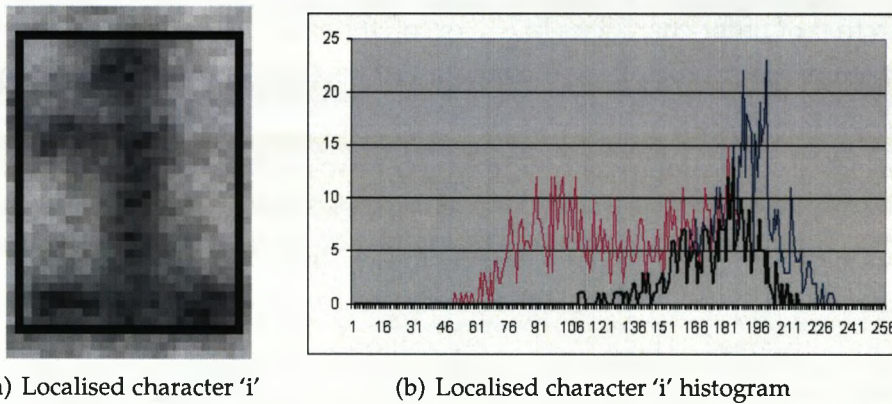


Figure 4.23: Localised character 'i' with intensity histograms of its Localised Foreground and Complemented Background

The sum of all the pixels in the common area is then compared to the total pixels in the *Localised Foreground*. For character 'i', the total number of pixels in the common area add up to 325 from the total 875. That makes 37.14% of the total. This percentage is then used to threshold the character. For character 'i' almost 38% of its lightest pixels belong to the background and therefore will be turned to white. To find out the threshold value, the histogram in analysed



from the lightest end by accumulating the amount of pixels found in the histogram one value at a time until the 38% of the total is found. The threshold value for character 'i' background was calculated to be 156. The results of this threshold can be seen in Figure 4.24.

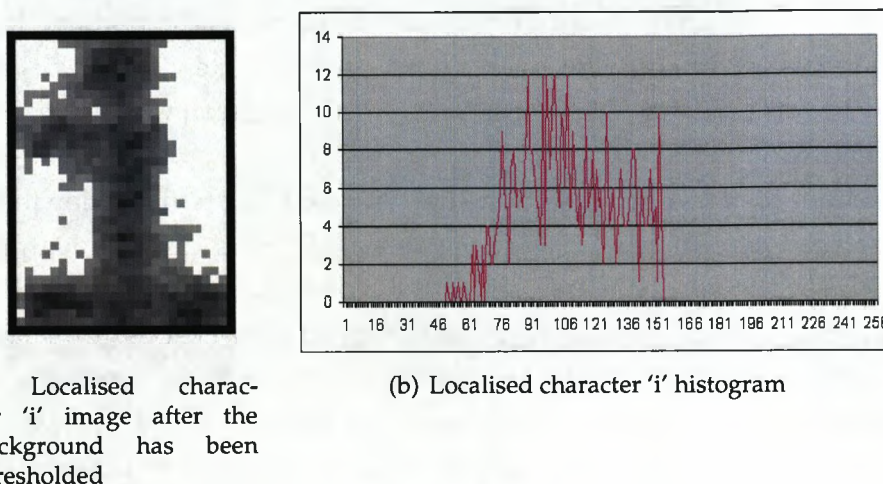


Figure 4.24: Localised character 'i' with intensity histograms of its Localised Foreground after background threshold

This operation is applied to each individual character to adapt to their own intensity values. The results after thresholding the background away using this values is shown in Figure 4.25.

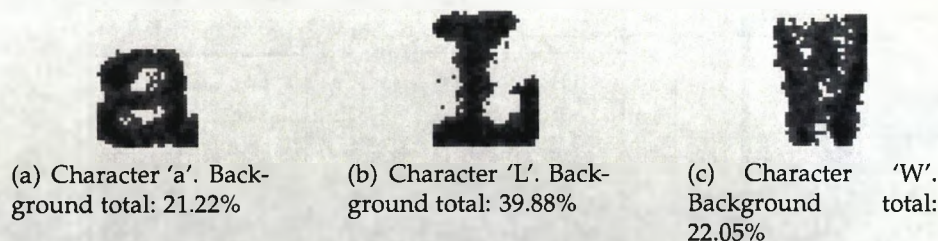


Figure 4.25: Histograms of several characters after background thresholding

After the background is thresholded, the next step is the extraction of the foreground. The background has been extracted and the remaining pixels belong either to the character stroke or are uncertain. The average area that the background covers was found to be 25%; this leaves an average 75% for the foreground and the uncertain area.

After studying several samples of typewritten characters thresholded and

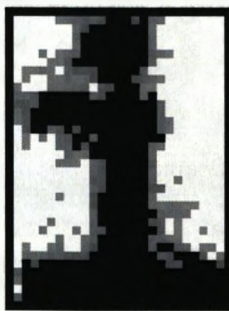
edited manually, it was observed that their foreground was occupying a percentage in the range between 40-60% of the total character area. Capital characters and those with ascenders present thicker strokes, but the area of their bounding box is larger than the one taken by smaller characters.

Even when carefully typing characters on clean white paper, their strokes do not display the same intensity throughout but a greyscale mixture. In all cases, the centre of the character strokes possesses the darkest values, smoothing away towards the edges. This fact can be used in a global way to highlight the central stroke.

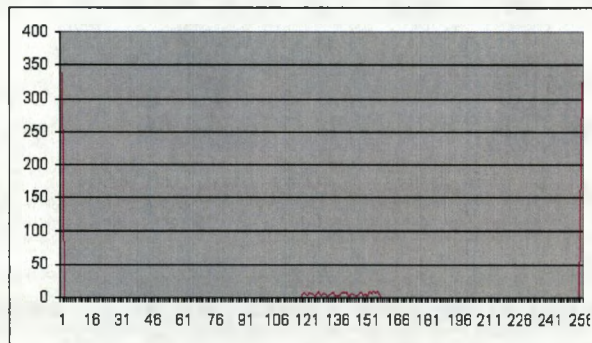
As mentioned earlier in this section, the idea behind the ternarisation thresholding method is to threshold away the areas where we are confident, and leave the rest as an uncertain area, not taking a decision just yet. Based on the analysed ground-truth characters, it was estimated that most perfect characters foreground is formed by at least 40% of all its pixels. In many cases, it was shown that the perfect characters could cover up to 60% of the total but since the intention is only to highlight the central section of the characters, the chosen threshold is left to 40%. That is, the darkest 40% of the pixels of each character are turned to black.

**Olschanskas Nietzsche**

Figure 4.26: Text line after CT.



(a) Localised character 'i' image after CT



(b) Localised character 'i' histogram after CT

Figure 4.27: Localised character 'i' with intensity histograms of its Localised Foreground after CT

In Figure 4.26 the results of the cautious ternarisation threshold are shown.



This chosen threshold effectively enhances the central area of the character strokes as seen in Figure 4.27.

However, as the threshold values are applied to the whole *Localised Foreground* area, CT does not connect strokes, where degradation has altered the intensity of the central section of the stroke. This is a known drawback of the utilisation of globally applied methods<sup>1</sup>. However, in this case, the output has the advantage of keeping the uncertain area which leaves further options open. It is important to highlight that the aim of this thresholding method is to avoid making mistakes. In that way the pixels that have been turned to black or white are confidently correct.

The output of this ternarisation threshold is a three-level image consisting on the *Foreground* area, (now in black), the *Background* area, (now in white), and the *Uncertain* area (left untouched). This method's strengths are the good extraction of the 'real' character strokes and the clearance of part of the background. The central area of the stroke is highlighted, some background is discarded and the remaining area is left untouched as uncertain. Since this is performed globally for each character, some areas of the strokes appear broken, although they are highlighting what is certainly foreground and are not affected by the background noise introducing erroneous data.

#### 4.5.2 Aggressive binarisation (AB)

To complement CT, an adaptive method was used. A good example of such method is that of Sauvola and Pietikainen [76]. Hereafter called SP. As mentioned in Chapter 2, the SP method is a specialisation of Niblack [59]. The



Figure 4.28: Text line after SP threshold.

image in Figure 4.28 shows a sample of an image after the SP threshold is applied to it. The same image after SP has been applied to each of its localised characters is shown in Figure 4.29. A great improvement is shown in the performance of the SP method when it is applied after excluding the background. Although the result is better than before, it does still create a large amount of

<sup>1</sup>Note the threshold here is applied globally to a local area.

## Oli sohauskas Mietschyslowas

Figure 4.29: Text line after SP's threshold was applied to each Localised Character.

noise. A solution to reduce the sensitiveness to noise is to smooth the image before SP is applied. Several smoothing algorithms were applied and after some tests a Gaussian filter was chosen. The Gaussian was a 5 by 5 filter described as follows:

0	0	0	0	0	0	0
0	1	4	7	4	1	0
0	4	16	26	16	4	0
0	7	26	41	26	7	0
0	4	16	26	16	4	0
0	1	4	7	4	1	0
0	0	0	0	0	0	0

The result of the smoothed SP output is shown in Figure 4.30.

## Oli sohauskas Mietschyslowas

Figure 4.30: Text line after SP threshold was applied to each *Localised Character* after being smoothed.

The new bi-level image produced by AB shows adaptation to intensity changes in the character strokes and the output are better connected. However, when the stroke edges are close to each other and blurred, the resulting AB image tends to merge them resulting in undesired closed holes that would mislead the subsequent recognition stage. The images shown in Figure 4.31 display a number of characters where AB results in touching strokes or adds extraneous pixels to their shape.

### 4.5.3 Results Combination

#### Combination Thresholding Results Combination

Both the CT and the AB methods are designed for different purposes. The CT method provides reliable results but it is cautious in not joining any strokes when in doubt. On the other hand, the AB method aggressively joins broken

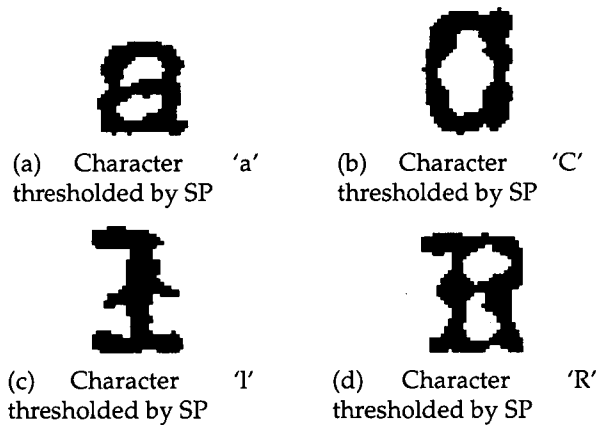


Figure 4.31: Several character images after SP threshold

strokes but also tends to join strokes that should not be connected. For example, the free ends of 'a' or 's' result connected to the main body of the characters. A number of deformations on the strokes are also observed in the results of the AB method. The result-combination step aims to draw from the advantages of each method and produce the final result by comparing corresponding pixels in each resulting localised character bounding box.

The rationale of the combination strategy is as follows. If the CT and AB pixels are both either black (foreground) or white (background), the final result is that value (effectively the value of the CT). The CT method is very reliable

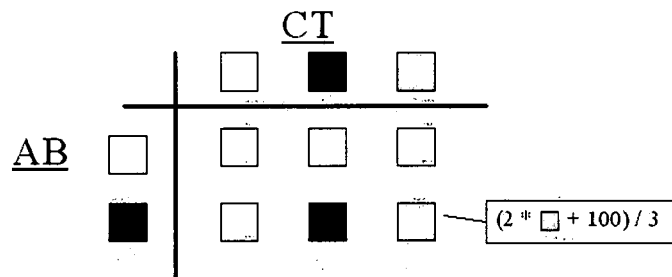


Figure 4.32: Combination table rules.

in outlining the character strokes and discarding the background. Therefore, when the AB pixel is black and the CT pixel is white, then the black pixel in AB is probably noise and the CT method is trusted. The resulting pixel will be white. If the CT pixel is grey (denoting uncertainty whether it is foreground or background) and the AB pixel is black, then the result will be a weighted mean of the CT and AB pixel values. The mean is weighted towards the value

of the CT pixel in order to be careful not to combine strokes that should not be together. The new pixel is calculated as  $(2*CT + 100)/3$ . Note that the AB value is 'represented' by a value of 100 as it is, naturally, 0.

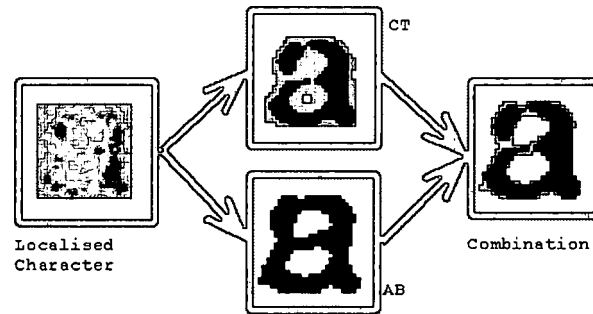


Figure 4.33: Combination threshold illustration.

In the cases where the AB pixel is white and the CT pixels are either grey or black, then the resulting pixel will be white. This decision is made as the AB methods tend to err towards producing more black pixels, and therefore, the fewer whites are trusted more. A table describing the combination strategy

## Olischauskas Mietschyslowas

Figure 4.34: Resulting image after combination of methods CT and AB

is shown in Figure 4.32 and an illustration of the resulting image is shown in Figure 4.33. An example of the combination result is shown in Figure 4.34.

### 4.5.4 Results and discussion

The methods here proposed provide a flexible text recovering system that extracts the characters in a documents image and enhances them individually. The combination thresholding method was designed to overcome the difficulties posed by degraded typewritten documents by flexibly analysing each individual character and cautiously repairing it. The output character images preserve ambiguous areas in order to allow more flexibility and provide with reliable information to the subsequent recognition stage.

The results from this method allow flexible interpretation of the areas of uncertainty by the recognition system. This is an advantage compared to the

recognition of a bi-level image with broken or touching strokes. When a bi-level image is recognised, there is no room for interpretation, and all the given black pixels will be used as the true part of the character. This is where the flexible method here proposed has an advantage.

In Figure 4.35 a number of characters are shown to highlight the results of the method.

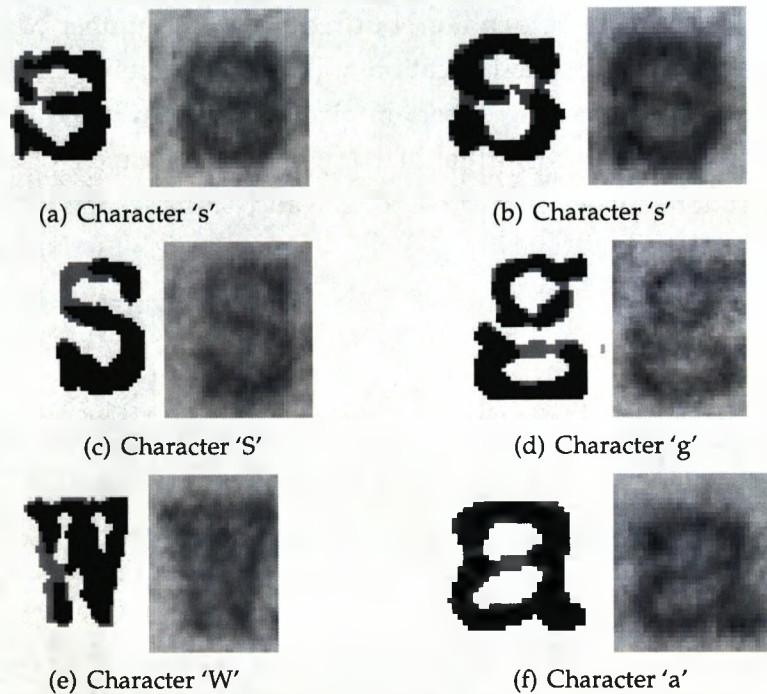


Figure 4.35: Flexibly recovered sample characters

## 4.6 Flexible Stroke Refinement

The image obtained from the previous steps contains pixels of three possible values. The background, the foreground and the grey, uncertain area. This uncertain area is the key of the flexible text recovery. It has been kept as grey since we were not able to classify it as foreground or background. Discarding it completely might end up in many characters being broken or very noisy. On the other hand, adding it all to the foreground will thicken the characters and, in many cases, wrongly connect strokes hiding valuable features.

The task of the flexible stroke refinement consists of reducing the uncertain third level in the image to minimise noise and errors and at the same time keeping the valuable key ambiguous points.

After analysing the resulting recovered images, a way to separate the third, uncertain level into three was conceived. One of the aims of this refinement was intended to locate the areas where characters were either broken or had wrongly connected strokes. Differentiating between these two key areas cannot be achieved without recognition, but reducing the number of uncertain pixels to give a more accurate indication of possible repairs (to an intelligent OCR package). The remaining uncertain pixels could be divided by whether they can be considered part of the stroke or whether they can be discarded.

For illustration, images in Figure 4.36, 4.38 and 4.41 are shown.

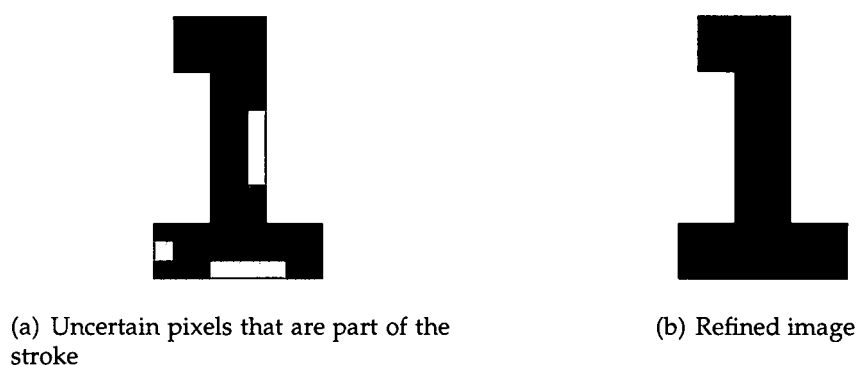


Figure 4.36: Uncertain pixels selected as part of the stroke

In Figure 4.36, the grey pixels are situated within the character edges. In most cases it can be said that this uncertain pixels should belong to the character strokes. The method used, looks for black pixels in both ends of the grey connected component, in either vertical or horizontal orientation. This area will be referred to as *Uncertain Edge Area (UEA)*.

Following, in Figure 4.37 the pseudocode for the algorithm to locate the UEA is shown.

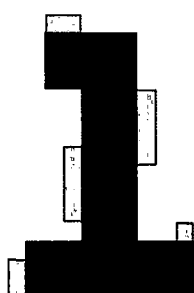
On the other hand, grey values that are standing out of the character edges tend to be additive noise and can be discarded. This pixels, referred to as the *Uncertain Noise Area (UNA)*, can be identified if white pixels in two, three or four of its opposite edges are found. Figure 4.38 shows an example image showing the mentioned uncertain areas, and Figure 4.39 shows the pseudo

```

1. if (pixel == GREY-AREA)
2. {
3.     find nearest non-grey pixel towards TOP → pixel-TOP
4.     find nearest non-grey pixel towards BOTTOM → pixel-BOTTOM
5.     find nearest non-grey pixel towards LEFT → pixel-LEFT
6.     find nearest non-grey pixel towards RIGHT → pixel-RIGHT
7.     if (((pixel-TOP == pixel-BOTTOM) OR (pixel-LEFT == pixel-RIGHT))
        AND (pixel-VALUE == BLACK))
8.     {
9.         pixel = FOREGROUND
10.    }
11. }

```

Figure 4.37: Algorithm to locate pixels that belong to the UEA



(a) Uncertain pixels that are part of the background



(b) Refined image

Figure 4.38: Uncertain pixels selected as part of the background

code for this algorithm.

```

1. if (pixel == GREY-AREA)
2. {
3.     find nearest non-grey pixel towards TOP → pixel-TOP
4.     find nearest non-grey pixel towards BOTTOM → pixel-BOTTOM
5.     find nearest non-grey pixel towards LEFT → pixel-LEFT
6.     find nearest non-grey pixel towards RIGHT → pixel-RIGHT
7.     if (((pixel-TOP == pixel-BOTTOM) OR (pixel-LEFT == pixel-RIGHT))
        AND (pixel-VALUE == WHITE))
8.     {
9.         pixel = BACKGROUND
10.    }
11. }

```

Figure 4.39: Algorithm to locate uncertain pixels that belong to the background

The remaining area to be identified is the *Uncertain Key Area (UKA)*. This last area is of great importance since it describes character areas that are either broken or wrongly connected. The UKA is located by following a combination of the two previous algorithms. That is, the uncertain pixels that are between black pixels in one of the directions (vertically or horizontally) and white pixels

in the other direction belong to the UKA. Figure 4.40 shows the pseudo code for this algorithm.

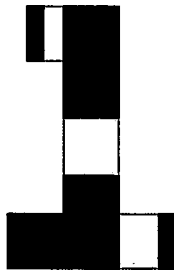
```

1. if (pixel == GREY-AREA)
2.   {
3.     find nearest non-grey pixel towards TOP → pixel-TOP
4.     find nearest non-grey pixel towards BOTTOM → pixel-BOTTOM
5.     find nearest non-grey pixel towards LEFT → pixel-LEFT
6.     find nearest non-grey pixel towards RIGHT → pixel-RIGHT
7.     if (((pixel-TOP == pixel-BOTTOM) AND (pixel-TOP/BOTTOM == WHITE))
        AND ((pixel-LEFT == pixel-RIGHT) AND (pixel-LEFT/RIGHT == BLACK)))
8.       {
9.         pixel = UNCERTAIN-STROKE
10.      }
11.     else
12.       {
13.         if (((pixel-TOP == pixel-BOTTOM) AND (pixel-TOP/BOTTOM == BLACK))
            AND ((pixel-LEFT == pixel-RIGHT) AND (pixel-LEFT/RIGHT == WHITE)))
14.           {
15.             pixel = UNCERTAIN-STROKE
16.           }
17.       }
18.   }

```

Figure 4.40: Algorithm to locate uncertain pixels that belong to the background

Figure 4.41 illustrates two images showing UKAs. In the character image shown in Figure 4.41(a), the grey pixels are clearly part of the character body. On the other hand, the UKA in Figure 4.41(b) is noise, and does not belong to the characters body but makes the character touch itself. How to differentiate these two is a challenging problem that requires recognition.



(a) Uncertain pixels selected as part of a broken stroke.



(b) Uncertain pixels part of the wrongly connected stroke

Figure 4.41: Uncertain pixels that belong, either to the character stroke, i.e. broken character, or are noise, i.e. like a wrongly connected stroke.

Images in Figures 4.36, 4.38 and 4.41 have been manually created to ease this description.



At this point in the refinement, the uncertain areas have been identified and the refinement can begin. The obvious actions would be to set the UEA to black, that is the foreground, and UNA to white, since they are considered noise. Before the refinement begins, some adjustments are made to optimise the results of the character refinement. In order to maximise the positioning and magnitude of the UKA, all the grey pixels that are in their neighbourhood (connectivity 8) will be set to UKA too, regardless if they belong or not to another of the two types. Following, the sequence of steps for the refinement is illustrated with two recovered character images.

A couple of examples of original characters are shown in Figure 4.42. In

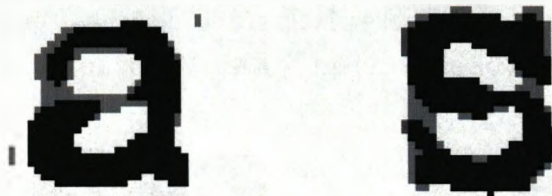


Figure 4.42: Recovered sample characters 'a' and 's'.

