

Robust and applicable handwriting biometrics

Axel Brink



Cover: Medieval handwriting in a charter, a legal administrative document (1309). Zeeuws Archief, Onze Lieve Vrouwe abdij te Middelburg, access number 27, item number 80, regist number 119. Photo: Jinna Smit. Design: Marije van der Hoest.

Printed by: Ipskamp Drukkers

ISBN: 978-90-367-5139-1 (printed) / 978-90-367-5174-2 (electronic)

This research was made possible thanks to NWO grant 634.000.434 (ToKeN/TriGraph).

RIJKSUNIVERSITEIT GRONINGEN

Robust and applicable handwriting biometrics

Proefschrift

ter verkrijging van het doctoraat in de
Wiskunde en Natuurwetenschappen
aan de Rijksuniversiteit Groningen
op gezag van de
Rector Magnificus, dr. E. Sterken,
in het openbaar te verdedigen op
vrijdag 2 december 2011
om 11.00 uur

door

Axel Arjan Brink

geboren op 28 juli 1979
te Dalflen

Promotor: Prof.dr. L.R.B. Schomaker

Beoordelingscommissie: Prof.dr. J.W.J. Burgers
Prof.dr. V. Govindaraju
Prof.dr. T. Paquet

Contents

Preface	ix
1 Introduction	1
1.1 Objectives of this dissertation	4
1.2 Overview	5
1.3 Individuality in handwriting	6
1.4 Handwriting biometrics	8
1.5 Digitization	9
1.6 Preprocessing	9
1.7 Segmentation	10
1.8 Feature extraction	10
1.9 Pairwise comparison	16
1.10 Classification for writer verification	18
1.11 Classification for writer identification	21
1.12 Performance evaluation	21
I Robustness analysis of handwriting biometrics	25
2 Robustness for text scarcity	29
2.1 Method	30
2.2 Results	33
2.3 Conclusion	34

3	Robustness for crossed-out words	37
3.1	Recognizing crossed-out words	38
3.2	Application to writer verification	43
3.3	Application to writer identification	47
3.4	Conclusion	49
4	Robustness for disguise by slant manipulation	51
4.1	TriGraph Slant Dataset	53
4.2	Slant estimation	55
4.3	Feature extraction and comparison	57
4.4	Experiment 1: information in slant	58
4.5	Experiment 2: is deliberate slant change an affine transform?	62
4.6	Conclusion	69
II	New methods for robust and applicable handwriting biometrics	71
5	Using ink width and directionality as a feature	75
5.1	Datasets	76
5.2	Analysis of trace-width production	79
5.3	Quill feature	84
5.4	Performance experiment	92
5.5	Results	95
5.6	Future work	97
5.7	Conclusion	99
6	Increasing explainability using vantage writers	101
6.1	Method	102
6.2	Results	105
6.3	Conclusion	108
7	Conclusions	111
7.1	Robustness analysis	111
7.2	New methods for robust and applicable handwriting biometrics	112
7.3	Future work	113

A GIWIS: a robust and applicable writer identification tool	117
A.1 Design principles	118
A.2 Functionality	119
A.3 Architecture	123
A.4 Discussion	123
Bibliography	125
Samenvatting (summary in Dutch)	137
Author publications	141
Curriculum vitae	143
Index	145

Preface

This is a dissertation about handwriting biometrics: automated attribution of handwriting to writers. This is done using writer-specific characteristics that are derived computationally, without regarding the textual contents. To be clear, the dissertation is not about computers *reading* handwriting. This is the domain of *handwriting recognition*, which is related but the goal is opposite: the objective is to transcribe the textual contents of the handwriting to ‘typed’ text, disregarding individual handwriting characteristics. The dissertation is not about *graphology* either: it does not attempt to derive a writer’s personality based on the handwriting. Instead, it will be investigated to what extent algorithms for handwriting biometrics are robust, and new techniques are introduced to increase both robustness and applicability of such systems. Before moving to the dissertation itself, I would like to spend a few words on how it was made.