Figure 4.43 the same characters are shown with their different uncertain areas in various colours to aid visual understanding. The pixels in green belong to the UNA. This green pixels are believed to be irrelevant and could be discarded later. The pixels in purple belong to the UEA and are believed to be part of the strokes. The orange regions are the UKA that could be either, foreground or background, but, as mentioned earlier, a decision cannot be taken without recognising the characters first. In the next step, the UKA are extended to all

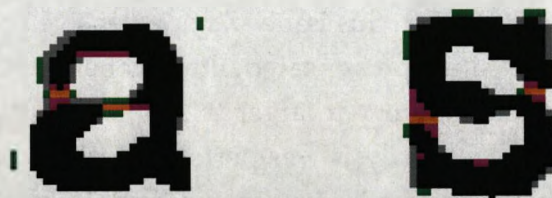


Figure 4.43: Characters 'a' and 's' with uncertain areas located.

its neighbouring pixels that were originally grey. These are turned orange as well. This helps defining the key areas better to cover better part of the broken

stroke or gap. The images in Figure 4.44 show the results of this enhancement.

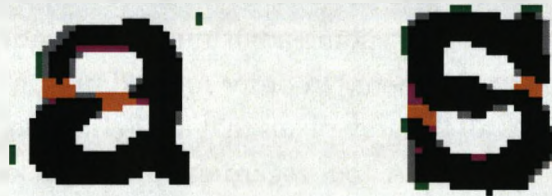


Figure 4.44: Characters 'a' and 's' with uncertain areas located and improved.

The refinement starts by discarding the pixels that are believed to be noise, that is the UNA pixels, in green. These pixels are turned white and an example of the resulting images can be seen in Figure 4.45. When applying and testing

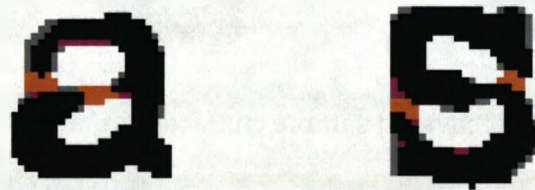


Figure 4.45: Characters 'a' and 's' with UNA pixels set to white.

this enhancement to a large set of character images, some problems with the method appeared. The problem occurred in some characters where degradation had affected regions of the character that were in a "corner" or in an angle where the previous algorithm would 'fail' to classify them as UKA. Those broken regions were classified as UNA but should be part of the character. Figure 4.46 shows two characters with this issue. After extensive observation, it was realised that when this situation appeared, the number of pixels of those regions created a connected component larger than the ones that were real noise. A connected component analysis was performed then to discard only the UNA components that were smaller (width and height) than a set threshold. The threshold was set to  $4 \times 4$  pixel components, where at least 50% of the area covered were UNA pixels.

If any large area is found, it is enhanced, like the case of the UKA regions described previously, by adding its neighbouring grey values. After that, any



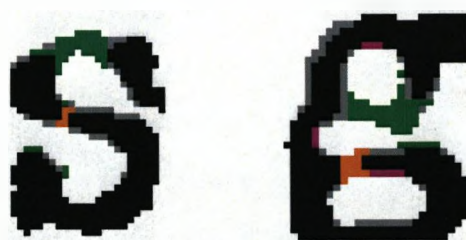


Figure 4.46: Characters 'S' and 'g' showing a broken corner misclassified as UNA.

large UNA component will be added to the foreground, and otherwise deleted. Figure 4.47 displays the previous images outlining the UNA connected components (CC), in red boxes, plus their refined output image, where the problem is solved. The large cc are enhanced and kept. The next region to be refined is

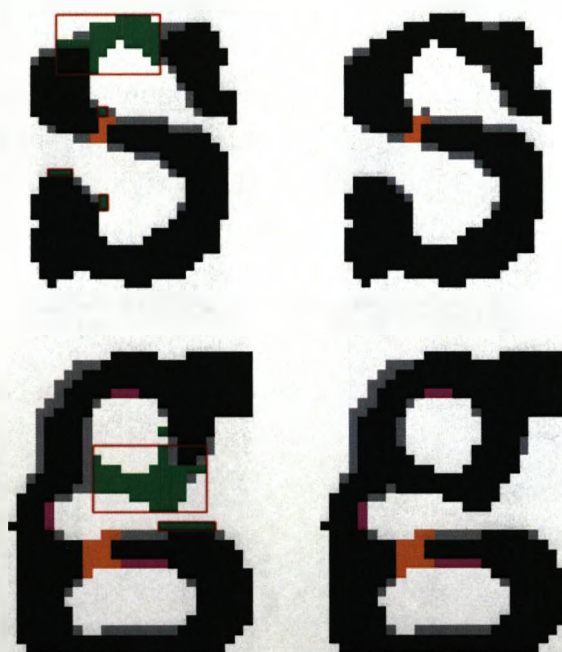


Figure 4.47: Characters 'S' and 'g' showing a broken corner misclassified as UNA.

the UEA. Uncertain pixels that are shown in purple, that are believed to be part of the foreground, will be set to black. Figure 4.48 shows the resulting images. Finally, only the UKAs and the black foreground pixels are kept in the character image. The remaining grey pixels that did not belong to either

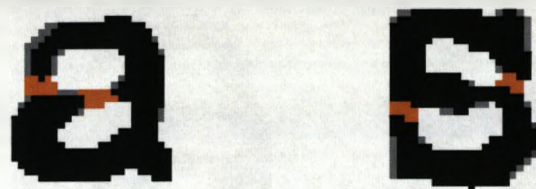


Figure 4.48: Characters 'a' and 's' with UEA pixels set to black.

UEA nor UNA are discarded as well. For a human reader is easy to identify which of the orange, UKA regions should be discarded and which should be kept. This is because the characters are recognised first and then a decision can be made. Attempting to binarise the image, by choosing any of the UKA regions, without recognising it first, could lead to further degradation of the characters. The UKA regions can be combined to create a number of candidate output images. One of them would correspond to the correct character. An option here is to send all this possible images to the OCR, or classifier, and select the one with higher recognition confidence. Figure 4.49 shows the possible combinations for characters 'a' and 's'. The fourth image for both characters shows the correct version. If the number of UKA regions was larger, then the number of combinations would increase exponentially:

$$\text{Number of combinations} = s^r$$

where  $s$  is the number of states, and is 2, either foreground or background, and  $r$  is the number of regions.

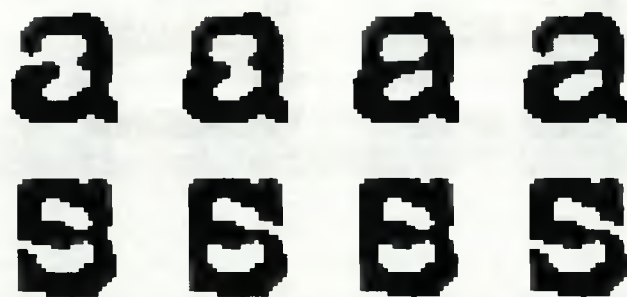


Figure 4.49: Possible UKA combinations for characters 'a' and 's'.



## 4.7 Results and Conclusions

In this chapter a new method for the extraction of highly degraded typewritten characters is given. The novel method strengthens the results of both, adaptive and global thresholding methods, by combining them in each individual character. The output is a ternary image where the third level represents pixels where it is uncertain to whether they belong to the foreground or the background. A set of the pixels in the uncertain category could be identified to belong to the foreground and background and these were turned black and white respectively. The remaining pixels were categorised to be part of a broken or touching stroke and were kept as grey. After the flexible recovery and refinement, the character images benefit at different levels. By extracting the characters individually, the segmentation process is facilitated. The structure of the characters is extracted and problematic areas located and finally refined. After the refinement, the resulting character images show smoother edges and cleaner strokes. The number of uncertain pixels has been reduced to its minimum and it is now used just where the characters are either wrongly connected or broken.

The results of recognising the recovered images by the OCR were all positive. Two different OCRs were used (MODI and ABBYY) and, in both cases, there was an improvement with respect to the original images. In Table 4.1

OCR	Original Image	FTR Image	Improvement
<b>MODI</b>	58.35%	68.68%	10.33%
<b>ABBYY</b>	52.02%	71.39%	19.37%

Table 4.1: Results improvement between the Original image and the FTR image

the improvement achieved by the FTR image is shown. The performance of the original degraded images when directly sent to the OCRs was not very good with an average recognition rate of less than 55%. The improvement experienced after the images were flexibly extracted and refined was of up to 19% for ABBYY's OCR and almost 10% for MODI's. These results were obtained from a sample set of images containing more than 11000 characters. In Chapter 6 these results will be presented in more detail. These results reflect that the FTR method has improved the recognition performance of both OCRs.

Providing the OCR with a grey-level representation of the areas where a de-

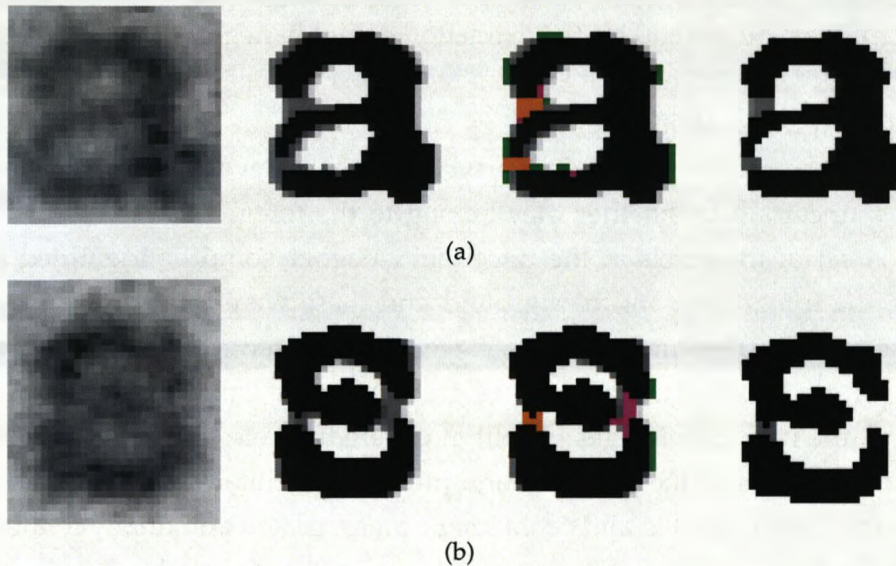


Figure 4.50: Extracted and refined resulting characters

cision could not be made to whether this belonged to the character provides higher recognition rates since it is left to the OCR to make the decision. Following in Figure 4.50, several resulting characters of the combination thresholding and refinement are shown together with their original version.

In Figure 4.50(a), a sample image of character 'a' is shown. Its refined version correctly maintains the key areas in grey level. On the other hand, in Figure 4.50(b), the character 's' is broken on the right hand side. The uncertain pixels in that area of character 's' have been selected as UEA instead of UKA due to their positioning. In such cases the refinement method will not maintain the uncertain area.

# Chapter 5

## Character Recognition

### 5.1 Introduction

In this chapter an OCR system for degraded typewritten characters is proposed. The input of this recognition system are the flexibly recovered characters as proposed in Chapter 4. The recognition method used is mainly based on a *Weighted Image Matching (WIM)* approach.

As explained in Section 3.3.1, template matching can be used when a limited set of fonts needs to be recognised and when it is possible to obtain the character prototypes in advance. Template matching can be made size invariant and rotation invariant, though the already high computation necessarily increases. Typewritten characters are good candidates for template matching since there is only one font type per typewriter and no size variations.

This image matching algorithm was enhanced to be more robust against degradation. And, to improve its performance, a set of features were extracted to reduce the number of matchings necessary. When classifying a set of patterns, in this case degraded characters, the selection of a feature extraction method is probably the single most important factor in achieving high recognition performance. When noise affects the characters, the extraction of useful, trustworthy features becomes even more difficult. The extraction of incorrect features will lead to misclassification, and therefore to recognition errors.

Many feature extraction methods have been studied, however, as these surveys in the literature [22, 57, 12] agree, a feature extraction method that proves to be successful in one application may not be useful in another domain. Different features work in different image types; Some in greylevel images, some

in binary images, some in skeletons and some in contours, which some are more robust than others against degradation.

To extract optimal features, the nature of the written characters must be known, as well as their possible variations and degradations. It is also important to know, whether more than one symbol is used to define the same character, like characters that can be written in several ways, or in the case of degraded characters, whether they are often broken or connected in certain areas. When characters are degraded, as in most historical documents, different features are needed than for non degraded characters.

Features should be *invariant* to the expected distortions, such as degradation, rotation, translation, orientation, scaling, stretching, skewing or mirrored images. If invariant features cannot be found, an alternative is to normalise the input to have a standard size, rotation, contrast, etc.

Performance also depends on the type of classifier used. However, classification results are also not comparable, since different data sets and different features may be used.

Discrete features, that can assume a limited number of values (2-3), are ideal for trees while real valued feature vectors are ideal for statistical classifiers. Graph descriptors or grammar-based descriptors are better suited for structural or syntactic classifiers.

If a statistical classifier is to be used, a large training data set is necessary. Discriminant analysis can help select the features with the highest discriminant power. Multi-classifiers may be used, combining statistical, structural and neural networks to use their differences. For some applications, such as degraded characters, robustness for degradations and variations is important. Skeletons and contours will provide very different features if characters are broken or self touching and different classes will have to be trained for these.

In both feature extraction and classification, experimental evaluation is necessary to decide which features are better for a given data set.

The documents to recognise are assumed to all be typewritten and may have been produced by several typewriters, however, only one typewriter font is expected in each text region. In our data set there are carbon copies and documents with different colour paper. The size of characters for each typewriter is invariant and is within reasonable similarity to characters size of other typewriters of that time period. However, degradation may alter the size of the characters with additive or subtractive noise. The scanning resolution for all



documents is assumed to be 300 dpi.

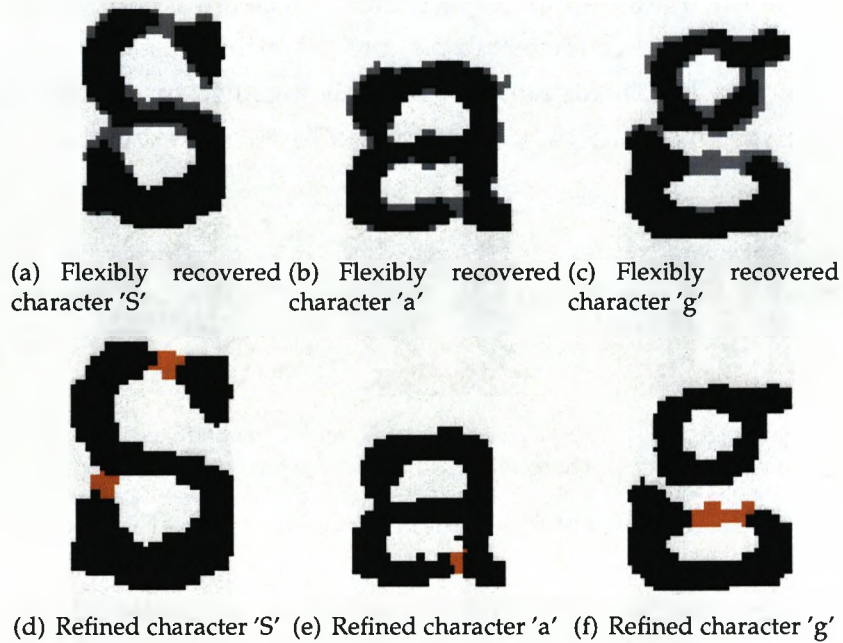


Figure 5.1: Extracted Character Images

The characters to be recognised are the extracted and recovered FTR images resulting from the method proposed in Chapter 4. The resulting characters maintain, in most cases, their shape and most distinguishing features. However, sometimes their quality can be very low and some features may be altered. The images of carbon-copy documents are the images with the lowest quality, since the printed characters are faded and blurred. This results in faded strokes or erroneously touching edges.

Degradation can appear in the characters in a number of ways, (see Figure 5.1). Broken and touching strokes are the most frequent, but as well other distortions in the character edges occur. A number of uncertain grey-level areas have been preserved in some characters where a decision could not be made as to whether they belonged to the character or not. For this reason, the character images are in the majority bi-level and in some cases ternary.

Broken strokes are found to be random and, although they are not restricted to a specific set of characters, tend to appear more in those delicate characters that have thin strokes. Such is the case of 'M' or 'W' where the long strokes are compressed in a small space (relative to their size), thinning them and making

them more vulnerable to fading away. On the other hand, touching strokes are more frequent in characters where thick strokes are concentrated in a small area. Highly affected are, in this respect, the characters free from ascenders or descenders. Thus the characters 'a', 's' or 'e' are the most damaged, but also any character whose serifs are placed closely together, such as the base of characters such as 'm', 'n', 'h' or 'x'. See Figure 5.2 for some examples.

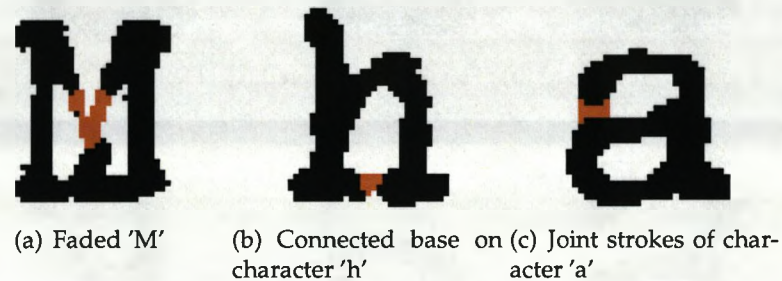


Figure 5.2: Examples of frequently degraded character images

Summarising, the characters to be recognised are to be found in one size only, and are expected to be rotation free. They are typewritten and therefore, the spacing between them is fixed. Each character is only represented with one symbol. However, due to degradation, the creation of different classes for some characters may be necessary. Characters might not always be degraded in the same place or by the same kind of degradation.

In Section 5.2 the Weighted Image Matching (WIM) method chosen will be described followed by its results and conclusions. Further, Section 5.3 and Section 5.4 present the feature extraction methods chosen and their performance together with the Classification methods used. In Section 5.5, a new method for stroke refinement will be described. Section 5.6 will go through the results obtained by these methods. Finally, Section 5.7 describes the conclusions obtained from the performance of the whole recognition system.

## 5.2 Weighted Image Matching (WIM)

Good character samples from several of the old typewriters used to produce a number of the documents here analysed were available to us. These were created recently by historians by using original paper from the period when the documents were created and one of the original typewriters used during

World War II. Sample characters were extracted from these documents, analysed and stored as binary images to be used for later classification and matching, (see Figure 5.3). The method used for the extraction of the characters is the same one as described in Chapter 4.

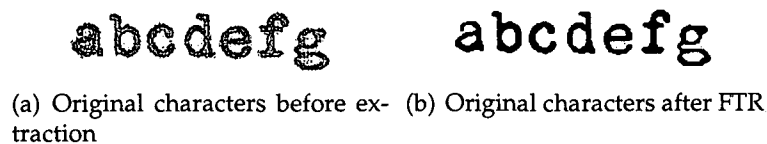


Figure 5.3: Character images used as prototypes

Although they have not been under age related degradation, the resulting characters are not perfect, showing irregular edges as a product of the textured paper. For some of the model character images, manual improvement was necessary to reduce the noise in edges and other degradations.

Having the character prototypes, template matching was the first sensible step towards recognition.

In image matching there is no feature extraction involved and the character image itself is used as a feature vector, avoiding the complicated decision concerning the feature extraction method. Similarity (or dissimilarity) is measured between each template and the character image to be recognised. If the similarity is above a certain threshold, or is greater than the similarity to any other template, then the character is assigned to that class.

Both templates and refined extracted characters, have been localised to their bounding box and are ready to be matched. Template matching usually consists of two steps: superimposing an input shape on a template and measuring the degree of coincidence between the input shape and the template.

To describe the template matching used, each of these two steps will be explained separately.

### 5.2.1 Superimposition

To superimpose two images it is necessary to consider that the characters may have different sizes and therefore a position in the image where they should be overlaid needs to be chosen.

Although typewritten characters are known to be of fixed pitch, this does not refer to the actual character size, this being variable from one character to

another. Degradation can also modify the characters size and different samples of the same character degraded in different ways can all produce different sizes. Smaller characters are particularly subjected to this size alteration since their bounding box, is in most cases, in contact with its outer edges.

There are a number of ways to superimpose two images. Two were tested: finding the centres of both images and through their foreground center of gravity. After comparing both approaches, it was found that using the center of the image was more reliable and robust against degradation. However, one simple calculation of the center of the image was found to be insufficient and the results were still poor. In order to accommodate the size variations caused by degradation, both images were superimposed also at different neighbouring positions from the center. After several tests with different neighbourhood sizes, the best results were obtained by taking a neighbourhood of size one in all directions from the center, making a total of 9 superimpositions, including the center.

At each superimposition, the center of the unknown image is moved around the template image center as explained above. For each of the 9 comparisons performed, a new image is created from the union of both images at the chosen center.

The degree of coincidence will be then measured from each of the union images created at each of the 9 superimpositions. As some of the enhanced images maintain grey-level (uncertain) areas, the matching needs to take it into account some more details described as follows. The pixels in the new image can be a *white-match* or a *black-match* if both pixels are either white or black respectively. They can be a *mismatch*, if one image is white and the other is black, or a *grey-mismatch* if the unknown image is grey and the template is either black or white. The mismatch can be divided into two,  $mismatch_t$  and  $mismatch_u$ , with  $mismatch_t$  being the case when the template is white and the unknown image is black, and vice versa for  $mismatch_u$ .

## 5.2.2 Measuring the degree of coincidence

After the images are superimposed the second step is to measure to what degree they match. This can be done by measuring their similarity or dissimilarity. The sum of the distances between each pair of pixels is calculated and normalised. That is, the distance for a *white-match* or *black-match* will be 0, the



distance for a *mismatch* will be 255, normalised to value 1, and 128 for a *grey-mismatch*, normalised to 0.5. For each pixel in the union image the distance is calculated and added up.

However, this basic approach is not robust enough against degradation. To improve the results, areas of non-matching pixels are measured in size and the larger and further the area, the higher the distance between the character and the template. In this way, thin image edges that do not match due to additive noise or erosion have less weight in the final score than a large non-matching region, for example a non-existent ascender.

We describe the *matching score*  $S$  as:

$$S_I(I_u, I_t) := \sum_{x=0}^{I_{Width}-1} \sum_{y=0}^{I_{Height}-1} S_{\langle I_u, I_t \rangle}(x, y)$$

$$S_{\langle I_u, I_t \rangle}(x, y) := \begin{cases} 0 & \text{if } I_u(x, y) = I_t(x, y) \\ 0.5 & \text{if } I_u(x, y) = 128 \\ n_{\langle I_u, I_t \rangle}(x, y) + d_{\langle I_u, I_t \rangle}(x, y) & \text{if } I_u(x, y) \neq I_t(x, y) \end{cases}$$

$$n_{\langle I_u, I_t \rangle}(x, y) := \sum_{i=x-1}^{x+1} \sum_{j=y-1}^{y+1} S'_{\langle I_u, I_t \rangle}(i, j)$$

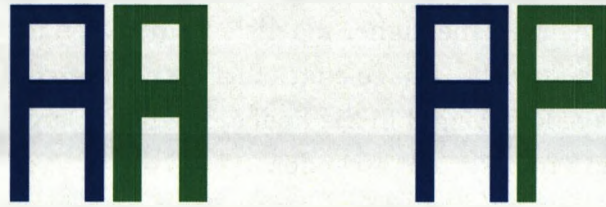
$$S'_{\langle I_u, I_t \rangle}(i, j) := \begin{cases} 0 & \text{if } I_u(i, j) = I_t(i, j) \\ 0.5 & \text{if } I_u(i, j) = 128 \\ 1 & \text{if } I_u(i, j) \neq I_t(i, j) \end{cases}$$

$$d_{\langle I_u, I_t \rangle}(x, y) := \begin{cases} \min_{\substack{x'=0..I_uWidth \\ y'=0..I_uHeight}} \{ED(x, y, x', y') \mid I_u(x', y') = 1\} & \text{if } I_t(x, y) = 0 \\ \min_{\substack{x'=0..I_tWidth \\ y'=0..I_tHeight}} \{ED(x, y, x', y') \mid I_t(x', y') = 1\} & \text{if } I_u(x, y) = 0 \end{cases}$$

$$ED(x, y, x', y') = \sqrt{(x - x')^2 + (y - y')^2}$$

where  $I_u$  is the *unknown* image and  $I_t$  is the *template* image.  $I_{Height}$  and  $I_{Width}$  are the dimensions of the *union* of both images, and  $I_u(x, y)$  and  $I_t(x, y)$

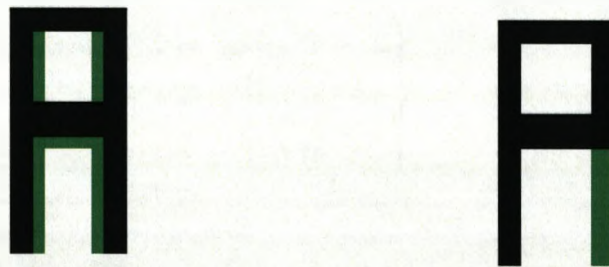
represent the pixel value for each image at coordinates  $x$  and  $y$ . The expression  $n_{(I_u, I_v)}(x, y)$  represents the neighbouring pixels around the evaluated non-matching pixel at coordinate  $(x, y)$ . Each of its neighbours, that is also a non-match, will contribute to an increase in the *score*  $S$  as mentioned above with the normalised distance from the two pixels. The images in Figure 5.4 show



(a) Two samples images of character 'A' (b) Sample images of characters 'A' and 'P'

Figure 5.4: Example character images to be matched

an example of the matching of two characters. In Figures 5.5(a) and 5.5(b) the non-matching pixels are shown in green. The difference of these two images by adding up the non-matching pixels results in a matching score  $S = 36$  for image 5.5(a). However, the score in 5.5(b) is only 20. Simply adding up the difference is not enough since it concludes that 'P' in Figure 5.4(b) is a better match than the thicker 'A' in Figure 5.4(a). An improved method, considering

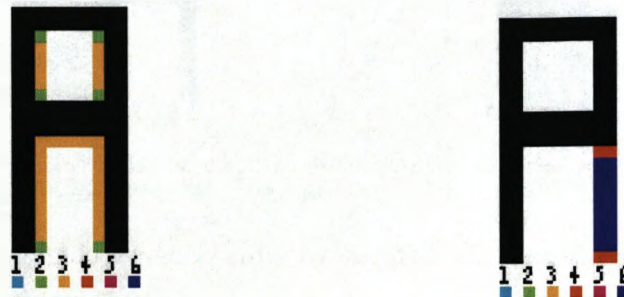


(a) Difference between the two samples of 'A'. Matching cost = 36. (b) Difference between the samples of 'A' and 'P'. Matching cost = 20.

Figure 5.5: Differences with between character images

the size of the neighbours of the evaluated non-matching pixel, (shown in Figures 5.6(a) and 5.6(b)) accounts for the weight of the non-matching pixels in

its neighbourhood with a value for each neighbour equal to their normalised distance, that is 1. In the images 5.6(a) and 5.6(b) the weight of each pixel is described by the colour scheme at the bottom of the image. In this case, the



(a) Neighbours weights. Neighbourhood size = 102. (b) Neighbours weights. Neighbourhood size = 112.

Figure 5.6: Differences with neighbours weights between character images

matching distance from the two 'A' characters in image 5.4(a) now is equal to  $6 \times 2$  plus  $30 \times 3$  making a total score of 102. On the other hand, the cost of the characters 'A' and 'P' shown in Figure 5.4(b) is now  $4 \times 4$  plus  $16 \times 6$ , a total cost of 112, making the 'A' a better match than the 'P'. To improve the measurement further, a distance measurement is taken to increase the cost of non-matching pixels that are further from the real character strokes. In this case, the different mismatch possibilities are taken into account. If the mismatch is  $mismatch_t$ , the distance  $d$  from the  $mismatch_t$  pixel (the template is white and the unknown image is black) to the closest black pixel in the unknown image is added to the previous score. In the same way, if the mismatching pixel is  $mismatch_u$ , the distance will be calculated to the closest black pixel in the template. Thus, non-matching pixels that are further from the body of the template will increase the cost more than pixels along the body. This is illustrated in Figures 5.7(a) and 5.7(b). The distance cost for the two images of character 'A' is  $36 \times 1$  and the distance between characters 'A' and 'P' is  $2 \times 1$  plus  $2 \times 2$  plus  $2 \times 3$  plus  $2 \times 4$  plus  $2 \times 5$  plus  $10 \times 6$  making a total of 90. Adding the costs up, the difference, the number of neighbours, and the distance of each mismatching pixel the Matching Cost for the two 'A's is now 138 and the Matching Cost for 'A' and 'P' is 202.

The resulting matching costs for these images are now well separated and proved robust in our experiments. To proceed with the recognition, this same



matching is attempted with all the characters in the alphabet. The Best Match,

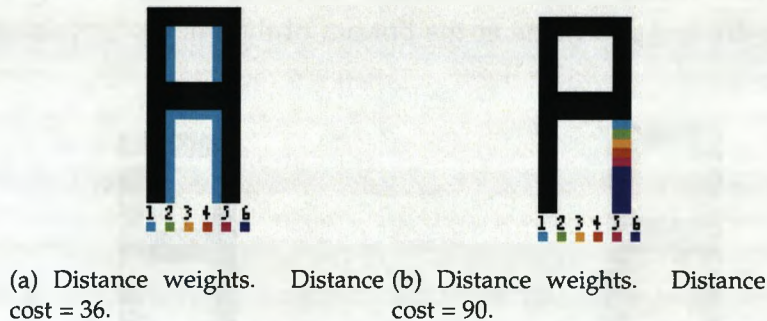


Figure 5.7: Differences with distance weights between character images

$BM$ , for each of the characters in the document can be described as:

$$BM(I_u) := t \text{ where } S_I(I_u, I_t) \leq S_I(I_u, I_{t'}) \text{ for all } t' \in \{a, b, c, \dots\} \text{ with } t \neq t'$$

The total number of matches for each character in the image is quite large. For each character, 9 union images are created and each of them is matched to all the characters in each font. Assuming each font has 26 lower case characters, 26 capitals, 10 numbers and around 7 punctuation symbols, the total number of comparisons per character is  $9 \times (26 + 26 + 10 + 7) = 621$ . If an average page can contain 1500 characters, for each page, the number of matchings can be almost one million.

The matching method described here was aimed at typewritten characters. It emphasizes the size of the non-matching areas and their distance to the closest stroke in order to increase the matching distance. The character images are well recognised using this method. However, as is expected, the main disadvantage of such matching is that it requires high computation and a reduced set of characters.

Computation can be lowered by reducing the number of templates that it is necessary to compare the image to. This could be done by classifying the templates into a smaller set of classes and attempting to match only the characters that belong to the same class. This would require the extraction of a set of robust features with good discriminating power.

New prototypes would be needed if the given document is produced by a typewriter with a different font. An option could be to ask the user to select the font type before proceeding with the recognition.



When extracting the set of character prototypes for this project, two different fonts were found in our dataset. The difference between the two fonts was not very evident in most characters. However, for some, it was very significant, varying in the distance among strokes, the curvature or even the width and height. Two examples of such differences are shown in Figures 5.8 and 5.9.

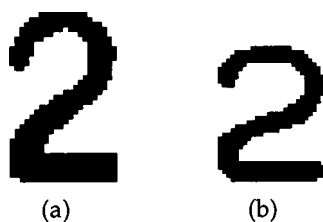


Figure 5.8: The character '2' from two different typewriter fonts.

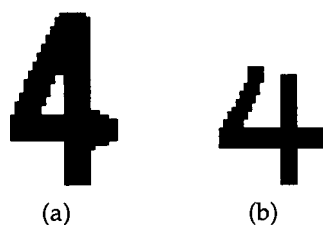


Figure 5.9: The character '4' from two different typewriter fonts.

Most errors in WIM are produced in highly degraded characters, or among characters with physical similarity. However, it was noticed that the correct character is usually among the best two matching characters.

### 5.2.3 Recognition Confidence

As a way to measure the recognition confidence of the best matching characters and identify possible recognition errors, a study of the matching results was carried out. To do this, the resulting best matching scores for a given character, against all the templates, are plotted in ascending order. When the character has been correctly identified, the cost is generally lower and a difference in the steepness at the beginning of the plot can be identified.

In Figure 5.10 and Figure 5.11 the plots of three well matched characters and three mis-matched characters are shown respectively. Each dot in the graph represents the cost to match a given unknown character to each template. The

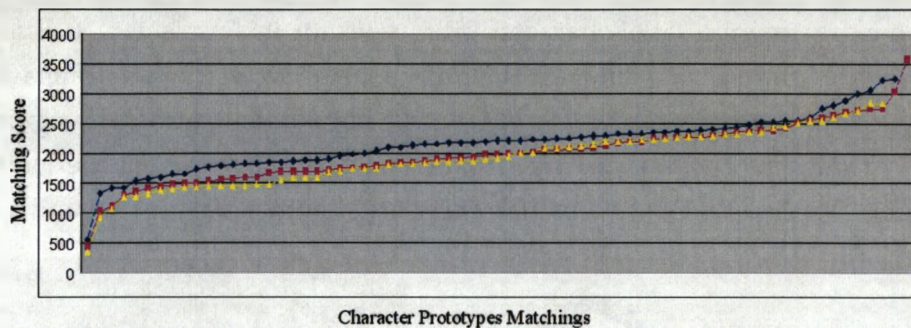


Figure 5.10: Plot of the WIM score for three random well matched characters.

three sample characters are compared to all the characters in the alphabet and the scores are shown in ascending order by cost; the best matching character has the lowest cost.

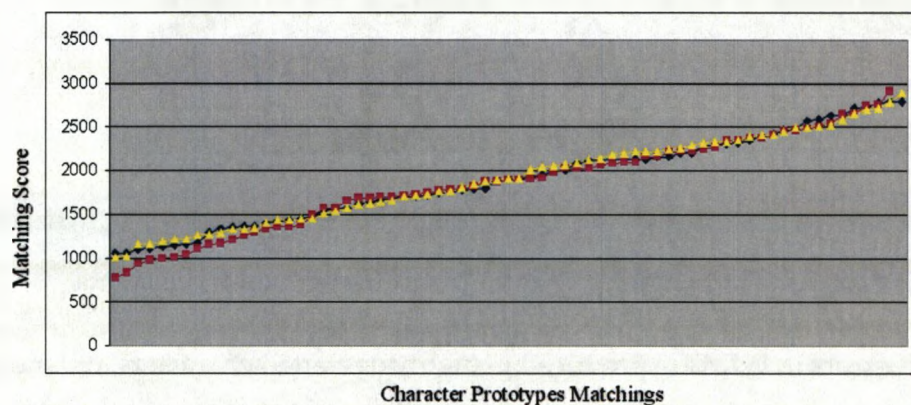


Figure 5.11: Plot of the recognition rates for three random mis-matched characters.

In the plot in Figure 5.11, there is no steepness at the beginning and increases at the same pace as the other characters. A distinctive feature observed in the plots is that the matching score of the best character, when this is mis-recognised, starts at a higher value matching score than when it is correctly recognised. In Figure 5.10, the best recognised characters have a cost around the value 500 or lower and the misrecognised ones in Figure 5.11 start from around cost 800 to 1000.

Based on the recognition cost of the best matched characters that were correctly and incorrectly recognised, a threshold was searched for to determine

whether a given matched character should be given higher or lower recognition confidence. This threshold was tested with values between 500 to 800 for the well recognised characters, however, the matching rates vary largely from one character to another and is influenced by the character size and degree of degradation. The same character can also produce very different matching scores depending on how degraded its shape is. For these reasons, a robust threshold could not be found and another feature was studied.

As mentioned earlier, it was noticed that when the best matching character was correctly recognised, the distance to the matching cost of the next best character was noticeably larger than when it was incorrectly recognised.

Since in most cases, the confidence in the matching is in doubt only between the first and second best matching characters, by measuring this distance between them we could measure the confidence. We are confident that the first character is a good match if the distance to the next best matching template is higher than the average distance among the other character matching scores. This larger distance to the next character is seen as a steeper section at the beginning of the plot.

As seen in Figure 5.10, for these three characters the distance from the first to the second best character matching score is considerably larger than the distances between the other values in the plot. However, when a character is misrecognised, the distance between the matching score for the best character and the next one is quite similar to the distance to the next values.

When the cost of the best two matched characters is relatively close, then the confidence on the recognition of the best matching character is low. In this case, the second matching character is also suggested as a possible candidate.

The larger the distance, the higher the confidence, and vice versa. Based on these facts, a measure of the recognition confidence was implemented.

To do this, the distance between each pair of matching costs is plotted in ascending order, see Figures 5.12 and 5.13 for the well recognised and misrecognised characters respectively. These plots display only the distance of the first ten values for a closer look.

If the distance between the first best match and the score of the second best is within a 10% of the score of the first character then it is considered too close and the confidence is lowered. This takes into account both the distance between the two best characters and the actual score of the chosen one. In this way if the match is very low then the distance allowed for a high confidence is



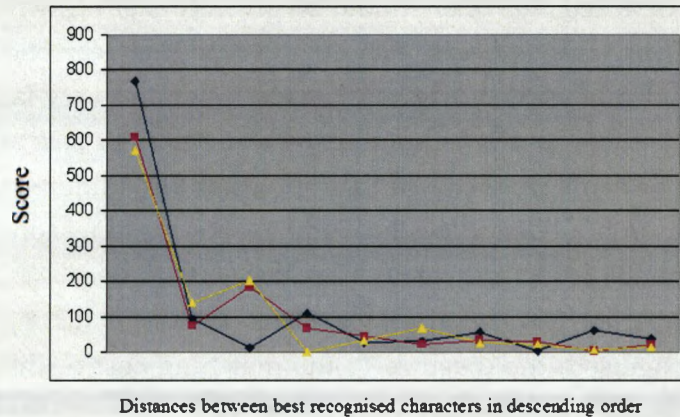


Figure 5.12: Plot of the recognition distances between three well matched characters.

smaller than if the matching cost is high.

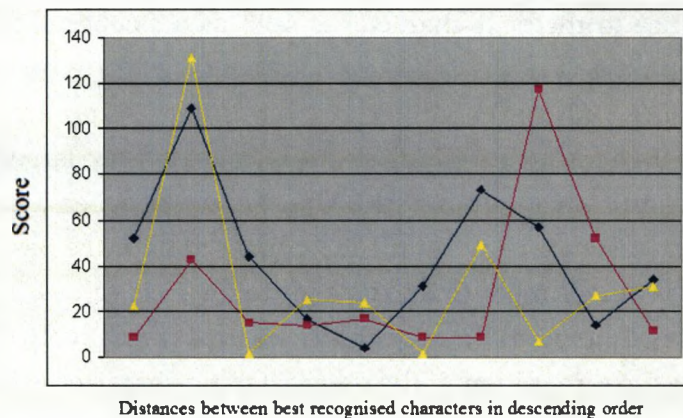


Figure 5.13: Plot of the recognition distances between three mis-matched characters.

When the recognition confidence of the best matched character is low the second best character should also be considered as well as a possible candidate for recognition.

A sample image displaying the recognition confidence is shown in Figure 5.14. These two images show the original image and its matching result. In this second image, the characters in blue are recognised with high confidence. The characters in red are recognised with low confidence. The results gathered from performing tests using the confidence measurement are shown

**Weliuonischkis**  
**Bondarenko**  
**Paschkewitschius**  
**Geguschis**

(a) FTR sample image

**Weliuonischkis**  
**Bondarenko**  
**Paschkewitschiue**  
**Oeguschis**

(b) Recognised image

Figure 5.14: Recognised image with confidence indicators. The blue colour indicates above threshold confidence, the red colour indicates below threshold confidence

in Table 5.1. In average, 70% of the correctly recognised characters had been correctly flagged as confident. The remaining 30% were incorrectly labelled as confident but were misrecognised. This is displayed as “recognised in Blue” and “misrecognised in Blue” in the mentioned table.

Among the misrecognised characters, almost 70% had been flagged with low confidence and an average of 30% of the misrecognised characters had been flagged as high confidence. These are displayed in the table as “misrecognised in Red” and “recognised in Red”, respectively.

Recognised in Blue	Recognised in Red	Misrecognised in Blue	Misrecognised in Red
70.27%	28.24%	31.77%	69.19%

Table 5.1: Recognition results obtained based on the confidence measure.

It has been measured that 89% of the times when a character was misrecognised, the second best recognised character turned out to be the correct character. This could be of much help for an algorithm that, when the recognition confidence was measured as low, the second best recognised character could be suggested to the user as a possible character. Another use would be creating a set of discriminating rules to choose among the pair of best matching characters when the recognition confidence is low.

In any of these cases, if the correct character was chosen among the best two, the recognition could improve. This can be expressed as that 89% of the

69.19% of misrecognised characters that were flagged as low confidence could be corrected improving the recognition even further.

That is, if the recognition rate of a given image was 90%, then, by using the recognition confidence calculation here proposed, the recognition could be improved by 6.3% raising it to 96.3%.

### 5.3 Feature Extraction

The main weakness of template matching is the high computational cost as each unknown character must be compared to all possible templates, lower and upper case characters, capitals, numbers, and punctuation symbols. To aid reducing the computation of template matching, the number of necessary matches can be reduced by classifying them into a smaller number of classes.

A number of features could be extracted from the prototype images. However, the features needed to classify the prototypes should also be reliable and robust on the degraded characters. As explained at the beginning of the chapter, finding robust features on degraded characters is a very complicated task. They must work on both grey-level and binary images and must be robust against broken and touching strokes, variable stroke thickness and irregular edges.

Robust features for highly degraded characters are difficult to find and recognition rates are usually much lower than in good quality images.

Features can be extracted from the different representations of an image such as grey-scale, binary, skeleton or contour representations. Each of these requires different extraction methods and the methods are not always reliable.

Skeletonisation and contour extraction are very sensitive to degradation, where irregular edges and additive noise produce unreliable results with erroneous branches and dents. Discrete features that can be extracted from the skeleton representation of the character, such as the *number of loops*, *T-joints*, *X-joints*, or the *number of end points* are therefore not reliable on the classification of degraded characters. These topological features are easily altered by degradation, producing a large number of classes for each character. As well as being very sensitive to noise, these methods can be used in binary images only, not taking into account the uncertain grey areas.



### 5.3.1 Character Size

The *character size*, on the other hand, was found to be a strong feature with excellent discriminating power and being very easy to extract. The size is extracted from the bounding box used to localise the characters. Both prototypes and degraded characters share the same size which is very robust against degradation. Three classes can be easily differentiated. The first contains the large, *tall* characters, including all capitals, numerals and letters with ascenders and descenders. The second class contains the small, *short* characters, lower case characters that do not have ascenders or descenders, while the third class contains the *punctuation* symbols, such as commas, dots, colons and semi-colons that are narrower or much shorter than the rest of the characters. The

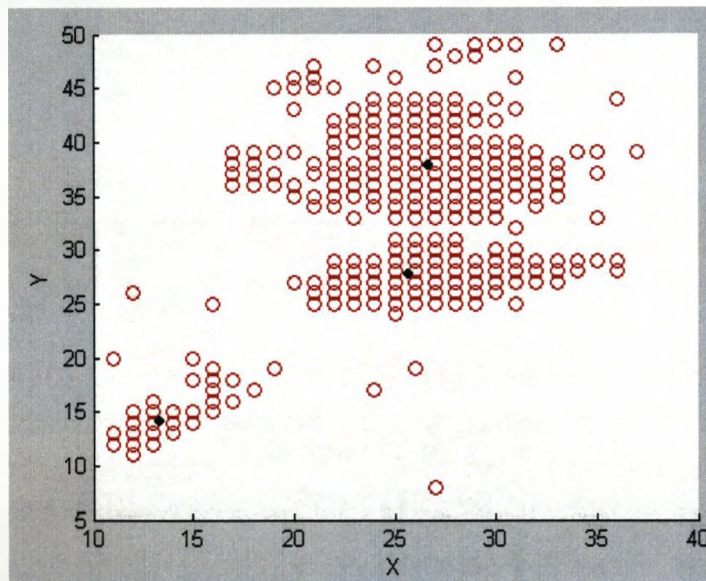


Figure 5.15: Character size clusters

plot in Figure 5.15 shows the sizes of almost 1000 characters where the *X axis* represents the characters *width*, and the *Y axis* the *height*, in pixels. To separate the three different size classes, a fuzzy c-means clustering algorithm was used. Each point in the plot has a degree of belonging to clusters, as in fuzzy logic, rather than belonging completely to just one cluster.

Using the *find-cluster* function in MATLAB, three clusters are identified, centred in each of the classes. The cluster at the top represents the *tall* characters. The middle one represents the *short* characters and the cluster on the bottom left corner corresponds to the *punctuation* symbols.

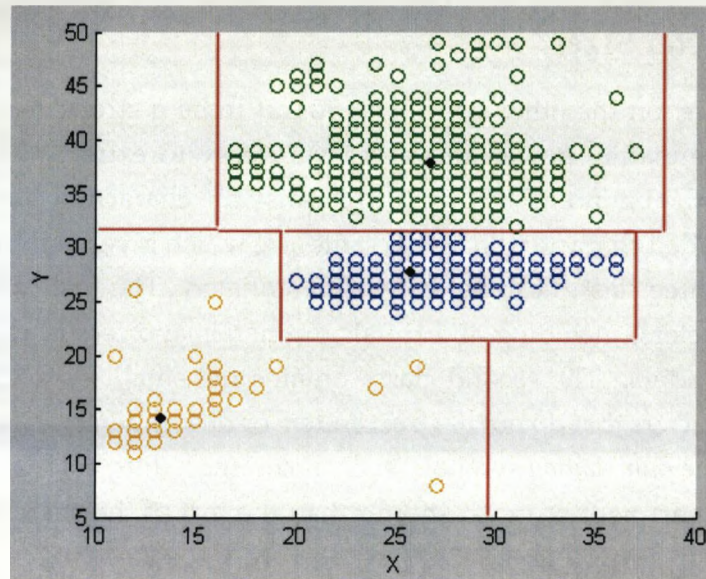


Figure 5.16: Character size clusters classified

The plot in figure 5.16 highlights the three different size classes created and their delimiters. After classifying the alphabet set into these three size classes the number of matchings is greatly reduced.

Jonas  
 Vytautas  
 Pranas  
 Leonas  
 Piotr  
 Wiktor

Figure 5.17: Characters classified by size.