The dissertation was written between 2005 and 2011. During 2005 – 2009, it was part of a full-time job at the ALICE department of the University of Groningen. Originally, the goal was to analyze contemporary (forensic) handwriting, but in the process the focus broadened to include historical handwriting as well. An important factor in the writing process was a sofa in our room: it facilitated a thinking position and it encouraged discussions with colleagues. I was surrounded by great colleagues and the job allowed me to visit Germany, France, Florida, and Brazil. But after the four-year contract period, I got a commercial job and the dissertation had to be finished in the evening hours. Writing pace decreased, although I learned that temporarily moving to new environments (in my case, outdoor cafés) can boost the process. And now it is finally done.

Many people have contributed to this thesis in a direct or indirect way, and I wish to express my gratitude to them. Most notably, the dissertation could not exist without prof. Lambert Schomaker, my promotor, supervisor, and guide in the world of science. He shared his broad experience on handwriting analysis with me, he knew what ingredients were needed to lift papers to scientific standards, and he convinced me to continue when it was needed.

Marius Bulacu set an example in techniques for handwriting biometrics, and he was

also eager to share his knowledge and ideas with me. His crystal-clear standpoints were always insightful as well as entertaining.

I have had a very nice cooperation with Jinna Smit, Mark Aussems and Judith Nobels in experiments on historical handwriting. They have become advocates of using biometrics with historical handwriting and their enthusiasm has been contagious. Especially Jinna had a big impact on the dissertation by opening the door to this new research direction.

Elisa van den Heuvel, partner in the TriGraph research project, kindly introduced me to the forensic way of handwriting analysis. We collaborated in an investigation on slant together with enthusiastic student Roeland van Batenburg and Ralph Niels, who was also a partner in the TriGraph research project. Ralph and I had good discussions and made many plans, and I am lucky to be a colleague of him once again.

Anja Lobanova, Bea Valkenier, Dirkjan Krijnders, Marco Wiering, and Martijn Dijkstra helped me writing the thesis by providing helpful comments and stimulating to continue. Tijn van der Zant pushed me to think ‘out of the box’ and convinced me to buy the sofa.

The colleagues that were not mentioned above contributed in other ways, at least everyone took part in creating a great atmosphere at the ALICE department: Arnold, Bart, Bart, Ben, Ben, Chris, Edith, Elina, Elske, Esther, Fokie, Geertje, Gert (my frisbee master), Hanneke, Hedde (my Go master), Hedderik, Ingrid, Jacolien, Jacomien, Jelmer, Jennifer, Joep, Jolie, Karin, Leendert, Liesbeth, Margriet, Mariëtte, Maria, Nancy, Niels (my boardgames master), Renante, Rineke, Ronald, Ronald, Roy, Rutger, Sietse, Sietse, Sjoerd, Sonja, Sujata, Tjeerd, and Wouter.

I am also grateful to the members of the reading committee, for reading and accepting the manuscript, and for their valuable feedback.

Finally, I want to thank my parents Albert and Myra for their unconditional support, and Karin, love of my life, for her patience, moral support and understanding.

Apeldoorn, October 2011
Axel Brink

Handwriting can be the key to solve a crime. Handwritten texts such as stalking letters, ransom notes, forged suicide notes and threat letters (Figure 1.1) contain individual characteristics that may give the offender away. Today, finding a suspect still requires secondary evidence. His or her handwriting is then studied by forensic document examiners (FDES): based on a handwriting sample, the expert judges to what extent it is likely that the suspect wrote the questioned document by determining correspondences and differences of individual characteristics in the handwriting. This can make the difference between acquittal and custody.

Handwriting also plays a key role in history research, since many historical sources are handwritten. See the document in Figure 1.2 for an example. In some cases, documents in a single collection could be attributed to more than one writer, or writings found at different locations could be attributed to the same writer. For example, the authorship of texts ascribed to famous authors such as Shakespeare or Christine de Pizan is disputed. Furthermore, the identification of anonymous traveling scribes of historical institutions can help to determine their organization and activities. Paleographers study historical documents in an attempt to connect documents to writers and to situate the documents in time and space.