The image in Figure 5.17 shows the localised characters being classified by their size by using the classified regions as described above. Tall characters can be divided further into two: characters with ascenders and capitals, and characters with descenders or *low*. To determine if a character is *low* the baseline of the rest of the characters is studied. Characters with descenders fall below



the baseline of the other characters in the text line. As can be appreciated in the image in Figure 5.17, the baseline of all the *short* characters, or characters without descenders, is not very regular and can move a couple of pixels either up or down.

Two methods were implemented to calculate whether a *tall* character was in a *low* position in the line. The first method would take an average of the base of all *short* characters in the same text line. The position of the given tall character would be compared to the obtained average and if it is lower than a certain threshold, usually 3 pixels, then it would be classified as a *low* character. The position of the baselines of the short characters were taken locally to their lower text line segmentation divider.

This method performed well in images without any skew. However in the presence of skew, in long text lines the local position of the characters at the beginning of the line is significantly different to the ones at the end, so corrupting the average.

A better method was found, that compares the base of each *tall* character to the base of its preceding and following characters, so the skew would be minimal. If the neighbour character being compared to is *short* then the top of both characters should be in a similar level, within 3 pixels difference, and the bottom of the lower one should be at least 4 pixels lower. When comparing to a *tall* character then both the top, and base, of the analysed character should be at least 4 pixels lower then the tall one for it to be classified as a low character.



Figure 5.18: Low characters classified (in orange)

In the image in Figure 5.18 all the characters have been classified. The characters in a green box are have been classified as *tall*, the blue ones denote the *short* characters and the orange boxes bound the *low* characters.

At this point, the number of characters has been reduced greatly for each of the classes. The *short* character class is composed of 13 characters and the total number of matchings is now  $9 \times 13 = 117$ . The *tall* characters class is still quite large with 34 characters, but the number of comparisons is reduced greatly

to  $9 \times 34 = 306$  matches. *Low* characters are the smallest group with only 5 characters in this class. The matching in this case will consist of  $9 \times 5 = 45$  matches.

Punctuation symbols, however, are smaller symbols that have their own class. Dots (.), commas (,), colons (:) and semi-colons (;), star (\*), hyphen (-), quotes (") and apostrophe (') are particularly thin and short and can be easily differentiated from short and tall characters as seen in Figure 5.16. Characters classified in this small punctuation class need only  $9 \times 8 = 72$  matchings for the recognition.

The number of comparisons for each unknown character should be reduced further and more reliable features need to be found.

### 5.3.2 Character Weight

Tall and low characters compose the largest class with 34 characters together. Most tall and low characters can be divided by the position in the image of their vertical largest/heaviest stroke. This can be found by projecting the character vertically, to obtain a horizontal projection profile. Projection histograms of the characters have been found to provide useful and robust data from degraded characters.

This method can be applied to both grey-scale and binary images and has proven to be very robust in noisy and touching or broken characters. The values of the histogram are inverted so that the areas where more black pixels accumulate are seen as maxima in the histogram. The maxima are extracted from the histogram by initially smoothing it to reduce the number of minor (local) maxima. The smoothing is performed by averaging with the pixels of its neighbourhood. Different neighbourhood sizes were tested and finally, the average was calculated by taking 2 neighbours at each side of the pixel. The global maxima are identified by first examining the inflexion points where the change is from ascending to descending slope. These maxima are filtered in two ways. One is by discarding maxima that are too close to each other, keeping the highest one. The second filter eliminates the maxima under a given threshold (this threshold value is measured from the maximum local minima at each side).

The way the *weight* is calculated is by first dividing each character's width into three sections. The left and right sections comprise 40% of the localised

character image width each and the center 20%. The *weight* is then classified by finding the highest maximum from the horizontal profile and measuring in what section this falls. The *weight* of the *short* characters was found not to be reliable as their strokes are condensed in a smaller area and the extraction of this feature was not robust against degradation. The same is the case for the *punctuation* symbols where this feature is neither robust enough nor discriminating enough. Based on this, *tall* and *low* characters can be divided into

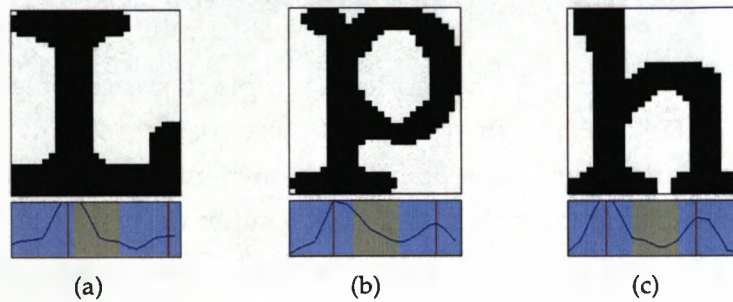


Figure 5.19: Characters in the *left-weight* class

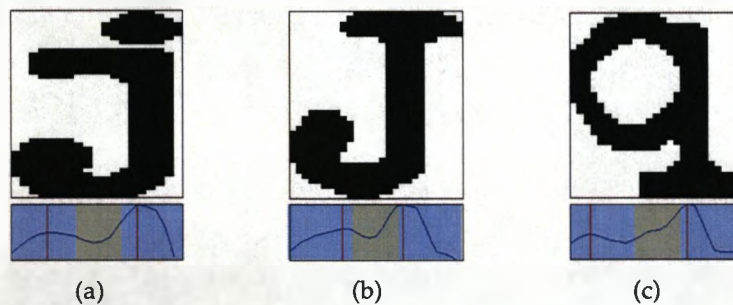


Figure 5.20: Characters in the *right-weight* class

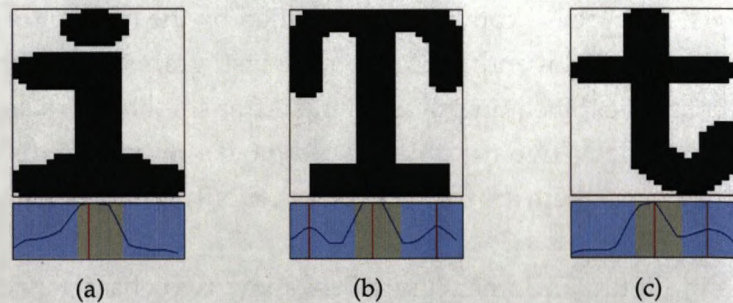


Figure 5.21: Characters in the *center-weight* class



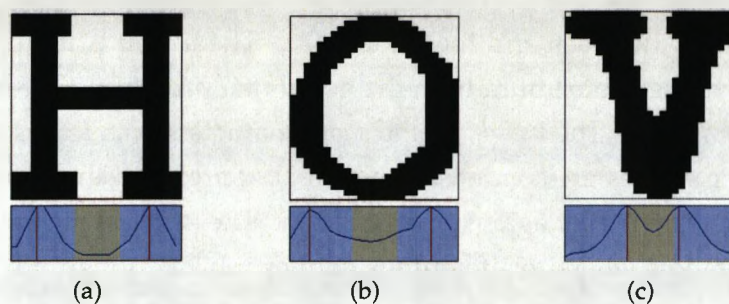


Figure 5.22: Characters in the *double-weight* class

*right*, *left* and *centered* and *double* categories. Example characters of left-weight are L, p, or h, in Figure 5.19. The profiles of these characters show the three different sections and the maxima points are drawn by red lines. In the case of the left-weighted characters their largest maximum falls in the left section. Some examples of right-weight are q, j or J, in Figure 5.20, and some examples of centered-weight are T, t or I in Figure 5.21.

The characters whose best two maxima have similar values and fall in different sections cannot be described as either right, left or center and will be classified as *double-weight*. That is the case of H, O or V, shown in Figure 5.22.

### 5.3.3 Vertical and Horizontal Maxima

Another set of features can be extracted from the character's projections. As used previously for the extraction of the *weight*, a vertical and a horizontal projection can be taken from the characters by accumulating its pixels in a horizontal and vertical manner. From these projections, the number of maxima can be obtained and used for classification.

As for the weight, the profile histogram is smoothed and all maxima points are calculated. The characters can be then classified by the number of maxima they have in their horizontal and vertical profile histograms. A threshold was created to select the most meaningful maxima. After smoothing, a smaller set of maxima are identified. Two maxima should have a minimum distance between them and if two of them are too close together only the highest maximum would be kept.

The images in Figure 5.23 and Figure 5.24 show two characters and their respective profiles. Certain of these characters are degraded, and broken in different areas, but the extraction of the maxima is shown to be reliable. The

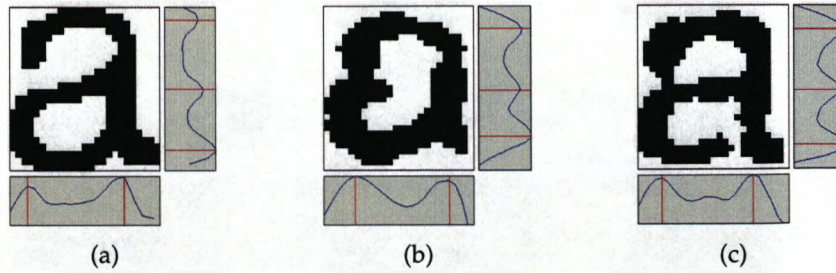


Figure 5.23: Character 'a' projections maxima with different degradations.

maxima chosen are represented by the red lines.

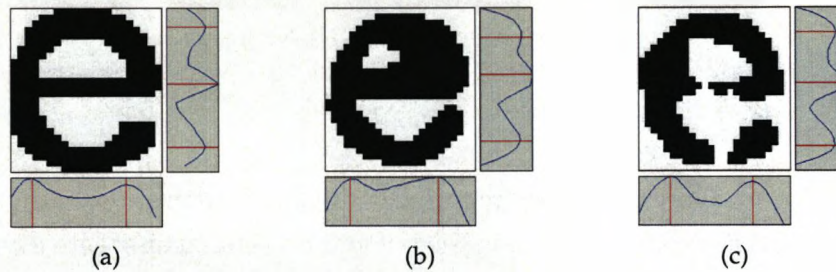
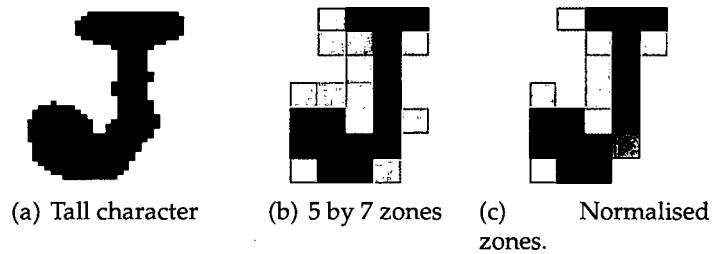
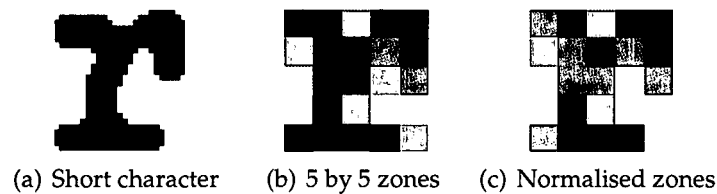


Figure 5.24: Character 'e' projections maxima with different degradations.

The extraction of these discrete features is fast, straightforward and reliable in most cases. If the characters are too damaged and their main strokes or cavities are broken or completely filled in then the profiles might be altered. The use of these features occurs under no, or very small, skew, and on non-italic typewriters. The range of the number of maxima in either projection is from 1 to 4.

#### 5.3.4 Other Features

A number of other features were also considered and implemented, although, these were not as robust against degradation. While not used, we should mention these approaches. They include zoning, the number of connected components, the aspect ratio or the number of inner holes or loops. Zoning consists on computing the average value of the pixels in different zones of the image and using them as features. The zones are defined by an  $m \times n$  grid where  $m$  and  $n$  are the size of each zone in pixels. The output is a feature vector of  $n \times m$  features. Different sizes are needed to better describe different types of

Figure 5.25: Extracted zones of a *tall* character.Figure 5.26: Extracted zones of a *small* character.

characters. Tall and low characters required more zones than short or punctuation symbols. After experimentation the tall and low characters were divided in  $5 \times 7$  zones and the small characters in  $5 \times 5$  zones. An illustration of these is given in Figures 5.25 and 5.26.

To be able to use these features better, the zones have been normalised to a range of 5 values. These ranges are calculated by dividing the greyscale range by 5. Each zone is assigned a class if its value is within its range. The normalised zones of the previous characters are shown in Figures 5.25(c) and 5.26(c). The number of values was reduced to 5 as this was found to provide a good description of the character's zones at the same time as keeping the features in a small range.

The extraction of zone features can be used in both binary and greyscale images. It has also been tried by Bokser in [12] in the character contours and skeletons. However, she reported that neither of these features were found reliable.

An interesting application of the zone features of the contours of an image, is to find the orientation of the segments within neighbouring contour pixels in that zone, and then to classify them into  $0^\circ$ ,  $45^\circ$ ,  $90^\circ$  and  $135^\circ$ . The number of line segments in each orientation was counted. This can help estimate the orientation of the stroke in that zone, or its curvature and so may be used as a feature. However, this feature was found highly sensitive to noisy edges. The



main weakness of zoning was observed when a character edge is close to the zone borders. Small variations in the contour, or different widths for the same character, could lead to large variations in the output features. When degradation affects the outer edge of the characters, their size may be modified. This would affect the zone's intensity by displacing the character to one side. An example of different zoning values for characters of the same class is shown in Figure 5.27.

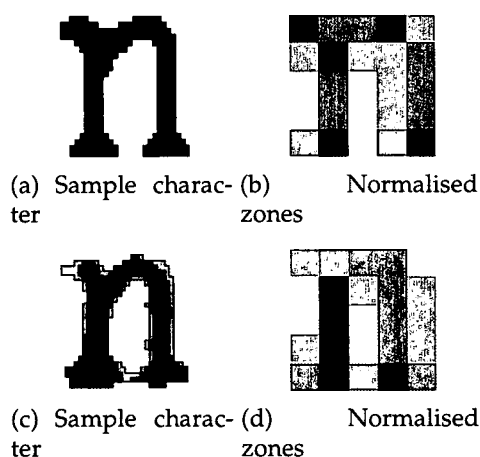


Figure 5.27: Two characters with different zones.

This inconsistency was found repeatedly in different character images. For this reason, zoning was discarded as a robust feature.

The possible use of the *number of connected components (CC)* was also studied. However, it was considered more as a measure of the degree of degradation of the characters than as a feature. Most character prototypes are made out of only one connected component. Only characters *i* and *j* are made out of two CCs, and in most degraded samples the dot has been linked to the body. However, a degraded character might be broken into several CCs. By measuring the number of CCs and their size we could estimate how degraded or broken a character is and pass this information to the OCR or classifier. This method named the *Broken Character Factor* was used by Cannon *et al.* in [14] as a measure of the quality of a document for subsequent restoration techniques. Such information could be of use to the recognition stage to indicate that the feature extraction might fail in such a broken character. However, this idea was not further developed but will be considered in the future.

As mentioned earlier in this chapter, the features extracted from the skeleton representation of degraded character images were discarded for this work due to their sensitivity to noise. There are three main reasons for their exclusion. First, that the extraction of a useful skeleton representation from degraded characters with irregular edges and broken strokes is very complex. Second, the features extracted from the skeleton representation could not be used to robustly classify the characters, since randomly touching or broken strokes would produce too many classes. And, third, they can not be used in the character images containing uncertain areas.

The *number of loops* in the characters was also discarded as a useful feature as it is easily affected by degradation. For the same reasons as explained above, the number of loops would not be consistent in highly degraded characters. An common example would be a character such as 'a' or 's' where samples have been found ranging from no loops to 2 loops in both cases. In addition, the characters that contain uncertain areas that are either linking a broken stroke or erroneously joining two strokes could not be considered.

Finally, the *stroke orientation* was studied. In theory, most character strokes are straight (except a, s, c, o, g, C, G, O, S, 2, 3, 5, 6, 8, 9, 0). By calculating the longest runs in most directions, and comparing the number of this to the character's dimensions, one could classify the characters by the angle and length of their longest and most repeated runs. However, this feature could not be reliably used on images with uncertain areas. Broken or touching strokes, would again alter the feature values and therefore the classification results.

## 5.4 Classification

In this project, the use of the features extracted only aims to aid the WIM by reducing the number of matches and speed up the process.

The need for a classification method is to narrow down the possible matches and, therefore, the computation required. The use of a tree-based classifier was found to be the simplest approach. Features are extracted from the template characters and a classification tree is created from them. Then, for each input character, these same features are extracted and a path in the tree is followed until a distinct class is reached. Then, the input character is matched against the templates in that class, and the best match is chosen.

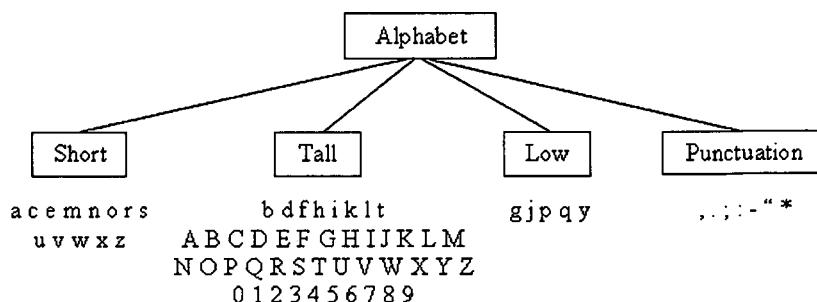


Figure 5.28: Tree classified by size into *Short*, *Tall*, *Low* and *Punctuation*

The first level of the tree is created by the classification of the characters by *size*. Four classes are created, for *short*, *tall*, *low* and *punctuation* characters. The tree at this level can be seen in Figure 5.28

The *tall* and the *low* classes are then classified by weight, as explained in Section 5.3, as *left*, *center*, *right* or *double* weight. *Short* and *punctuation* characters are not suitable for this feature as it is not a reliable measure for them. They will not be classified further at this level.

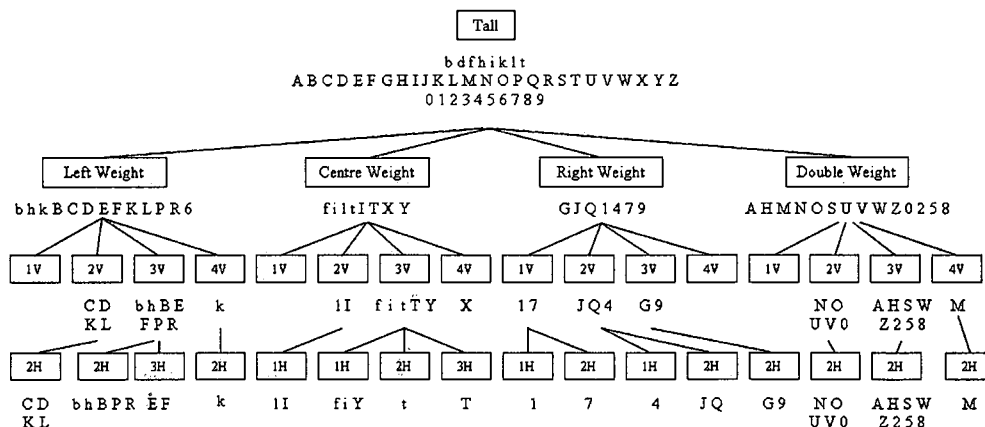


Figure 5.29: Tall characters classification tree

The next feature in the tree is based on the number of maxima from the character projections. The maxima extracted from the vertical projection profile was found to provide more discrimination than the one from the horizontal profile and is used first.

The number of *vertical maxima* ranges from 1 to 4 and this measure can be

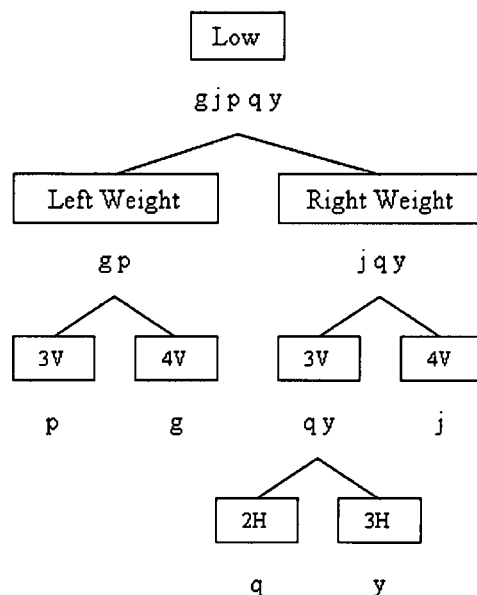


Figure 5.30: Low characters classification tree

applied in all the previous classes created.

Finally, the number of maxima in the horizontal projection-profile is used in the classification tree. At this stage, the number of characters per class is largely reduced and the matching can take place. Note that the vertical number of maxima refers to the maxima extracted from the vertical profile.

The classification tree for the *tall*, *short*, *low* and *punctuation* classes obtained by the classification by size, as shown in Figure 5.28, are shown in Figures 5.29, 5.30, 5.31 and 5.32 respectively. In the graphs, the different classes created by the vertical and horizontal profile features are denoted by  $nV$  and  $nH$  respectively, 'n' being the number of maxima for that class.

Although the chosen features are quite robust, their extraction in highly degraded characters can fail. Such a tree has the disadvantage that an unexpected feature value will lead to a recognition error since the group of templates to be matched will not contain the correct character template. As discussed in Section 5.2, the measure used for the recognition confidence can also be used here. However, in this case, when the first character is not well recognised, the second best candidate might also not be the best character either and so the error is in the classification, meaning that the character we are looking for is not in the selected class. In such a case the character should ideally be sent to be

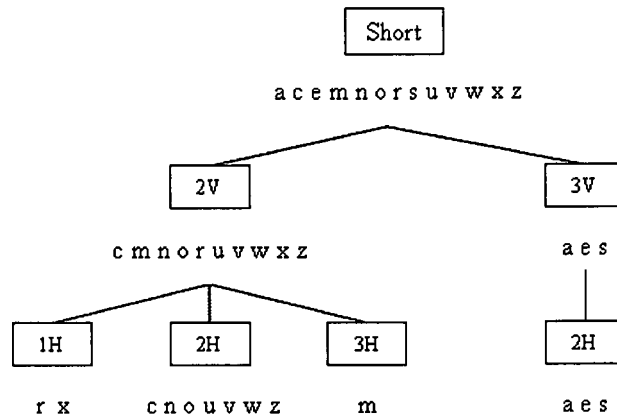


Figure 5.31: Short characters classification tree

matched again, this time against the whole set of characters or to the parent node in the tree, repeating the process.

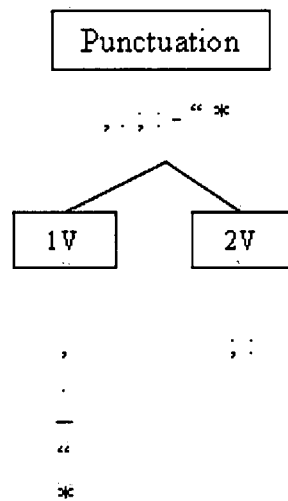


Figure 5.32: Punctuation characters classification tree

To account for more robustness against random degradation and to gain flexibility in the recognition of degraded characters, a statistical classifier was implemented. At least 50 samples of each character were taken to train the Bayesian classifier and the chosen features were extracted from them.

A contingency table was implemented from the training samples computing the probability for each character for a given feature value where each feature

was considered independently from each other. The 10 characters with the highest probabilities would be sent for matching.