Forensic and historical document examiners perform an important and valuable task, but they are not flawless [36, 37, 56]. This is partly caused by a lack of objectivity [96] and quantification [35]: document examiners compare handwriting based on personal skill and experience, partly based on personal choices and estimations, and many experts do not follow a standardized methodology [66]. Moreover, the experts may be prejudiced by knowledge of context information. Furthermore, humans cannot always perform a complete analysis if a large quantity of handwriting is involved.

To some extent, the limitations of human experts can be complemented by the power of computers. Both the application areas of law enforcement and history research can be advanced [41] by a system for *handwriting biometrics*: a system that attributes handwrit-

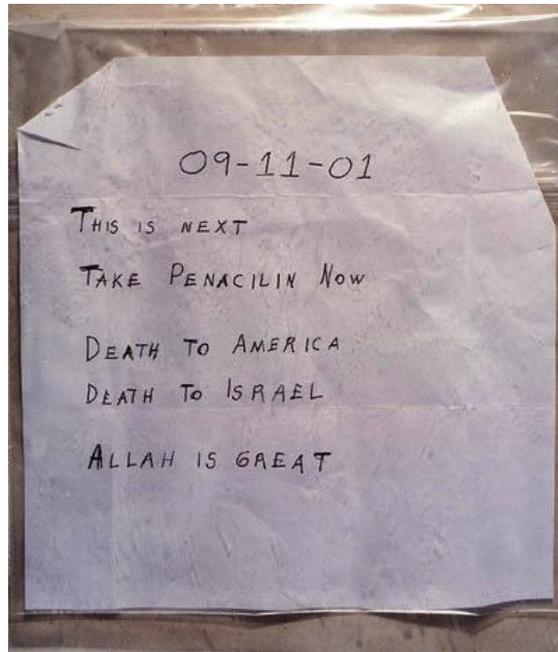


Figure 1.1: Threat letter: one of the letters containing lethal anthrax spores, sent one week after the September 11 attacks in 2001.

ing to a writer based on computationally derived individual characteristics. Handwriting biometrics is a form of *behavioral biometrics*, which also includes biometrics for human gait, voice, and key press timing. In handwriting biometrics, not the actual behavior (the act of writing) is used as a source of individual characteristics, but its recorded result: the handwriting.

Handwriting biometrics comprises two types: writer verification and writer identification. A system for *writer verification* performs a one-to-one comparison of two documents and decides whether the documents have probably been written by the same person. It could be used to strengthen or weaken the judgment of forensic document examiners and paleographers. A system for *writer identification*, on the other hand, is a one-to-many comparison: it identifies the writer of a single query document by searching for similar handwriting in a dataset and yielding a *hit list* of closely matching documents. In law enforcement, this could be used to find suspects without the need for secondary evidence. The only requirement is a dataset of handwriting of known writers. In the