Unfortunately, the results of the statistical classifier were not reliable in the cases where a character was degraded in a way where the extracted feature had little representation in the training data. In such cases, the probability of the class of the correct character could be lower than other characters and it would not be picked up for the matching. It is difficult to collect training samples that cover all the possible degradations. The statistical classifier was not found to be of use when using a small number of features. Instead, the training set could

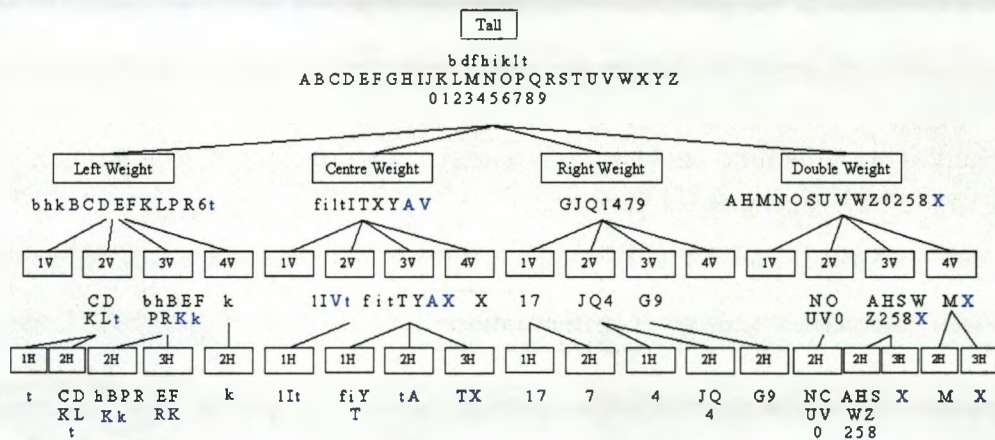


Figure 5.33: Tall characters classification tree created from dataset

be used to create a new probability tree. Thus, each character in the training set, whose probability for a given feature is greater than 0, will be added to the corresponding tree class. In this way, the number of characters per class increases, since each character will belong to as many classes as the range of values taken by its features in the training data. Multiple classes can contain the same character. This means that the number of matchings increases with respect to the previous fixed tree but, also, it allows more flexibility against uncommon degraded characters.

The new trees created from the training data are shown in Figures 5.33 and 5.34. The characters added from the classifier data are shown in blue. The tree for the classification of the *punctuation* symbols is not shown again as it does not change.



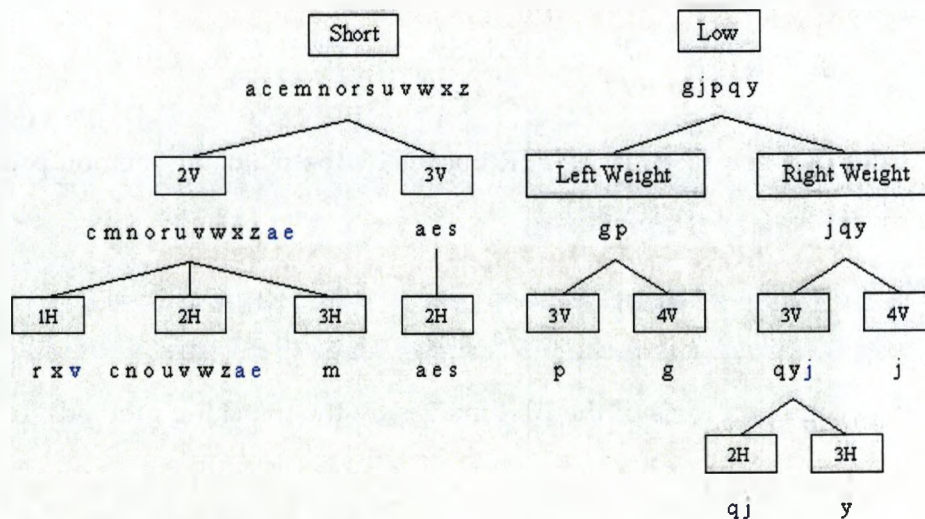


Figure 5.34: Short characters classification tree created from dataset

### 5.4.1 Results and discussion

The results of WIM have improved with respect to the results obtained by blindly matching each character to all the given templates. However, there is a trade in the *time vs. recognition rate* as we step down in the classification tree. The more features used in the classification, and thus, the more levels descended in the tree, the less comparisons were needed to be performed, however, the higher the chances that a character is misclassified, and therefore a reduction in the recognition rate.

The best results obtained from the classification, where only the *size (tall, short, low and punctuation)*, the *weight (left, centre, right and double)* and the *number of vertical maxima (#V)* were used. The last level of the tree, created by the *number of horizontal maxima (#H)*, lowered the recognition and was not used to obtain the final results. The average results of matching using the classification system were, in average, of 89.78%.

In Tables 5.2 and 5.3, a comparison of the results of the original images being recognised by the OCRs ABBYY and MODI and the results of the recovered images proposed in Chapter 4 are shown. The improvements achieved by WIM reach up to 37% improvement in average when compared to the results obtained from the original images, and between 18% and 21% better than the results obtained by the recovered version of the image.

The results given here are the average performance in all the images in the

FTR WIM	OCR	Original Image	Improvement
89.78%	<b>MODI</b>	58.35%	31.43%
	<b>ABBY</b>	52.02%	37.76%

Table 5.2: Recognition rates of the FTR images by the matching method proposed.

FTR WIM	OCR	FTR Image	Improvement
89.78%	<b>MODI</b>	68.68%	21.10%
	<b>ABBY</b>	71.39%	18.39%

Table 5.3: Recognition rates of the FTR images by the matching method proposed.

dataset regardless of the quality. In a group of images where the quality was slightly better than the average the matching results reached up to 94%. In Chapter 6, more results will be given by image type and quality level.

## 5.5 Use of Uncertain Strokes

In this section, a new method is introduced to restore the character strokes. The extracted and refined character images (FTR) may still contain uncertain key areas in grey. These grey-level regions have been kept in the image to maintain information that could aid the recognition process. Before recognition, a decision could not be made concerning whether those grey pixels were part of the character's stroke or just noise.

The new method restores the character strokes by selectively turning the uncertain regions that truly belong to the characters into black and discarding the rest. To identify the uncertain regions that will become part of the character, the method employs the WIM method as described in Section 5.2. This restored images will be referred to as R-FTR. In the following, two alternative methods are described. The first method works as follows. Given a character image, all possible combinations of uncertain regions are computed by either turning them into black or removing them by turning them into white. Notice that the regions turned into black become part of the character. In this way, a new character image is created for each of the possible combinations. The number of possible combinations is equal to  $2^n$ , where  $n$  is the number of uncertain areas of the given character. From experience we can say that  $n$  is rarely above





Figure 5.35: A character with its uncertain areas set in orange. The number of uncertain areas is 2; the number of possible characters is  $2^2 = 4$ .

4.



(a) Cost = 669



(b) Cost = 626



(c) Cost = 618



(d) Cost = 575

Figure 5.36: The four candidates obtained from the combination of the uncertain areas in Figure 5.35

Each new character image is a candidate for the restoration of the original character. In the second step, the best candidate is selected. This is done by employing the WIM recognition procedure in Section 5.2. For each candidate, the character template is chosen that has the lowest matching cost. Now, the candidate that matches best to its recognised template, i.e., the candidate with the lowest matching cost, is selected to be the restored version of the character.

In this way, the original character images are repaired and, at the same time, recognised. This approach needs a larger number of matchings for each character that has uncertain areas. The number of matches  $M$  necessary for the restoration can be defined as:

$$M = \sum_{i=1}^{n_C} 2^{n_{U_i}}$$

where  $n_C$  is the number of character images and  $n_{U_i}$  is the number of uncertain areas of character  $i$ .

The second way of choosing the best candidate image is simpler. First, the FTR image is recognised, with the uncertain areas in grey, as in Section 5.2.

Then, the recognised template is matched to all the candidates created from the combinations of the uncertain areas. The candidate with the lowest cost will be chosen to represent the character.

This second approach is faster since each character is only recognised once. Then the best candidate combination is chosen by matching the recognised character to all the possible candidates of the unknown character. However, if the character is misrecognised in the first place, the character could be restored incorrectly ending up broken or touching other edges. After extensive testing it was found that, in most cases, the results of the WIM were the same for both the greyscale image and its candidates, varying only the matching score, and this method was selected as best performer.

## 5.6 Results

The two methods suggested for the character restoration output very similar results. Among them, the faster and simpler method was chosen to display the results. The method chosen is the second method suggested in Section 5.5 where the characters are first recognised and then the candidate with the lowest matching cost to the recognised template would be chosen.

The results obtained by the R-FTR characters can be divided into two. The results obtained by sending the images to the OCR, and the results obtained by recognising the images using the image matching proposed in Section 5.2.

The results obtained by recognising the R-FTR characters with OCRs ABBYY and MODI were 79.03% and 67.05% respectively. In Figure 5.4, these results, plus their comparison to the recognition performance of the original image and the recovered image proposed in Chapter 4 are shown.

OCR	FTR Image	R-FTR	Improvement
<b>MODI</b>	68.68%	67.05%	-1.63%
<b>ABBY</b>	71.39%	79.03%	7.64%

Table 5.4: Recognition rates of the R-FTR images compared to the original images and the FTR images.

In this table can be seen that for the OCR MODI, the recognition of the FTR image was, in fact, a 1.63% better than the R-FTR. However, for the OCR by ABBYY, the recognition has improved a substantial 7.64%, reaching almost 80% recognition rate.



The second results obtained were by recognising with WIM the R-FTR images. These results were very similar to the results of template matching of the FTR images. The average recognition rate of the R-FTR character images was 89.60% and 89.78% for the FTR. This can be explained because the obtained R-FTR images are the FTR images recognised by WIM and based on the results of that matching are then restored. However, the results are not exactly the same and in some cases these are lower than the FTR WIM.

We can say that 0.18% of the restored images were not correctly restored or were restored in a way where other characters would be a better match. The improvement results obtained by comparing the results to the recognition of the restored images by the two OCRs, ABBYY and MODI are shown in Figure 5.37.

R-FTR WIM	OCR	FTR Image	Improvement
89.60%	<b>MODI</b>	68.68%	20.92%
	<b>ABBY</b>	71.39%	18.21%

Figure 5.37: Recognition rates of the restored images by WIM compared to the results obtained by the two OCRs ABBYY and MODI.

The improvement achieved by WIM on the R-FTR images compared to the results obtained by MODI are an impressive 25% and almost 10% compared to the results of ABBYY.

## 5.7 Summary and Conclusions

In this chapter a robust and simple way to recognise typewritten degraded characters was presented. The method is based on an intelligent weighted image matching approach. The character templates, or prototypes, were extracted from typewritten images of similar characteristics. The image matching algorithm was strengthened by measuring the *size* and *distance* of the non-matching areas. The matching distance between a character and its template and the distances between that character and other templates, is measured and used to define the recognition confidence. Measuring the confidence can increase the recognition performance. When the confidence is low, both characters should be considered for recognition. The application of the recognition confidence is suggested as future work.

A set of features was extracted to aid the WIM. Finding robust features for such degraded characters is a major problem and many commonly used and reliable features on higher quality documents could not be used here. The character's *size* has proved to be the most robust and discriminating feature between tall, low and short characters. The remaining three features are based on peak-valley analysis of the projection profiles of the characters, horizontal and vertical. Although projection profile based features are not very popular among image recognition systems, there have been found to be robust for typewritten characters where the strokes present breaks, changes in the thickness or joined ends. In addition, typewritten characters are free of orientation, size and slant changes, that would modify the profiles. The vertical and horizontal projection profile features might not perform so well in slanted typewritten fonts.

The results obtained by the WIM have improved considerably the results of the two OCRs used for this comparison. As well, using the image matching has allowed the possibility to restore the FTR characters based on the identified uncertain areas (R-FTR). The R-FTR restores the characters, increasing the recognition performance by providing a binary representation of the characters. Only the characters where key uncertain areas have been identified could be restored.

The whole restoration system depends directly on the character templates. If the character templates were modified, thickened or of a different font, the R-FTR method would produce different results.

This R-FTR method works best if all the pixels in the uncertain area belong to either the characters foreground or the background. If the pixels of an uncertain area belong some to the foreground and some to the background, this method might not restore the character optimally. It would require a partition of the uncertain area for a correct restoration.



# Chapter 6

## Results

### 6.1 Introduction

In this chapter the results of the enhancement and recognition methods proposed in this thesis are discussed. To obtain this, two different OCR systems will be used for recognition and results comparison.

The structure of this chapter is described as follows. In Section 6.2, a description of the data set chosen for the tests will be given. In Section 6.3, the techniques used to measure the recognition rates and the kinds of errors evaluated will be discussed. In Section 6.4, the recognition results of the original images will be shown, highlighting their main weaknesses and recognition errors. In Section 6.5, the recognition results of the Flexible Text Recovered (FTR) images by the same OCR systems will be presented, followed by the results of their restored version (R-FTR). These results will be compared to three well known thresholding methods, the global method by Otsu [63], Sauvola and Pietaksinen adaptive method [76] and Gatos et al. [31] method for historical images, in Sections 6.5.1, 6.5.2 and 6.5.3 respectively.

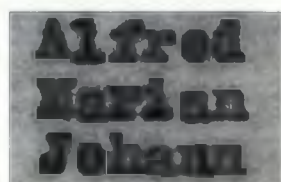
In Section 6.6, the results of the WIM method proposed in Chapter 5 are presented and compared to the results of the two commercial OCRs, and finally, Section 6.7 will discuss the overall results obtained.

### 6.2 The Data Set

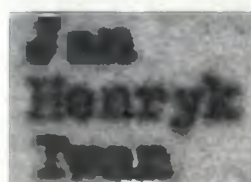
The data set comprised a total of approximately 11000 characters from 45 different documents. The images were selected manually from the data set used

in the project MEMORIAL [5, 6]. The quality, and the way in which these documents have been created vary. The images were selected to create a representative set containing the kind of degradations expected in such documents. Two main types of documents were selected. The first type is comprised of a set of heavily printed *File Cards* (FC) and the second type is comprised of highly blurred *Carbon Copies* of prisoners transport lists (CC).

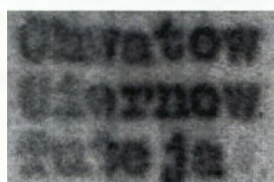
To better compare the performance of the different chosen methods, the CC document images were categorised by quality. The CC images have been divided, by their level of quality, into *high*, *medium* and *low*. The quality of the CC images is generally lower than the one of the FC documents creating different challenges for the recognition process. In general CC images are highly blurred and their foreground and background are textured. Sample images of the *high*, *medium* and *low* CC categories are shown in Figures 6.1(a), 6.1(b) and 6.1(c) respectively.



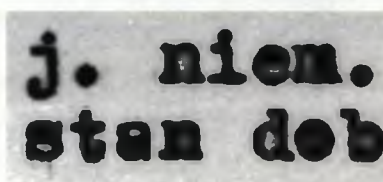
(a) High quality CC sample



(b) Medium quality CC sample



(c) Low quality CC sample



(d) FC sample

Figure 6.1: Images categories used in the results statistics

On the other hand, the FC images present higher contrast and better defined strokes. However, they also contain a number of degradations such as highly pressed characters. These can close up the inner holes of the characters and create thicker strokes making some characters and strokes touch. An example of a card image is given in Figure 6.1(d).

From the data set selected for testing, the chosen *high quality* CC images correspond to 37.04% of the total characters and the *medium* and *low quality* CC classes correspond to 36.89% and 12.01% respectively. The FC images represent by 14.05% of the total (see Table 6.1).

	Data Set Images
High Quality CC Images	37.04%
Medium Quality CC Images	36.89%
Low Quality CC Images	12.01%
FC Images	14.05%

Table 6.1: Description of the images contained in the data set categorised by quality and type.

These classes have been created based on the results obtained in the tests. They can be also classified visually as the *high* quality level images show higher contrast and less blur than the lower qualities. *Medium* and *low* quality images are both highly blurred. However, the *low* quality level also contains other degradations that, in some cases, can affect the readability. Both these classes are comprised by carbon copy images. Due to the difference in quality and type of degradation between the CC and the FC images, the FC images have been categorised separately.

### 6.3 Measuring Recognition Errors

Two OCR systems have been chosen to recognise the images for testing and result comparison, the Microsoft Office Document Imaging (MODI) and the OCR system by ABBYY Finereader Professional.

MODI has been chosen as a standard OCR which is broadly available and inexpensive since it is free to download and it is also included with Microsoft Office XP and above. MODI is a high quality OCR created by ScanSoft (Nuance). The second OCR used, ABBYY Finereader 6.0 Professional, was selected as a rival high quality OCR engine that also has specific support for typewritten fonts.

To compare the OCR results, a number of valuable measurements have been extracted from the resulting recognised text. These include the *total number of characters*, the *total number of correctly recognised characters* and the *number of incorrectly recognised characters*, and as well, the *number of missed characters* and the *number of noise characters*. A *missed character* is considered to be any character that was present in the original image but the OCR failed to recognise or detect and therefore is missing in the recognised version. *Spurious*, or

*noise characters* are the opposite case, that is, characters that appear in the recognised version of the image, but do not exist in the original document and are probably result of noise in the image.

A total of approximately 11000 characters were analysed to obtain these results. The task of comparing the groundtruth of each image and its recognised version was done using the *Levenshtein Distance* [53]. The groundtruth representation of each chosen image was manually created. This task was lengthy and complicated in some cases, since some characters were too faded and blurred to be confidently recognised even by a human reader. See an example in Figure 6.2.



Figure 6.2: Sample image of a word difficult to recognise by a human.

The Levenshtein distance is a string metric which measures the edit distance. The Levenshtein distance between two strings is given by the minimum number of *operations* needed to transform one string into the other, where an operation is an *insertion*, *deletion*, or *substitution* of a single character [53].

The algorithm in pseudo code looks like this:

```
int LevenshteinDistance(char s[1..m], char t[1..n])
{
    // d is a table with m+1 rows and n+1 columns
    declare int d[0..m, 0..n]

    for i from 0 to m
        d[i, 0] := i
    for j from 1 to n
        d[0, j] := j

    for i from 1 to m
        for j from 1 to n
            if s[i] = t[j] then cost := 0
                else cost := 1
            d[i, j] := minimum(
```



```

        d[i-1, j] + 1,           // deletion
        d[i, j-1] + 1,         // insertion
        d[i-1, j-1] + cost     // substitution
    )
    return d[m, n]
}

```

		S	a	t	u	r	d	a	y
	0	1	2	3	4	5	6	7	8
S	1	<u>0</u>	<u>1</u>	<u>2</u>	3	4	5	6	7
u	2	1	1	2	<u>2</u>	3	4	5	6
n	3	2	2	2	3	<u>3</u>	4	5	6
d	4	3	3	3	3	4	<u>3</u>	4	5
a	5	4	3	4	4	4	4	<u>3</u>	4
y	6	5	4	4	5	5	5	4	<u>3</u>

Figure 6.3: Example of Levenshtein distance of strings “Saturday” and “Sunday”

The matrix created by the Levenshtein algorithm is shown in Figure 6.3. The bottom-right element of the matrix contains the minimum edit distance between the two strings.

The distance on its own was found to be insufficient for the statistics and after a small addition to the algorithm, the number of *insertions*, *deletions* and *substitutions* were also obtained. The number of *insertions* represents the number of *missing characters* in the OCR output. The number of *deletions* represents the number of *false or noise characters*, probably created by the OCR due to the presence of noise, and the number of *substitutions* represents the number of *incorrectly recognised characters*.

To obtain the number of *incorrect*, *missing* and *noise* characters from the Levenshtein matrix, as shown in Figure 6.3, the reverse minimum steps are followed, starting from the bottom-right corner. The minimum steps path is determined then, by moving towards the lowest value, between the value above, to the left and to the diagonal up-left, until the opposite corner is reached (the underlined numbers in the matrix). In this way, the number of correct characters is determined by each move to the diagonal up-left cell, where the value of both cells is the same. A character has been incorrectly recognised if the



value on the left-up diagonal is lower than the one in the current cell. A vertical movement indicates that there was a missing character, and a move to the left would correspond to a noise character. In this way the number of missing, noise and incorrectly recognised characters was calculated. The Levenshtein distance, however, could not determine the number of segmentation errors.

## 6.4 The Original Images

As described in previous chapters, the original document images can suffer from severe ageing, such as blur, textured background, noise or over-pressed thickened characters. As expected, the results of applying OCR directly onto these highly degraded images resulted in very low recognition rates.

Original	OCR	Recognition Rate
High Quality CC	MODI	66.34%
	ABBYY	52.85%
Medium Quality CC	MODI	54.81%
	ABBYY	56.87%
Low Quality CC	MODI	42.43%
	ABBYY	28.20%
All CC Images	MODI	54.80%
	ABBYY	52.85%
FC Images	MODI	69.82%
	ABBYY	70.15%

Table 6.2: Recognition rates of original images by OCR and by quality.

Original	OCR	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	MODI	32.55%	25.31%	42.13%
	ABBYY	3.05%	31.11%	65.83%
Medium Quality CC	MODI	20.13%	28.97%	48.66%
	ABBYY	3.33%	35.44%	61.21%
Low Quality CC	MODI	7.10%	30.87%	47.66%
	ABBYY	3.28%	43.02%	53.51%
All CC Images	MODI	20.84%	31.42%	47.73%
	ABBYY	3.23%	35.93%	60.79%
FC Images	MODI	18.32%	37.27%	44.39%
	ABBYY	45.98%	36.54%	17.46%

Table 6.3: Percentages of missing, noise and incorrect characters in the original images by OCR and by quality.

Table 6.2 shows the results of the recognition rate of the original images by both OCRs. The table displays the recognition rates categorised by the three

CC quality classes selected, plus the average of all the CC images and the results of the FC images. The average results for both OCRs are very similar, being MODI slightly over ABBYY in all the three quality sections. The recognition results of CC are in all cases under 60%. In addition, the layout analysis by both OCRs failed as well on several occasions. In Table 6.3 the table displays the percentage of each of the errors measured for these images. As can

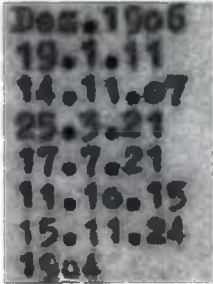
	.s.19*6 19.1.11 14.1*..? t5.Ji 17.7.21 11. 15.11.24 19.4	B**.!\$o6 19.1.11 14.11+97 25.3.?1 17.7.21 11.13.15 i5.11.24 19t>4
(a) Original image Sample	(b) MODI	(c) ABBYY

Figure 6.4: Original image sample with recognition results of MODI and ABBYY OCRs.

be seen in this table, the errors made by MODI are fairly divided among the three types of errors. ABBYY's errors, on the other hand, produced only less than 5% noise and around 60% were misrecognition errors. This can be interpreted as ABBYY being able to extract more characters from the document, and being less sensitive to noise. However, the recognition failed, probably due to an inappropriate thresholding technique. In conclusion, the original CC images underperformed when recognised by two standard OCRs where noise and an inappropriate binarisation technique were probably the reasons for most errors.

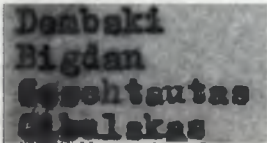
	Db ski lgdan Gesohtautas QShvJ aka	Dambaki BlgAan Seaohtautaa bnlaka
(a) Original image	(b) MODI	(c) ABBYY

Figure 6.5: Original CC image sample with recognition results of MODI and ABBYY OCRs.

Figures 6.4 and 6.5 show two sample CC images and their corresponding recognised text by both OCRs, MODI and ABBYY.

The recognition of the original FC images by the two OCRs will be presented next. The results obtained by these images were considerably better than the results of CC. These images presented a clearer background and distinct and well formed characters. At the bottom of Table 6.2 the recognition results for the FC by both OCRs are shown.

The results obtained by MODI are comparable to the ones obtained on CC of high quality. On the other hand, the results obtained by ABBYY have improved by 18%. However, both methods still performed poorly when the images showed darkened inner character holes. This is probably due to overpressed characters, to dirt in the typewriter key or excessive ink in the ribbon. In Figure 6.6 a sample image is shown with the recognised text by both OCRs. Both OCRs failed to recognise some of the filled in characters, such as 'a' and 'e'.

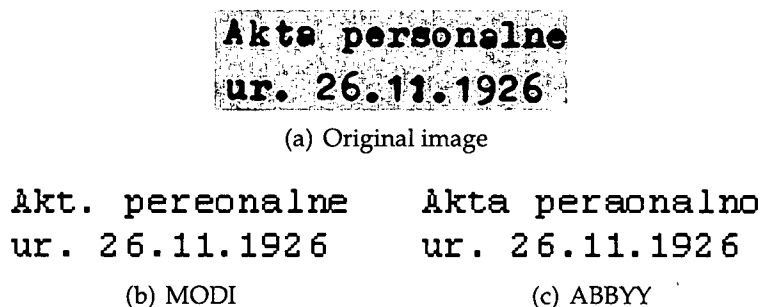


Figure 6.6: Original FC image sample with recognition results of MODI and ABBYY OCRs.

## 6.5 Flexible Text Recovery Results

The results on the Flexible Text Recovery method (FTR) proposed in Chapter 4 will be presented next. In Table 6.4, the recognition rates of both the Original images and the FTR images by the two OCRs are shown highlighting the improvement gained by the FTR method on the right column. The results of the recognition after FTR by both commercial OCRs are greatly improved. High quality CC images reach up to 82% correctly recognised characters, up to 76% for the medium quality CC images. ABBYY achieved higher improvement than MODI reaching almost 28% for the high quality CC images and almost



20% for the medium quality and an impressive 24% for the low quality images. MODI on the other hand did not improve so much for the low quality images, less than 3%, but improved almost 15% on average.

	OCR	FTR Image	Original Image	Improvement
High Quality CC	MODI	82.04%	66.34%	15.70%
	ABBY	80.49%	52.85%	27.64%
Medium Quality CC	MODI	73.42%	54.81%	18.61%
	ABBY	76.41%	56.87%	19.54%
Low Quality CC	MODI	44.51%	42.43%	2.08%
	ABBY	52.84%	28.20%	24.64%
All CC Images	MODI	66.66%	54.80%	11.86%
	ABBY	69.91%	52.85%	17.06%
FC Images	MODI	74.75%	69.82%	4.93%
	ABBY	75.82%	70.15%	5.67%

Table 6.4: Results improvement between the recognition results of the original CC images and the FTR CC images

The types of errors created by the OCRs in the FTR images are shown in Table 6.5.

FTR	OCR	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	MODI	34.94%	19.07%	45.98%
	ABBY	4.35%	24.24%	71.39%
Medium Quality CC	MODI	24.16%	26.58%	48.47%
	ABBY	3.28%	24.29%	72.24%
Low Quality CC	MODI	14.23%	22.88%	62.87%
	ABBY	11.09%	14.24%	74.65%
All CC Images	MODI	24.94%	22.92%	51.68%
	ABBY	5.50%	21.85%	72.56%
FC Images	MODI	24.90%	37.85%	37.24%
	ABBY	55.20%	37.25%	23.15%

Table 6.5: Percentages of noise, missing and incorrect characters of FTR by OCR and by quality.

Although FTR documents have been cleaned and are mostly noise free, MODI created a considerable amount of errors created from noise, making up almost 25% of its mistakes. From this we can interpret that the OCR creates additional characters when it fails to correctly segment the characters, and, in that way, attempts to recognise the joined or broken characters leading to failure.

On the other hand, ABBYY responded to noise better, with only 5.5% of the recognition errors. The typewriter setting in ABBYY was selected and the number of segmentation errors was visually lower than MODI. Both methods

however, omitted a large amount of characters during recognition being these around 22% of their errors. A sample of an FTR CC image with its recognised text version is shown in Figures 6.7.

**B y k**  
**Brutwinas**  
**Bendoraitis**  
**Bankowski j**

(a) FTR CC image

Byk

Brutwina a

Bendoraiti a

Bankowakij

(b) MODI

B y k

Brntwinas

Bendoraitl s

Bankowaklj

(c) ABBYY

Figure 6.7: FTR CC image sample with recognition results of MODI and ABBYY OCRs.

For the FC images, both OCRs have improved their performance in comparison with the recognition rates obtained by the original version. The improvement for MODI's OCR is of 4.93% and 5.67% for ABBYY. The improvement achieved on the recognition of the FC images is not as high as on the CC images. See bottom of Table 6.4 for the improvement obtained by FC images compared to the original images recognition rates. A sample of an FTR FC image with its recognised text version is shown in Figures 6.8.

Although layout analysis results were not measured here, there is a visual improvement in the recognition results of the FTR images that show a better preserved layout, maintaining the character size. In addition, in most cases, the characters are recognised in a Courier font resembling the typewritten characters.

In addition to the FTR images, the Restored FTR (R-FTR) images were also



**Akta personalne  
ur. 26.9.1923 r**

(a) FTR FC image

Akta peraonalna  
ur. 26.9.1923 r

(b) MODI

Akta personalne  
ur. 26.9.1923 r

(c) ABBYY

Figure 6.8: FTR FC image sample with recognition results of MODI and ABBYY OCRs.

tested. To restore the images, the faster, second method proposed in Section 5.5 was chosen. In this method, the images are restored based on the recognition output of the FTR image. The R-FTR images are bi-level and have been restored to maintain connectivity and shape.

In Table 6.6 the recognition results of the R-FTR images and the comparison of the recognition rates of the FTR and the R-FTR methods are given. The results obtained are interesting when compared to the results obtained by the FTR CC images. For ABBYY's OCR, the R-FTR images have improved the recognition up to 9.19% in average. On the other hand, MODI seemed to respond better to the ternary FTR images and actually its recognition rate has degraded by an average of 3.46%.

	OCR	R-FTR	FTR	Improvement
High Quality CC	MODI	71.39%	82.04%	-10.65%
	ABBYY	88.87%	80.49%	8.38%
Medium Quality CC	MODI	67.21%	73.42%	-6.21%
	ABBYY	84.22%	76.41%	7.81%
Low Quality CC	MODI	50.99%	44.51%	6.48%
	ABBYY	64.21%	52.84%	11.37%
All CC Images	MODI	63.20%	66.66%	-3.46%
	ABBYY	79.10%	69.91%	9.19%
FC Images	MODI	78.60%	74.75%	3.85%
	ABBYY	78.82%	75.82%	3.00%

Table 6.6: Results improvement of the recognition rates between the R-FTR CC images and the FTR CC images

In Table 6.7 the types of errors created by both OCRs in the R-FTR images are shown. A difference can be appreciated in the error rates for both OCRs. Most of the ABBYY errors are due to misrecognised characters, these being

these 80% of the total errors. The noise and missing character errors have been reduced to 10% each. The number of missing characters was reduced compared to the one of FTR and most errors are now just due to incorrect recognition.

R FTR	OCR	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	MODI	44.31%	12.01%	43.22%
	ABBYY	12.61%	10.36%	77.02%
Medium Quality CC	MODI	36.11%	19.71%	44.04%
	ABBYY	7.13%	12.73%	79.92%
Low Quality CC	MODI	21.18%	19.56%	58.86%
	ABBYY	8.19%	5.36%	86.43%
All CC Images	MODI	33.87%	17.09%	48.71%
	ABBYY	9.31%	9.48%	81.12%
FC Images	MODI	20.58%	47.04%	32.37%
	ABBYY	43.77%	42.58%	13.63%

Table 6.7: Percentages of noise, missing and incorrect characters of R-FTR images by OCR and by quality.

On the other hand, the errors created by MODI are 40% noise characters. This high amount could be interpreted as a failure in the segmentation by the OCR breaking some characters into two and, therefore, outputting two recognised characters instead of one. The extra character recognised by the OCR would be interpreted as noise when comparing it to the groundtruth.

A sample of an R-FTR CC image with its corresponding recognised text by both OCRs is shown in Figure 6.9. In this figure, the text recognised by MODI shows one common segmentation error as mentioned above. In Figure 6.9(b) on the first line, the character 'k' has been recognised by MODI as 'ic'. The character segmentation from MODI has divided the character 'k' into two and attempted to recognise them separately. Note that the character 'k' in the R-FTR image is not broken or touching other characters.

The results of the R-FTR FC images will be discussed next. In comparison with the results obtained by FTR, R-FTR has improved the recognition of FC images a further 3% for both OCRs as shown in Table 6.6. Both methods, the FTR and the R-FTR improve the results obtained by the original FC images. An example of an R-FTR FC image with its corresponding recognised text by both OCRs is shown in Figure 6.10. In the next section, their results will be compared to the results achieved by three well known thresholding methods, Otsu, Sauvola and Pietaksinen and Gatos et al.

**Binkauskas  
Janeliauskas  
Danilewitschius  
Tschechawitschus**

(a) R-FTR CC image sample

Binkausicas	Binkauskas
Jameliauskas	Janeliauskas
Danilewitschius	Danilewitschiue
Taohechawitachus	Tschechawitschus

(b) MODI

(c) ABBYY

Figure 6.9: R-FTR CC image sample with recognition results of MODI and ABBYY OCRs.

**nia: D o m o n  
Kowno /Litwa/**

(a) R-FTR FC image

riia: D o m o n	nia: D o m o n
Kowno /Litwa/	Kowno /Litwa/

(b) MODI

(c) ABBYY

Figure 6.10: R-FTR FC image sample with recognition results of MODI and ABBYY OCRs.

### 6.5.1 Comparison to Otsu's method

In this section, the results obtained by thresholding the original images with Otsu's method [63] will be given. The implementation of Otsu's method was performed using MATLAB. Otsu's global thresholding method, failed to output readable results in many images with variable intensity.

The CC images were mostly over-thresholded producing closed holes and touching characters, and in some cases completely rendering the characters unreadable by transforming the characters into black blobs. In Figure 6.11 an



example of Otsu's output in CC images and its correspondent recognised versions is shown.

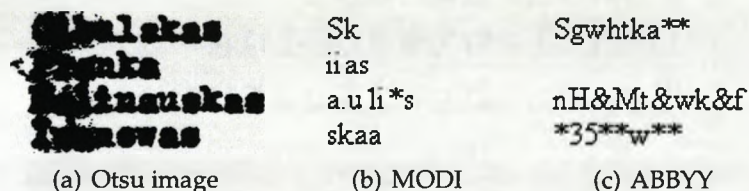


Figure 6.11: Otsu CC image sample with recognition results of MODI and ABBYY OCRs.

The recognition rates of the CC and FC images thresholded with the OTSU method are shown in Table 6.8. The results achieved by Otsu's thresholding method on the CC images are the lowest so far, and performed worse than the original images without any pre-processing. In comparison with the FTR method, the recognition rates are inferior by 30%. When applying Otsu's thresholding method to the FC images the OCR results were similar to those obtained by the original FC images. Both OCR results obtained by the FC images, recovered by FTR and R-FTR methods, outperform those obtained by Otsu's method. The improvement is found to be dependent on the OCR. MODI reflects a greater improvement reaching more than 12%. ABBYY's results have also been improved up to 7%. ABBYY, was more efficient in locating

	OCR	Otsu	FTR	Improvement
High Quality CC	MODI	51.48%	82.04%	30.56%
	ABBYY	57.48%	80.49%	23.01%
Medium Quality CC	MODI	46.36%	73.42%	27.06%
	ABBYY	45.56%	76.41%	30.85%
Low Quality CC	MODI	14.13%	44.51%	30.38%
	ABBYY	25.05%	52.84%	27.79%
All CC Images	MODI	54.80%	66.66%	11.86%
	ABBYY	52.85%	69.91%	17.06%
FC Images	MODI	65.82%	74.75%	8.93%
	ABBYY	71.72%	75.82%	4.10%

Table 6.8: Results improvement between OTSU images and FTR images

the individual characters and most of its errors were due to misrecognition. The error rates for the images thresholded by Otsu's method can be seen in Table 6.9.

A comparison of the results of the FTR and R-FTR methods to Otsu is shown in Tables 6.8 and 6.10.

OTSU	OCR	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	MODI	19.42%	46.53%	34.03%
	ABBYY	10.14%	15.80%	74.01%
Medium Quality CC	MODI	19.14%	44.70%	35.18%
	ABBYY	5.95%	19.11%	74.82%
Low Quality CC	MODI	6.64%	28.45%	64.90%
	ABBYY	6.16%	32.72%	61.03%
All CC Images	MODI	10.95%	52.04%	24.81%
	ABBYY	7.41%	22.55%	69.95%
FC Images	MODI	18.94%	32.52%	48.52%
	ABBYY	41.87%	28.11%	30.00%

Table 6.9: Percentages of noise, missing and incorrect characters of Otsu by OCR and by quality.

	OCR	Otsu	R-FTR	Improvement
High Quality CC	MODI	51.48%	71.39%	19.91%
	ABBYY	57.48%	88.87%	31.39%
Medium Quality CC	MODI	46.36%	67.21%	20.85%
	ABBYY	45.56%	84.22%	38.66%
Low Quality CC	MODI	14.13%	50.99%	36.86%
	ABBYY	25.05%	64.21%	39.16%
All CC Images	MODI	54.80%	63.20%	8.40%
	ABBYY	52.85%	79.10%	26.25%
FC Images	MODI	65.82%	78.60%	12.78%
	ABBYY	71.72%	78.82%	7.10%

Table 6.10: Results improvement between the CC OTSU images and the R-FTR images

The results of the recognition of the Otsu FC images improves the recognition obtained by the original images See Figure 6.12 for a sample FC image thresholded by Otsu plus its OCR text outputs.

**D o m o w i t z S e g e i n e ,**

(a) Otsu FC image

D \* o w i t z S e g e i n . , D \* m ^ w i t z S e g e i n e ,

(b) MODI

(c) ABBYY

Figure 6.12: Otsu FC image sample with recognition results of MODI and ABBYY OCRs.



### 6.5.2 Comparison to Sauvola and Pietaksinen's method

The next thresholding method chosen was the one of Sauvola and Pietaksinen (SP) [76]. The result of applying the SP method directly on the original CC images creates very noisy characters. Both OCRs failed to recognise any text on the CC images. In an attempt to enhance its performance, the images were smoothed to reduce the noise. However, the quality obtained from this operation was still not good enough for the OCRs. An example of both outputs can be seen in Figure 6.13.



Figure 6.13: Original CC image sample with its SP and smoothed SP outputs.

On the CC images the SP adaptive method resulted in images too noisy for both OCRs and the recognition was unsuccessful. Comparisons to this method were not performed for the CC images. However, the results of applying SP method on the FC images were better due to their cleaner and less textured background and the comparison to these images was carried out. The results obtained by SP method in the FC images are shown in Table 6.11.

Although both OCRs were able to recognise text in the FC images, the recognition rates obtained are slightly lower than the results obtained by the original FC images. As in previous cases, both methods proposed here, FTR and R-FTR, outperform SP results. In Tables 6.11 and 6.12, the recognition improvement of the FTR and R-FTR methods in comparison with the ones obtained by SP method is shown.

	OCR	SP	FTR	Improvement
FC Images	MODI	68.70%	74.75%	6.05%
	ABBY	68.34%	75.82%	7.48%

Table 6.11: Results improvement between the FC SP image and the FTR image

A sample of an FC image thresholded by SP method and its corresponding recognised text is shown in Figure 6.14.

	OCR	SP	R-FTR	Improvement
FC Images	<b>MODI</b>	<b>68.70%</b>	78.60%	9.90%
	<b>ABBYY</b>	<b>68.34%</b>	78.82%	10.48%

Table 6.12: Results improvement between the FC SP image and the R-FTR image

**Ak-ta personalne**

(a) SP FC image

**Akttta4personaine . Al6Stja\*personalne**

(b) MODI

(c) ABBYY

Figure 6.14: SP FC image sample with recognition results of MODI and ABBYY OCRs.

### 6.5.3 Comparison to Gatos et al. method

Finally, the results of the FTR and R-FTR methods here proposed will be compared to the method by Gatos et al. (GPP) [31]. As described in Chapter 2, this method is aimed at historical, noisy and degraded document images. The results of the recognition by the OCRs in the CC images processed by the GPP method can be seen in Table 6.13. The results obtained are slightly better for

	OCR	GPP	FTR	Improvement
High Quality CC	<b>MODI</b>	<b>70.54%</b>	82.04%	11.50%
	<b>ABBYY</b>	<b>62.27%</b>	80.49%	18.22%
Medium Quality CC	<b>MODI</b>	<b>47.95%</b>	73.42%	25.47%
	<b>ABBYY</b>	<b>42.41%</b>	76.41%	34.00%
Low Quality CC	<b>MODI</b>	<b>38.47%</b>	44.51%	6.04%
	<b>ABBYY</b>	<b>26.80%</b>	52.84%	26.04%
All CC Images	<b>MODI</b>	<b>54.86%</b>	66.66%	11.80%
	<b>ABBYY</b>	<b>44.81%</b>	69.91%	25.10%
FC Images	<b>MODI</b>	<b>73.38%</b>	74.75%	1.37%
	<b>ABBYY</b>	<b>80.39%</b>	75.82%	-4.57%

Table 6.13: Results improvement between the GPP image and the FTR CC image

MODI reaching up to 70% recognition rates for the higher quality CC images. The average of the CC images is 54% for MODI and 44% for ABBYY. The GPP method was also applied to the file card (FC) images. This method was found to perform very well in the cleaner FC images.

The type of errors produced by the OCRs in the GPP CC images are shown



in the table in Table 6.14. As seen in images 6.15(b) and 6.15(c), both OCRs interpreted large amounts of characters from the noisy background on the right hand side of the GPP image (see 6.15(a)).

Gatos	OCR	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	MODI	35.05%	7.53%	55.78%
	ABBYY	15.83%	17.18%	66.98%
Medium Quality CC	MODI	17.80%	26.05%	53.65%
	ABBYY	18.51%	9.60%	71.60%
Low Quality CC	MODI	16.11%	31.80%	49.21%
	ABBYY	14.96%	14.40%	62.59%
All CC Images	MODI	22.08%	23.18%	52.25%
	ABBYY	16.82%	13.11%	69.84%
FC Images	MODI	22.47%	37.86%	39.65%
	ABBYY	51.55%	36.04%	12.40%

Table 6.14: Percentages of missing, noise and incorrect characters of GPP by OCR and by quality.

When comparing these results to the performance of the FTR method, the later shows an improvement for ABBYY's OCR of 27% in average and almost 15% percent improvement for the MODI OCR (see Table 6.13). In addition, the R-FTR method performs in average 10% better using MODI's OCR and up to 35% if using ABBYY. See Table 6.15 for a comparison between R-FTR and the method by GPP. A sample of a GPP CC image and its OCR text output is shown

	OCR	GPP	R FTR	Improvement
High Quality CC	MODI	70.54%	71.39%	0.85%
	ABBYY	62.27%	88.87%	26.60%
Medium Quality CC	MODI	47.95%	67.21%	19.26%
	ABBYY	42.41%	84.22%	41.81%
Low Quality CC	MODI	38.47%	50.99%	12.52%
	ABBYY	26.80%	64.21%	37.41%
All CC Images	MODI	52.32%	63.20%	10.88%
	ABBYY	43.82%	79.10%	35.28%
FC Images	MODI	73.38%	78.60%	5.22%
	ABBYY	80.39%	78.82%	-1.57%

Table 6.15: Results improvement between GPP images and R-FTR CC images

in Figure 6.15. The GPP method is the most competitive of the three methods chosen for comparison since it was specially created for degraded historical documents. In the FC images, the results of ABBYY were slightly better when using the GPP than the methods here proposed. However, the results obtained by MODI's OCR are better for the FTR and R-FTR methods. An example of the

**Marian  
Pranoiszek  
Czesław**

(a) GPP image

Ma.ri.an: \_tçE '  
!Pranoiszek  
O!30w1aw

(b) MODI

Marlaa - ' ' '  
nra&eisg^k  
Sseslaw .,,-''''

(c) ABBYY

Figure 6.15: GPP image sample with recognition results of MODI and ABBYY OCRs.

**D o m o n e s o w  
ur. 26.9.1923 r.**

(a) GPP FC image

O o m o n o a o w  
in'. 26.9.1923 r.

(b) MODI

D o m o n e s o w  
ur. 2o.9.1923 r.

(c) ABBYY

Figure 6.16: GPP FC image sample with recognition results of MODI and ABBYY OCRs.

FC image enhanced by the GPP method with its corresponding recognised text by both OCRs, ABBYY and MODI is shown in Figure 6.16.

## 6.6 Comparison of Character Recognition Results

In this section the recognition results obtained by the template matching OCR proposed in Chapter 5 will be discussed. To obtain these results, two sets of images were recognised: the images enhanced by the FTR method (see Chapter 4) and the R-FTR method (see Section 5.5).

The results of each matching were saved in a text file maintaining the local

positions of the characters in the document to preserve the layout. Levenshtein's distance was used to compare the images groundtruth to the recognised version as in previous sections.

The results obtained by WIM on the FTR images are shown in Table 6.16.

FTR WIM	Recognition Rate
High Quality CC	94.35%
Medium Quality CC	90.67%
Low Quality CC	80.54%
All CC Images	88.52%
FC Images	93.54%

Table 6.16: Recognition rates of the FTR images by the WIM method proposed.

The recognition results achieved by the template matching on the FTR image improve those obtained by ABBYY and MODI. In Table 6.17 the obtained results are compared to those obtained from both OCRs. In average for the CC images, an improvement of almost 22% has been achieved in comparison to MODI's recognition performance, and almost 19% when comparing to ABBYY. An outstanding recognition rate of more than 94% was achieved in the

	OCR	FTR	FTR WIM	Improvement
High Quality CC	MODI	82.04%	94.35%	12.31%
	ABBYY	80.49%		13.86%
Medium Quality CC	MODI	73.42%	90.67%	17.25%
	ABBYY	76.41%		14.26%
Low Quality CC	MODI	44.51%	80.54%	36.03%
	ABBYY	52.84%		27.70%
All CC Images	MODI	66.66%	88.52%	21.86%
	ABBYY	69.91%		18.61%
FC Images	MODI	74.75%	93.54%	18.79%
	ABBYY	75.82%		17.72%

Table 6.17: Results improvement between the images recognised by WIM and the ones recognised by the commercial OCRs

higher quality CC images reaching an average recognition rate for the CC images of 88%. This can be considered a significant recognition rate for such



degraded images. In Table 6.18, the types of errors are displayed for the WIM recognition results.

FTR WIM	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	2.92%	7.65%	89.42%
Medium Quality CC	2.19%	7.39%	80.23%
Low Quality CC	5.45%	13.33%	81.20%
All CC Images	3.52%	9.46%	83.62%
FC Images	2.22%	83.80%	89.40%

Table 6.18: Percentages of missing, noise and incorrect characters obtained by WIM the FTR images arranged by the level of quality.

As can be appreciated in this table, the errors created by the WIM recognition system are mainly due to misrecognised characters where the noisy characters are reduced to 3% and missing characters to less than 10% in average. This reduction in the noise and missing characters represents an improvement in the performance of the recognition tool. Misrecognised characters are recoverable and can be sent back for recognition at some stage as opposed to missing characters. The recognition rate of the FTR FC images reaches 93% outperforming the rates obtained by all the other methods and OCRs previously described.

The results obtained from recognising the R-FTR images with WIM were very similar to the results of the FTR. The WIM recognition results for the R-FTR images are shown in Table 6.19. As shown in Section 5.5, the R-FTR images are obtained from the FTR images by computing all the possible character images obtained by switching on and off the identified areas of uncertainty. The best match of the resulting character images is then chosen as its restored version. For this reason, the WIM results for both FTR and R-FTR images, are very similar. In Table 6.20, a comparison of the results of the WIM R-FTR and R-FTR when recognised by MODI and ABBYY are shown. The recognition improvement for the CC images is more notable for MODI's results reaching 25% on average. The recognition results for the FC images have improved more than 14%.

The error rates obtained from applying WIM to the restored images are

R-FTR WIM	Recognition Rate
High Quality CC	94.10%
Medium Quality CC	90.18%
Low Quality CC	80.67%
All CC Images	88.32%
FC Images	93.44%

Table 6.19: Recognition rates of the R-FTR images by the WIM method proposed.

	OCR	R-FTR	R-FTR WIM	Improvement
High Quality CC	MODI	71.39%	94.10%	22.71%
	ABBYY	88.87%		5.23%
Medium Quality CC	MODI	67.21%	90.18%	22.97%
	ABBYY	84.22%		5.96%
Low Quality CC	MODI	50.99%	80.67%	29.68%
	ABBYY	64.21%		16.46%
All CC Images	MODI	63.20%	88.32%	25.12%
	ABBYY	79.10%		9.22%
FC Images	MODI	78.60%	93.44%	14.84%
	ABBYY	78.82%		14.62%

Table 6.20: Results improvement between the FTR images recognised by WIM and the ones recognised by the commercial OCRs

shown in Table 6.21. The errors created by the matching of the R-FTR images are also similar to those of the FTR images. The errors created by WIM on the R-FTR images are also similar to those obtained by the FTR images.

After obtaining the recognition and misrecognition statistics from the WIM results, a study was performed to identify the most repeated types of misrecognition. To perform the study, each misrecognised character by the WIM method here proposed was stored together with its groundtruth version. The most repeated misrecognised pairs of characters were selected. Having the knowledge of the most repeated errors in the recognition can lead to future improvements in the recognition tool by refining the classifier and/or the matching. To display the results, only the most repeated or meaningful errors for each character will be shown. See Table 6.22.



R-FTR WIM	Noise Rate	Missing Rate	Incorrect Rate
High Quality CC	4.03%	9.87%	86.09%
Medium Quality CC	2.14%	6.19%	91.66%
Low Quality CC	3.94%	3.94%	92.11%
All CC Images	3.37%	6.67%	89.95%
FC Images	4.71%	7.39%	87.90%

Table 6.21: Percentages of missing, noise and incorrect characters of the WIM R-FTR images arranged by the level of quality.

(a)

a	e, n, c	s	e
b	h	t	i
c	o, r	u	n, h
d	c, 9	v	w
e	c, a	w	n, r
f	T	x	r
g	a	y	v
h	b	z	c, s
i	t	0	O, C
j	t	1	i
k	R	2	P
l	1, i	3	9, 2
m	n	4	j
n	u, a	5	9
o	c, e	6	5
p	o	7	V
q	g	8	0, 5
r	n, a	9	1, 5

Table 6.22: Most common template matching errors.

From this study it was found that small characters are more often misrecognised than tall characters. The most misrecognised characters in our data set were 'a', 'o', 'c' and 's'. These characters were misrecognised a greater percentage of times than the rest of misrecognised characters. All four of them appeared to be misrecognised in repeated occasions with character 'e' (see Table 6.22). Character 'c' was found to be mistaken repeatedly by a considerable number of characters, such as 'a', 'd', 'e', 'o', 's', or 'z'. The small size of these

character images and a similar physical appearance are considered the main reason for these mistakes.

## 6.7 Conclusion

In this chapter, a comprehensive description of the results obtained by the methods here proposed has been presented. To obtain the results, two professional OCRs, ABBYY and MODI, were used.

To compare the performance of the typewritten text recovery techniques here proposed, FTR and R-FTR, three well known methods were employed, Otsu's global thresholding method, Sauvola and Pietaksinen adaptive thresholding method, and Gatos et al. thresholding method for degraded historical documents. In the comparison, the recognition of the original version of all the images was also included.

Finally, the performance of the proposed OCR, based on image matching, was as well compared to the performance of the commercial OCRs.

For a better overview, a complete table containing the results obtained by both types of images, the CC and the FC, all methods used in the comparison and all the OCRs used for recognition has been compiled. See Table 6.23. In this table, for each type of image, CC and FC, the best performing method for each OCR has been highlighted.

The text in the cells containing the best performing method for a particular OCR are highlighted in that OCR's colour. Then, the cell of the best performing OCR for that particular type of image is filled with orange.

As can be seen in the table, the OCR by ABBYY performed better on the R-FTR images. This was observed in the *high, medium* and *low quality* CC images. In the same way, MODI's results were consistently better with the FTR images. In general, the FTR and the R-FTR were the best performing methods for the CC images based on the recognition results from MODI and ABBYY. The different nature of the FC images resulted in different performance results. The method that best improved ABBYY's recognition rates was actually the method by GPP closely followed by the R-FTR method. MODI's performance was higher when using the images enhanced by the R-FTR method.

Finally, the best performing OCR among the three chosen ones, was the WIM method here proposed. This OCR obtained the best recognition results

for all types of images reaching high recognition rates of 94.35% for the high quality images, and 93.54% for the FC images.

The highest improvement has been experienced in the low quality images. The recognition rates of the original images were as low as 28% and now reach 80% with an improvement of 52%.

In conclusion, the results presented in this chapter show the improvement obtained by the proposed text recovery methods.

Both of them considerably improve the recognition of typewritten documents with different kinds of degradations and of different natures and qualities. To prove it, the recognition results obtained by two well known OCRs were compared to the original images, and three other thresholding methods. In addition, the results obtained by the implemented WIM OCR also outperformed both OCRs by up to 25%.



		<b>ABBYY</b>	<b>MODI</b>	<b>WIM</b>
<b>High Quality CC</b>	<b>Original Images</b>	66.22%	66.34%	n.a.
	<b>FTR Method</b>	80.49%	<b>82.04%</b>	<b>94.35%</b>
	<b>R-FTR Method</b>	<b>88.87%</b>	71.39%	94.10%
	<b>Otsu Method</b>	57.48%	51.48%	n.a.
	<b>SP Method</b>	n.a.	n.a.	n.a.
	<b>GPP Method</b>	62.27%	70.54%	n.a.
<b>Medium Quality CC</b>	<b>Original Images</b>	56.87%	54.81%	n.a.
	<b>FTR Method</b>	76.41%	<b>73.42%</b>	<b>90.67%</b>
	<b>R-FTR Method</b>	<b>84.22%</b>	67.21%	90.18%
	<b>Otsu Method</b>	45.56%	46.36%	n.a.
	<b>SP Method</b>	n.a.	n.a.	n.a.
	<b>GPP Method</b>	42.41%	47.95%	n.a.
<b>Low Quality CC</b>	<b>Original Images</b>	28.20%	42.43%	n.a.
	<b>FTR Method</b>	52.84%	<b>44.51%</b>	80.54%
	<b>R-FTR Method</b>	<b>64.21%</b>	38.47%	<b>80.67%</b>
	<b>Otsu Method</b>	25.05%	14.13%	n.a.
	<b>SP Method</b>	n.a.	n.a.	n.a.
	<b>GPP Method</b>	26.80%	38.47%	n.a.
<b>FC</b>	<b>Original Images</b>	70.15%	69.82%	n.a.
	<b>FTR Method</b>	75.82%	74.75%	93.44%
	<b>R-FTR Method</b>	78.70%	<b>78.06%</b>	<b>93.54%</b>
	<b>Otsu Method</b>	71.72%	65.82%	n.a.
	<b>SP Method</b>	68.34%	68.70%	n.a.
	<b>GPP Method</b>	<b>80.39%</b>	73.38%	n.a.

Table 6.23: Performance comparison of all methods, images and OCRs

# Chapter 7

## Summary and Conclusions

### 7.1 Introduction

In this research, a solution to the recognition problems posed by highly degraded historical typewritten documents was proposed.

The low quality and special characteristics of historical typewritten documents posed new challenges that had been not yet addressed to in the literature. In particular, those created by the difficult carbon copies and over pressed typed characters. As it was discussed in Chapter 2, extensive research has been carried out in the document image analysis field, however a technique that enhances degraded typewritten documents, including carbon copies, was not found.

While there have been several projects and methods specialised on the extraction and enhancement of text from historical documents, as seen in Section 2, most of them were application oriented and could not be successfully applied to the documents considered here.

Standard OCR systems, such as MODI or ABBYY Professional OCRs were found not reliable in these documents. They have not been designed to recognise degraded documents, and do not cope well with the noisy background and broken and touching characters. Their performance on such documents achieved the low recognition rate of 61.9%<sup>1</sup>

---

<sup>1</sup>This rate was obtained from the average of the rates achieved by MODI and ABBYY OCRs in all the original images in the data set used for testing, see more in Chapter 6.

This research's methodology is based in the main features of historical type-written documents as described in Chapter 1. This research proposes a type-written text recovery system that extracts the characters from the image, enhances them individually, repairs possible broken strokes and recognises them. The proposed approach outperformed three well known thresholding methods. The selected methods were Otsu's global thresholding technique [63], Sauvola and Pietaksinen adaptive threshold [76] and Gatos et al. thresholding method for low quality historical documents [31]. See the results of these methods in Sections 6.5.1, 6.5.2 and 6.5.3 respectively.

From the four different categories of images selected from the data set, the greatest recognition improvement on commercial OCR performance was experienced in the *low quality* CC images, with a recognition rate of 64%, reaching 36% improvement, compared to the results obtained by the original images. Medium quality CC images, achieved 84.22% recognition rate, with 27% improvement. The *high quality* CC images achieved 88.87%, with an improvement of 22%. However, for the FC images, the recognition accomplished by the text recovery method here proposed was 1.69% lower than when enhanced by GPP method.

However, the greatest recognition improvement was achieved by the WIM OCR here proposed that reached recognition rates for the high quality CC images of up to 94.35%, 5.5% above the professional OCRs. For medium quality reached 90.67%, with 6.5% of improvement, and 80.67% for the low quality CC images, improving the commercial OCRs by 16.46%. The FC images recognition rate increased by 13.54% reaching 93.54%. The resulting improvement rates were obtained by comparing the best rate to the second best performance method and not to the original image results.

## 7.2 Conclusions about the research

The main objective of this research was to improve the quality of degraded historical typewritten documents to improve recognition by extracting the characters from the background and restore broken and touching characters. The objective of the thesis has been accomplished by the proposed character extraction and recognition methods proposed.

The proposed methods in this research were based on the initial segmentation of each typed character for individual extraction and enhancement, called the Flexible Text Recovery (FTR).

This individual enhancement allowed for a better image thresholding by reducing the background to a minimum, avoiding noise affecting the characters and by adapting to each character's individual characteristics. The resulting characters are free of edge and background noise. This resulted in lower number of errors produced by OCR systems created from noisy pixels, improving the recognition rates.

The combination of two thresholding methods extracted the essence of each character from a localisation box. The AB technique adapts to possible sudden changes that otherwise could lead to broken strokes. However, the adaptive technique was sensitive to the noisy texture in the character image and in some cases merged strokes and created spurious artefacts from noise that altered the character's shape and features.

To counteract that effect, the CT technique was used to confidently extract the character's body and background by selecting a range of the darkest and lightest pixels in the located character box. The remaining uncertain pixels provided key information to the subsequent stage by determining where broken or touching strokes were. This technique has been found more effective in character images with blurry edges since more uncertain pixels remained. This simple but effective method extracted the character's body without noise. Globally applied to each character, this method resulted successful in determining the character's body without any noise or degradation.

The union of these two methods, combining their strengths, created a ternary character image that maintained key broken or touching strokes in greylevel values supplying more information to the subsequent recognition stages. This was found particularly useful as it allowed for flexibility during recognition and improved the recognition results.

As described in the literature, touching and broken characters are considered responsible for most OCR errors. During the image refinement process, broken and wrongly connected strokes were partly restored. This was allowed by analysing the uncertain pixels, where only a small number of greylevel pixels were left in the image. Those pixels are located in conflictive positions, such as touching strokes or broken strokes, that only by recognising the characters could be differentiated. This novel refinement approach creates cleaner

images where uncertain areas in the character are determined and uninformed binarisation is not necessary. The commercial OCR results obtained by the FTR characters improved the recognition results and outperformed a number of thresholding methods, as shown in Chapter 6.

The FTR method refines and repairs the characters shape. Repairing broken characters is one of the most difficult problems in the character image enhancement field. In this research, a method for successfully restoring highly blurred and noisy typewritten characters has been presented.

As well as a text extraction technique, a typewritten character OCR was proposed based on a weighted image matching and aided by a small number of features and a flexible classification tree. The results show a distinct improvement compared to two professional OCR systems and reach up to 94% recognition rates.

Regarding the performance of the commercial OCRs, ABBYYs performance was in overall better than MODI. However, during the recognition tests, ABBYY failed in many occasions to locate the text in the page leaving large amounts of characters unrecognised. In an attempt to allow ABBYY to improve its performance, the layout analysis was performed by hand. MODI on the other hand does not supply information about the extraction of the documents layout nor an option to select it manually and the output was produced automatically.

### 7.3 Limitations and Implications

For completeness, while this research has been successful in achieving its objectives there are some issues related to this application to documents other than those considered here.

The method here proposed for the extraction and recognition of degraded text from historical documents is aimed at greyscale typewritten documents. The FTR method can only be applied to greyscale images, however, binary typewritten document images could be segmented and further recognised by using the character recognition method here proposed.

The FTR method assumes as an input a region containing only text without underlines and without any graphics, logos, or handwritten annotations. Page layout analysis must take place before the document can be recovered and



recognised.

The method here proposed also assumes a document where the main skew is zero degrees. However, as mentioned in Chapter 2, different skew angles may be in the document if the paper is taken out or repositioned in the machine. The method here proposed can operate in the presence of such skew if this is reasonably low, that is, as long as the text lines are not corrupted in the corresponding profile.

The image matching OCR here proposed would have to be modified if a different typewriter font needs to be recognised.

Although these limitations reduce the scope of this method, the performance attained by this type of documents is a major achievement.

## 7.4 Further research

The current FTR method could be improved in a number of ways. A better adaptive thresholding method can be chosen to improve the performance of the AB threshold. As seen during the experimentation, the method by Gatos et al. was the best of the existing thresholding methods. It was less sensitive to noise, and merged less characters strokes wrongly than the chosen SP method. The GPP method could be used instead in the AB threshold. Note that the method by Gatos et al. was published while this research had already progressed further than this stage.

As well, the character stroke refinement could be improved by analysing the uncertain pixels also in the diagonal directions and not only the horizontal and vertical. Like this, key uncertain areas that are located on a curve of a character stroke could also be identified.

Another possible improvement for the intelligent image matching system proposed would consist on adding an extra step in the recognition. As it was discussed in Chapter 6, there are a number of misrecognised pairs of characters that are repeated frequently, e.g. 'e', 'c', 'a', 's', etc. A number of rules can be created to distinguish further such pairs of characters based on other features such as zoning. The rules could point out a reduced number of comparisons needed to differentiate the two characters using zoning features. For example, to differentiate character 'c' from character 'e' the zone in the centre stroke could be evaluated. The rules could be applied to all the characters or mainly

to the most problematic ones, such as those without ascenders or descenders.

The WIM OCR here proposed could be extended to recognise different typewritten fonts. Character prototypes for the new fonts would need to be extracted and added to the classifier. To extend the WIM system to recognise additional typewritten fonts without increasing the computational needs substantially, two possibilities arise. The first possibility would be to ask the user to select the font to be recognised before the recognition. The second possibility would be based on finding the similarities and dissimilarities between the two fonts. Many typewritten fonts share a number of characters while only few are remarkably different. In this way, only the different character prototypes would be added to the classifier. This second possibility would not differentiate between the different typewriter fonts but would not need to ask for information to the user. The WIM system could also be modified to recognise a non-typewriter font as long as this is monospace.

And finally, the recognition achieved by the WIM system could be improved by using the recognition confidence studied in Chapter 5. As discussed in Chapter 5, the second best recognised character by the WIM recognition system was in 89% of the cases the correct match. The recognition confidence algorithm used would correctly determine when a character was misrecognised in 69.19% of the cases. In conclusion, 89% of the 69.19% of misrecognised characters that were flagged as low confidence could be corrected improving the recognition further.

# Bibliography

- [1] A. S. Abutableb. Automatic thresholding of gray-level pictures using two-dimensional entropy. *Computing Vision Graphics and Image Process.*, 47(1):22–32, 1989.
- [2] B. Allier and H. Emptoz. Degraded character image restoration using active contours: A first approach. In *Symposium on Document Engineering (DocEng2002)*, pages 142–148, Washington, 2002.
- [3] A. Antonacopoulos. Introduction to document image analysis. 1995.
- [4] A. Antonacopoulos. Local skew angle estimation from background space in text regions. In *International Conference on Document Analysis and Recognition (ICDAR'97)*, pages 684–688, 1997.
- [5] A. Antonacopoulos and D. Karatzas. Semantics-based content extraction in typewritten historical documents. In *International Conference on Document Image Analysis and Recognition*, volume 1, pages 48–53, 2005.
- [6] A. Antonacopoulos, D. Karatzas, H. Krawczyk, and B. Wiszniewski. The lifecycle of a digital historical document: Structure and content. In *Document Engineering (DocEng'04)*, pages 147–154, 2004.
- [7] H. S. Baird. High-performance ocr preclassification trees, 1995.
- [8] H. S. Baird. The skew angle of printed documents. pages 204–208, 1995.
- [9] J. Bernsen. Dynamic thresholding of gray level images. In *International Conference on Pattern Recognition (ICPR'86)*, page 1251–1255, 1986.
- [10] L. Blando, J. Kanai, and T. A. Nartker. Prediction of ocr accuracy using simple image features. In IEEE, editor, *International Conference on Document Analysis and Recognition (ICDAR'95)*, pages 319–322, 1995.

- [11] M. Bohner. An automatic measurement devise for the evaluation of the print quality of printed character. *Pattern Recognition*, 9:11–19, 1977.
- [12] M. Bokser. Omnidocument technologies. *Proceedings of the IEEE*, 80(7), 1992.
- [13] M. Cannon, M. Fugate, D. Hush, and C. Scovel. Selecting a restoration technique to minimize ocr error. Technical LA-UR-01-6860, Los Alamos National Laboratory, USA, 24 January 2003.
- [14] M. Cannon, J. Hochberg, and P. Kelly. A remarkably effective method for increasing the ocr accuracy of degraded typewritten documents. In *Symposium on Document Image Understanding Technology*, Annapolis, Maryland, 1999.
- [15] M. Cannon, P. Kelly, S. Sitharama Iyengar, and N. Brener. An automated system for numerically rating document image quality. In *SPIE*, volume 3027, pages 161–167, 1997.
- [16] R. G. Casey and E. Lecolinet. A survey of methods and strategies in character segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):690–706, 1996.
- [17] B. Chanda and D. Majumder. A note on the use of grey level co-occurrence matrix in threshold selection. *Signal Processing*, 15:149–167, 1988.
- [18] C. Chen and J. DeCurtis. Word recognition in a segmentation-free approach to ocr. In IEEE, editor, *International Conference on Document Analysis and Recognition (ICDAR'93)*, pages 573–576, 1993.
- [19] Y. Chen and G. Leedham. Decompose algorithm for thresholding degraded historical document images. *Visual Image and Signal Processing*, 152(6):702–714, 2005.
- [20] A. Dawoud and M. Kamel. Iterative model-based binarization algorithm for cheque images. *International Journal on Document Analysis and Recognition*, 5:28–38, 2002.
- [21] R. O. Duda, P. E. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition, 2001.

- [22] O. Due Trier, A. Jain, and T. Taxt. Feature extraction methods for character recognition—a survey. *Pattern Recognition*, 29(4):641–662, 1996.
- [23] O. Due Trier and T. Taxt. Evaluation of binarization methods for document images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(3):312–315, 1995.
- [24] L. Eikvil. Optical character recognition. Unpublished, 1993.
- [25] M. C. F. O’Gorman. Finding picture edges through collinearity of feature points. *IEEE Transaction on Computers*, 25(4):449–456, 1976.
- [26] T. A. N. G. Nagy and S. V. Rice. Optical character recognition: An illustrated guide to the frontier. In *Proceedings of SPIE: Document Recognition and Retrieval VII*, volume 3967, 2000.
- [27] U. Garain, T. Paquet, and H. L. On foreground - background separation in low quality document images. *International Journal of Document Analysis*, 8(1):47–63, 2006.
- [28] B. Gatos, T. Konidakis, K. Ntzios, I. Pratikakis, and S. J. Perantonis. A segmentation-free approach for keyword search in historical typewritten documents. In *Proceedings of the 8th International Conference on Document Analysis and Recognition (ICDAR’05)*, pages 54–58, Washington, DC, USA, 2005. IEEE Computer Society.
- [29] B. Gatos, N. Papamarkos, and C. Chazmas. A binary-tree-based ocr techniques for machine-printed characters. *Engineering Applications of Artificial Intelligence*, 10(4):403–412, 1997.
- [30] B. Gatos, N. Papamarkos, and C. Chazmas. Using curvature features in a multiclassifier ocr system. *Engineering Applications of Artificial Intelligence*, 10(2):213–224, 1997.
- [31] B. Gatos, I. Pratikakis, and S. Perantonis. An adaptive binarization technique for low quality historical documents. In *Proceedings of the 6th IAPR Workshop on Document Analysis System (DAS 2004)*, volume 3163, pages 102–113, 2004.
- [32] R. Gonzalez and R. Woods. *Digital Image Processing*. Prentice Hall, New Jersey, 2nd edition, 2002.



- [33] J. C. Handley. Improving ocr through combination: A survey. In *IEEE*, 1998.
- [34] R. Hartley and K. Crumpton. Quality of ocr for degraded text images, 1999.
- [35] J. He, Q. D. M. Do, A. C. Downton, and J. H. Kim. A comparison of binarization methods for historical archive documents. In *International Conference on Document Analysis and Recognition (ICDAR'05)*, pages 538–542, 2005.
- [36] J. He and A. C. Downton. Evaluation of a user-assisted archive construction system for online natural history archives. In *International Conference on Document Analysis and Recognition (ICDAR'05)*, pages 442–446, 2005.
- [37] J. D. Hobby and H. S. Baird. Degraded character image restoration. In *Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'96)*, Las Vegas, Nevada, 1996.
- [38] J. D. Hobby and T. Ho. Enhancing degraded document images via bitmap clustering and averaging. In *International Conference on Document Analysis and Recognition (ICDAR'97)*, page 394, Ulm, Germany, 1997.
- [39] C. V. Jawahar, P. K. Biswas, and A. K. Ray. Investigations on fuzzy thresholding based on fuzzy clustering. *Pattern Recognition*, 30(10):1605–1613, 1997.
- [40] D. Jung, M. Krishnamoorthy, G. Nagy, and A. Shapira. N-tuple features for ocr revisited. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):734–745, 1996.
- [41] D. Jung and G. Nagy. Joint feature and classifier design for ocr. In *Third International Conference on Document Analysis and Recognition (ICDAR'95)*, volume 2, page 1115, 1995.
- [42] M. Kamel and A. Zhao. Extraction of binary character/graphics images from grayscale document images. *Graphical Models and Image Processing*, 55(3):203–217, 1993.

- [43] J. Kanai, T. A. Nartker, and F. Jenkins. Using ideal images to establish a baseline of ocr. Annual report, Information Science Research Institute (ISRI), 1993.
- [44] R. Kasturi, L. O'Gorman, and V. Govindaraju. Document image analysis: A primer. *Sadhana*, 27(1):3–22, 2002.
- [45] E. Kavallieratou and H. Antonopoulou. Cleaning and enhancing historical documents. In *Advanced Concepts for Intelligent Vision Systems (ACIVS 2005)*, pages 681–688, 2005.
- [46] R. Kirby and A. Rosenfeld. A note on the use of (gray level, average gray level) space as an aid in threshold selection. *IEEE Transactions On Systems, Man and Cybernetics*, 9:860–864, 1979.
- [47] J. Kittler and J. Illingworth. Minimum error thresholding. *Pattern Recognition*, 19(1):41–47, 1986.
- [48] H. A. J. Koskinen, L. Huttunen. Text enhancement method based on soft morphological filters. In *SPIE*, volume 2181, pages 243–253, San Jose, CA, USA, 1994. Document Recognition.
- [49] T. T. L. Eikvil and K. Moen. A fast adaptive method for binarization of document images. In *The 1st International Conference on Document Analysis and Recognition (ICDAR'91)*, pages 435–443, 1991.
- [50] S.-W. Lee and Y. Kim. Direct extraction of topographic features for grey scale character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):724–729, 1995.
- [51] G. Leedham, S. Varma, A. Patankar, and V. Govindaraju. Separating text and background in degraded document images - a comparison of global thresholding techniques for multi-stage thresholding. In *Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02)*, page 244, Ontario, Canada., 2002.
- [52] G. Leedham, C. Yan, K. Takru, J. Hadi Nata Tan, and L. Mian. Comparison of some thresholding algorithms for text/background segmentation in difficult document images. In I. C. Society, editor, *Seventh International*

- Conference on Document Image Analysis and Recognition (ICDAR'03)*, volume 2, page 859, 2003.
- [53] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Doklady akademii nauk, SSSR*, 1965.
- [54] M. P. I. Liang, J. Haralick. Document image restoration using binary morphological filters. In *SPIE*, volume 2660, pages 274–285, San Jose, CA, USA, 1996. Document Recognition III.
- [55] Y. Liu and S. N. Srihari. Document image binarization based on texture features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):540–544, 1997.
- [56] R. Loce and E. Dougherty. Restoration by increasing binary filters. In S. Press, editor, *Enhancement and Restoration of Digital Documents: Statistical Design of Nonlinear Algorithms*, pages 189–200. SPIE, 1997.
- [57] S. Mori, C. Suen, and K. Yamamoto. Historical review of ocr research and development. *Proceedings of the IEEE*, 80(7):1029–1058, 1992.
- [58] G. Nagy. At the frontiers of ocr. In *Proceedings of the IEEE*, volume 80, pages 1093–1100, 1992.
- [59] W. Niblack. *An Introduction to Digital Image Precessing*. Prentice-Hall, 1986.
- [60] L. O’Gorman. Binarization and multithresholding document images using connectivity. *Graphical Models and Image Processing*, 56(6):494–506, 1994.
- [61] L. O’Gorman and R. Kasturi. *Document Image Analysis*. IEEE Computer Society, 1996.
- [62] M. Oguro, T. Akiyama, and K. Ogura. Faxed document image restoration using gray level representation. In *Fourth International Conference on Document Analysis and Recognition (ICDAR'97)*, pages 679–683, Ulm, Germany, 1997.
- [63] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions On Systems, Man and Cybernetics*, 9:62–66, 1979.

- [64] Z. P. and C. L. A novel feature extraction method and hybrid tree classification for handwritten numeral recognition. *Pattern Recogn. Lett.*, 23(1-3):45-56, 2002.
- [65] S. K. Pal and A. Rosenfeld. Image enhancement and thresholding by optimization of fuzzy compactness. *Pattern Recognition Letters*, 7(2):77-86, 1988.
- [66] N. Papamarkos. A technique for fuzzy document binarization. In *Document Engineering (DocEng'01)*, pages 152-156, 2001.
- [67] S. Perantoni, B. Gatos, K. Ntzios, I. Pratikakis, I. Vrettaros, A.S.Drigas, C. Emmanouilidis, A. Kesidis, and D. Kalomoirakis. System for processing and recognition of old greek manuscripts (the d-scribe project). In *4th WSEAS International Conference on Applied Informatics and Communications*, volume 3, pages 2049-2057, 2004.
- [68] S. Perantonis, B. Gatos, I. Pratikakis, A. Drigas, C. Emmanouilidis, and A. Kesidis. D-scribe project: A system for digitization and processing of greek manuscripts. *International Journal Information Theory and Applications*, 11:232-240.
- [69] S. Pletschacher. Ocr alternatives for electronic publishing of digitised documents. In *Proceedings of the 9th ICC International Conference on Electronic Publishing*, Katholieke Universiteit Leuven, Leuven-Heverlee(Belgium), 2005.
- [70] S. Rasoul and D. Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21, 1991.
- [71] T. Rath and R. Manmatha. Features for word spotting in historical manuscripts. In *The Seventh International Conference on Document Image Analysis and Recognition (ICDAR'03)*, volume 1, page 218, 2003.
- [72] S. Rice. The fifth annual test of ocr accuracy. In *SDAIR96*, page XX, 1996.
- [73] C. S. Ridler TW. Picture thresholding using an iterative selection method. *IEEE Transactions On Systems, Man and Cybernetics*, 8(8):630-632, 1978.
- [74] C. Rodriguez, J. Mugerza, M. Navarro, A. Zarate, J. Martin, and J. Perez. A two-stage classifier for broken and blurred digits in forms. In

- Fourteenth International Conference on Pattern Recognition (ICPR '98)*, pages 1101–1105, 1998.
- [75] A. Rosenfeld and P. De La Torre. Histogram concavity analysis as an aid to threshold selection. *IEEE Transactions on Systems, Man and Cybernetics*, 13:231–235, 1983.
- [76] J. Sauvola and M. Pietaksinen. Adaptive document image binarization. *Pattern Recognition*, 33:225–236, 2000.
- [77] M. Seeger and C. Dance. Binarising camera images for ocr. In *International Conference on Document Analysis and Recognition (ICDAR'01)*, pages 54–58, 2001.
- [78] M. I. Sezan. A peak detection algorithm and its application to histogram-based image data reduction. *Computer Vision Graphics Image Processing*, 49(1):36–51, 1990.
- [79] M. Sezgin and B. Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging*, 13(1):146–165, 2004.
- [80] Z. Shi and V. Govindaraju. Character image enhancement by selective region-growing. *Pattern Recognition Letters*, 17:523–527, 1996.
- [81] Z. Shi and V. Govindaraju. Historical document image enhancement using background light intensity normalization. In *The 17th International Conference on Pattern Recognition (ICPR'04)*, volume 1, pages 473–476, 2004.
- [82] Y. Solihin and G. Leedham. Integral ratio: A new class of global thresholding techniques for handwriting images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):761–768, 1999.
- [83] R. J. Spinrad. Machine recognition of hand printing. *Information and Control*, 8(2):124–142, 1965.
- [84] A. L. Spitz. Correcting for variable skew in document images. *International Journal on Document Analysis and Recognition (IJ DAR)*, 6(3):181–189, 2003.



- [85] P. Stubberud, J. Kanai, and V. Kalluri. Adaptive character image enhancement to improve ocr accuracy. In *10th International Conference on Systems Engineering*, pages 1093–2000, Coventry, UK, 1994.
- [86] P. Stubberud, J. Kanai, and V. Kalluri. Adaptive image restoration of text images that contain touching or broken characters. In *3rd International Conference on Document Image and Recognition (ICDAR'95)*, volume 2, pages 778–, 1995.
- [87] T. N. S.V. Rice, G. Nagy. *Optical Character Recognition: An illustrated guide to the frontier*. Kluwer Academic Publishers, 1999.
- [88] P. D. Thouin and C. I. Chang. A method for restoration of low-resolution document images. *International Journal on Document Analysis and Recognition (IJ DAR)*, 2(4):200–210, 1999.
- [89] W. Throssell and P. Fryer. The measurement of print quality for optical character recognition systems. *Pattern Recognition*, 6:141–147, 1974.
- [90] A. Tonazzini, S. Vezzosi, and L. Bedini. Analysis and recognition of highly degraded printed characters. *International Journal on Document Analysis and Recognition, (IJ DAR)*, 6:236–247, 2004.
- [91] O. Trier, A. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, pages 641–662.
- [92] W.-H. Tsai. Moment preserving thresholding: A new approach. *Computer Vision, Graphics and Image Processing*, 29:377–393, 1985.
- [93] S. Tsujimoto and H. Asada. Resolving ambiguity in segmenting touching characters. In *First International Conference On Document Analysis and Recognition (ICDAR'91)*, pages 701–709, Saint-Malo, France, 1991.
- [94] S. Venkatesh and P. L. Rosin. Dynamic threshold determination by local and global edge evaluation. *Graphic Models and Image Processing*, 57(2):146–160, 1995.
- [95] S. Vezzosi, L. Bedini, and A. Tonazzini. An integrated system for the analysis and the recognition of characteres in ancient documents. In *Document Analysis Systems V : 5th International Workshop, DAS 2002*, Princeton, NY, USA, 2002.

- [96] L. Wang and T. Pavlidis. Direct gray-scale extraction of features for character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1053–1067, 1993.
- [97] J. Weszka and A. Rosenfeld. Threshold evaluation techniques. *IEEE Transactions On Systems, Man and Cybernetics*, 8:622–629, 1978.
- [98] J. M. White and G. D. Rohrer. Image thresholding for optical character recognition and other applications requiring character image extraction. *IBM Journal of Research and Development*, 27(4):400, 1983.
- [99] V. Wu and R. Manmatha. Document image clean-up and binarization. Technical report, Computer Science Department, University of Massachusetts, December 1997.
- [100] C. Yan and G. Leedham. The multistage approach to information extraction in degraded documents images. In *The 17th International Conference on Pattern Recognition (ICPR'04)*, volume 1, pages 445–448, 2004.
- [101] Y. Yang, K. Summers, and M. Turner. A text image enhancement system based on segmentation and classification methods. In 1st ACM workshop on Hardcopy document, editor, *Conference on Information and Knowledge Management*, pages 33–40, 2004.
- [102] Y. Yang and H. Yan. An adaptive logical method for binarization of degraded document images. *Pattern Recognition*, 33:787–807, 2000.
- [103] S. D. Yanowitz and A. M. Bruckstein. A new method for image segmentation. *Computer Vision Graphics and Image Processing*, 46(1):82–95, 1989.
- [104] Q. Zheng and T. Kanungo. Morphological degradation models and their use in document image restoration. Technical report, Language and Media Processing Laboratory, 2001.
- [105] H. Zhu, F. Chan, and F. Lam. Image contrast enhancement by constrained local histogram equalization. *Computer Vision and Image Understanding*, 73(2):281–290, 1999.

## List of Tables

4.1	Results improvement between the Original image and the FTR image . . . . .	105
5.1	Recognition results obtained based on the confidence measure.	121
5.2	Recognition rates of the FTR images by the matching method proposed. . . . .	138
5.3	Recognition rates of the FTR images by the matching method proposed. . . . .	138
5.4	Recognition rates of the R-FTR images compared to the original images and the FTR images. . . . .	140
6.1	Data set categories . . . . .	145
6.2	Recognition rates of the original images. . . . .	148
6.3	Percentages of missing, noise and incorrect characters in the original images . . . . .	148
6.4	Results improvement between the recognition results of the original CC images and the FTR CC images . . . . .	151
6.5	Percentages of noise, missing and incorrect characters of FTR images . . . . .	151
6.6	Results improvement of the recognition rates between the R-FTR CC images and the FTR CC images . . . . .	153
6.7	Percentages of noise, missing and incorrect characters of R-FTR images . . . . .	154
6.8	Results improvement between the OTSU image and the FTR images . . . . .	156
6.9	Percentages of noise, missing and incorrect characters of Otsu .	157
6.10	Results improvement between the CC OTSU images and the R-FTR images . . . . .	157

---

6.11	.....	158
6.12	Results improvement between FC SP images and R-FTR images	159
6.13	Results improvement between GPP images and FTR CC images	159
6.14	Percentages of missing, noise and incorrect characters of GPP	160
6.15	Results improvement between GPP images and R-FTR CC images	160
6.16	Recognition rates of FTR images by WIM	162
6.17	Results improvement between WIM and commercial OCRs	162
6.18	Percentages of missing, noise and incorrect characters obtained by WIM the FTR images	163
6.19	Recognition rates of the R-FTR images by WIM	164
6.20	Results improvement between FTR WIM images and the com- mercial OCRs	164
6.21	Percentages of missing, noise and incorrect characters of the WIM R-FTR images	165
6.22	Most common template matching errors	165
6.23	Performance comparison of all methods, images and OCRs	168

# List of Figures

1.1	Document Image Analysis Diagram . . . . .	11
2.1	Image examples on the results of the method by Yang and Yan [102]	30
2.2	Three pixel classes in the approach by Solihin and Leedham [82]	32
2.3	Block diagram of the method proposed by Gatos et al. [31] . . .	34
2.4	Schematic diagram of the approach by Garain et al. [27] . . . . .	34
2.5	Sample image from a Greek manuscript from the D_SCRIBE project [67] . . . . .	44
2.6	Integrated System implemented by Tonazzini et al [90, 95] . . .	44
2.7	A sample image form the documents used by Tonazzini et al [90, 95] . . . . .	45
2.8	Digital historical document life-cycle [5, 6]. . . . .	46
2.9	Image samples from the MEMORIAL project. . . . .	47
2.10	Results of applying the different processing methods to different region levels. [5, 6] . . . . .	47
3.1	Template matching example . . . . .	52
3.2	OCR-A font . . . . .	53
3.3	Illustration of chain coding and a coding rule. . . . .	58
3.4	Contingency table . . . . .	62
3.5	Illustration of the limitation of skeletonisation [12] . . . . .	68
3.6	Illustration of the limitations found by topological and geometric features [12] . . . . .	69
3.7	Example of a text-like block non-suitable for conventional OCR [69]	71
4.1	Outline of the proposed method . . . . .	74
4.2	Sample image we want to segment . . . . .	75



4.3	Vertical and Horizontal Projection Profiles of the image in Figure 4.2 . . . . .	76
4.4	Degraded image and its horizontal projection affected by fade characters . . . . .	76
4.5	Sigmoid curve used for the stretching. $\sigma = 1.2$ . . . . .	78
4.6	Image with its greylevel histograms and vertical projection profiles before and after histogram transformation . . . . .	78
4.7	Comparison of horizontal profiles before and after histogram transformation . . . . .	79
4.8	Vertical projection of image in Figure 4.2 with segmentation points.	80
4.9	A segmented line and its horizontal profile. . . . .	81
4.10	A segmented line and its horizontal profile after pre-processing.	81
4.11	Image in Figure 4.9(a) with segmentation points. . . . .	82
4.12	Image with fixed segmentation points . . . . .	83
4.13	Image with fixed segmentation points flexibly adjusted . . . . .	84
4.14	Examples of improved segmentation. . . . .	84
4.15	Two segmented areas . . . . .	85
4.16	Histogram Stretched image 'a' . . . . .	86
4.17	Stretched image . . . . .	86
4.18	Localised images . . . . .	87
4.19	Localised character image . . . . .	88
4.20	Greyscale histogram of the Segmentation Foreground and Background. . . . .	88
4.21	Histograms of <i>Localised foreground</i> and background regions . . .	89
4.22	Histograms of several characters representing foreground and background . . . . .	90
4.23	Localised character 'i' with intensity histograms of its Localised Foreground and Complemented Background . . . . .	90
4.24	Localised character 'i' with intensity histograms of its Localised Foreground after background threshold . . . . .	91
4.25	Histograms of several characters after background thresholding	91
4.26	Text line after CT. . . . .	92
4.27	Localised character 'i' with intensity histograms of its Localised Foreground after CT . . . . .	92
4.28	Text line after SP threshold. . . . .	93

4.29	Text line after SP's threshold was applied to each Localised Character. . . . .	94
4.30	Text line after SP threshold was applied to each <i>Localised Character</i> after being smoothed. . . . .	94
4.31	Several character images after SP threshold . . . . .	95
4.32	Combination table rules. . . . .	95
4.33	Combination threshold illustration. . . . .	96
4.34	Resulting image after combination of methods CT and AB . . .	96
4.35	Flexibly recovered sample characters . . . . .	97
4.36	Uncertain pixels selected as part of the stroke . . . . .	98
4.37	Algorithm to locate pixels that belong to the UEA . . . . .	99
4.38	Uncertain pixels selected as part of the background . . . . .	99
4.39	Algorithm to locate uncertain pixels that belong to the background	99
4.40	Algorithm to locate uncertain pixels that belong to the background	100
4.41	Uncertain pixels that belong, either to the character stroke, i.e. broken character, or are noise, i.e. like a wrongly connected stroke.	100
4.42	Recovered sample characters 'a' and 's'. . . . .	101
4.43	Characters 'a' and 's' with uncertain areas located. . . . .	101
4.44	Characters 'a' and 's' with uncertain areas located and improved.	102
4.45	Characters 'a' and 's' with UNA pixels set to white. . . . .	102
4.46	Characters 'S' and 'g' showing a broken corner misclassified as UNA. . . . .	103
4.47	Characters 'S' and 'g' showing a broken corner misclassified as UNA. . . . .	103
4.48	Characters 'a' and 's' with UEA pixels set to black. . . . .	104
4.49	Possible UKA combinations for characters 'a' and 's'. . . . .	104
4.50	Extracted and refined resulting characters . . . . .	106
5.1	Extracted Character Images . . . . .	109
5.2	Examples of frequently degraded character images . . . . .	110
5.3	Character images used as prototypes . . . . .	111
5.4	Example character images to be matched . . . . .	114
5.5	Differences with between character images . . . . .	114
5.6	Differences with neighbours weights between character images	115
5.7	Differences with distance weights between character images . .	116
5.8	The character '2' from two different typewriter fonts. . . . .	117

5.9	The character '4' from two different typewriter fonts. . . . .	117
5.10	Plot of the WIM score for three random well matched characters.	118
5.11	Plot of the recognition rates for three random mis-matched characters. . . . .	118
5.12	Plot of the recognition distances between three well matched characters. . . . .	120
5.13	Plot of the recognition distances between three mis-matched characters. . . . .	120
5.14	Recognised image with confidence indicators. The blue colour indicates above threshold confidence, the red colour indicates below threshold confidence . . . . .	121
5.15	Character size clusters . . . . .	123
5.16	Character size clusters classified . . . . .	124
5.17	Characters classified by size. . . . .	124
5.18	Low characters classified (in orange) . . . . .	125
5.19	Characters in the <i>left-weight</i> class . . . . .	127
5.20	Characters in the <i>right-weight</i> class . . . . .	127
5.21	Characters in the <i>center-weight</i> class . . . . .	127
5.22	Characters in the <i>double-weight</i> class . . . . .	128
5.23	Character 'a' projections maxima with different degradations. . . . .	129
5.24	Character 'e' projections maxima with different degradations. . . . .	129
5.25	Extracted zones of a <i>tall</i> character. . . . .	130
5.26	Extracted zones of a <i>small</i> character. . . . .	130
5.27	Two characters with different zones. . . . .	131
5.28	Tree classified by size into <i>Short, Tall, Low</i> and <i>Punctuation</i> . . . . .	133
5.29	Tall characters classification tree . . . . .	133
5.30	Low characters classification tree . . . . .	134
5.31	Short characters classification tree . . . . .	135
5.32	Punctuation characters classification tree . . . . .	135
5.33	Tall characters classification tree created from dataset . . . . .	136
5.34	Short characters classification tree created from dataset . . . . .	137
5.35	A character with its uncertain areas set in orange . . . . .	139
5.36	The four candidates obtained from the combination of the uncertain areas in Figure 5.35 . . . . .	139
5.37	Recognition rates of the restored images by WIM compared to the results obtained by the two OCRs ABBYY and MODI. . . . .	141

---

6.1	Images categories used in the results statistics . . . . .	144
6.2	Sample image of a word difficult to recognise . . . . .	146
6.3	Example of Levenshtein distance of two strings . . . . .	147
6.4	Original CC image sample with recognition results of MODI and ABBYY OCRs . . . . .	149
6.5	Original image sample with recognition results of MODI and ABBYY OCRs. . . . .	149
6.6	Original FC image sample with recognition results of MODI and ABBYY OCRs . . . . .	150
6.7	FTR CC image sample with recognition results of MODI and ABBYY OCRs. . . . .	152
6.8	FTR FC image sample with recognition results of MODI and AB- BYY OCRs. . . . .	153
6.9	R-FTR CC image sample with recognition results . . . . .	155
6.10	R-FTR FC image sample with recognition results . . . . .	155
6.11	Otsu CC image sample with recognition results . . . . .	156
6.12	Otsu FC image sample with recognition results . . . . .	157
6.13	Original CC image sample with its SP and smoothed SP outputs	158
6.14	SP FC image sample with recognition results . . . . .	159
6.15	GPP image sample with recognition results . . . . .	161
6.16	GPP FC image sample with recognition results . . . . .	161

# Index

- Aggressive binarisation (AB), 93
- Carbon Copy (CC), 17, 144
- Cautious Ternarisation (CT), 87
- Character Classification, 132
- Character Localisation, 85
- Character Position Validation, 84
- Character Segmentation, 80
- Character Size Feature, 123
- Character Weight Feature, 126
- Combination Thresholding, 86
- Document Image Analysis (DIA), 21
- Feature Extraction, 54, 122
- File Cards (FC), 144
- Fixed-grid segmentation, 82
- Flexible Stroke Refinement, 97
- Flexible Text Recovery (FTR), 72
- Gatos et. al Method (GPP), 33
- Histogram Matching, 38
- Histogram Transformation, 77
- Matching Score, 113
- MEMORIAL Project, 45
- Niblackt's Method, 27
- Number of Horizontal Maxima Feature,  
128
- Number of Vertical Maxima Feature,  
128
- Optical Character Recognition(OCR),  
51
- Otsu's Method, 25
- Pre-Processing, 21
- Projection Profile, 54, 74
- Recognition Confidence, 117
- Restored Flexible Text Recovery (R-FTR),  
138
- Sauvola and Pietaksinen's Method (SP),  
27
- Skew Correction, 22
- Template Matching, 51, 55
- Text Lines Segmentation, 79
- The Data Set, 143
- Thresholding, 24
- Uncertain Edge Area (UEA), 98
- Uncertain Key Area (UKA), 99
- Uncertain Noise Area (UNA), 98
- Weighted Image Matching (WIM), 110