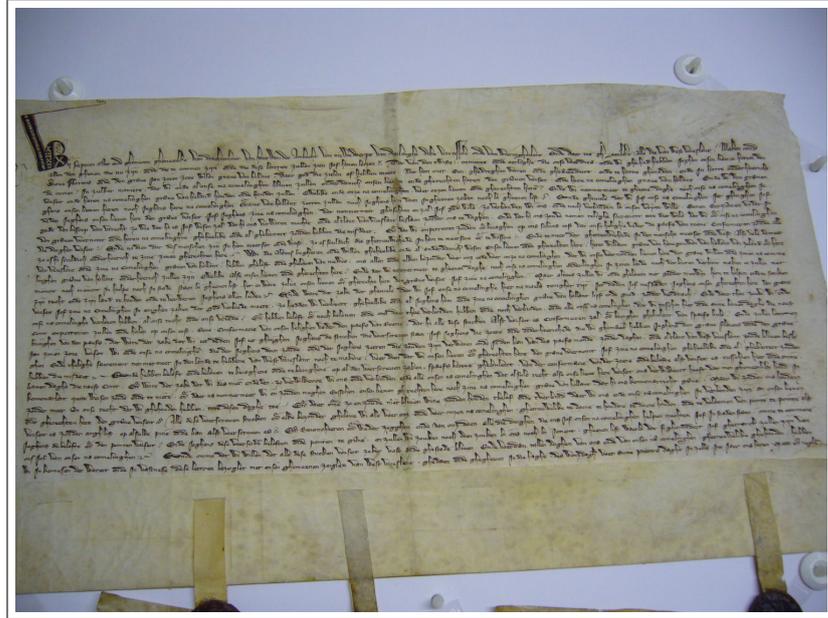


Figure 1.2: Medieval handwriting in a charter, a legal administrative document (1310). *The original charter can be accessed via Nationaal Archief (National Archives of the Netherlands), Archief van de graven van Holland, access number 3.01.01, item number 333. Photo: Jinna Smit.*

case of historical handwriting, it could be used to group anonymous documents that have likely been written by the same hand.

Today, biometric systems for handwriting biometrics are not widely applied yet. Several systems for handwriting biometrics have emerged and some of them seem to be quite accurate (see Table 1.2 on page 17), but these systems require that the handwritten material is neatly and naturally written. In practice however, irregularities in handwriting are common. Realistic input documents have challenging properties such as a scarcity of text, deletions, disguised handwriting or irregular background. It is not known to what extent current systems are robust for such difficulties and how these can be countered. Another shortcoming of current systems is a lack of explainability: the results of the systems are usually expressed in numbers that are incomprehensible to domain experts.

These issues affect systems for writer identification: the performance in real-world conditions is unknown and may be improved. However, since these systems just present suggestions, a limited number of errors is acceptable without impeding applicability.

For systems for writer verification on the other hand, the real-world challenges have to be solved to make such systems applicable. A verification system that is not robust or explainable cannot be used as a second opinion. Handwriting biometrics has the potential to help support or contradict expert opinions and to elicit a speed boost in the work-flow of the two application areas of forensics and paleography, when it is robust and applicable.

1.1 Objectives of this dissertation

Two objectives are central to this dissertation:

1. Analysis of robustness of handwriting biometrics: types of difficulties in realistic input documents will be identified that can be admitted while maintaining high performance.
2. Introduction of new methods to make handwriting biometrics more robust and applicable: principles that aid obtaining high performance and explainability in realistic conditions will be presented.

Furthermore, as a proof-of-concept, the feasibility of robust and applicable handwriting biometrics will be demonstrated in the form of a graphical computer application for writer identification.

This work is part of the NWO/TOKEN-funded research project *TriGraph*, which aims at combining three types of handwriting analysis:

- Manual examination by forensic document examiners at the Dutch Forensic Institute (NFI);
- Semi-automatic methods focusing on character-wise comparison, which have been developed at the Donders Institute for Brain, Cognition and Behaviour in Nijmegen [70];
- Fully-automatic methods based on statistical pattern recognition, to be developed at the Institute of Artificial Intelligence and Cognitive Engineering (ALICE) in Groningen.

This dissertation represents the third type of methods. Only methods that are based on *statistical pattern recognition* will be considered, which are typically fast, accurate, and require little human intervention. The methods are only tested on *off-line handwriting*: scanned or photographed handwriting, which is the form in which forensic and historical material exists.

1.2 Overview

The dissertation is split in two parts. Part I constitutes objective 1: analysis of robustness of current methods when confronted with realistic input. Three common issues will be investigated: text scarcity (Chapter 2), crossed-out words (Chapter 3), and disguised handwriting (Chapter 4).

Part II constitutes objective 2: the introduction of methods to make handwriting biometrics more robust and applicable. A robust system contains several powerful methods, each suitable for a known range of conditions. In Chapter 5, a new feature for writer identification will be introduced, to be used as one of the available methods in a robust system for handwriting biometrics. The feature is powerful and explainable. It is inspired by modern paleography and exploits information in the combination of ink trace angle and width. In Chapter 6 a method is introduced to make handwriting biometrics more applicable by increasing explainability: instead of a list of abstract numbers, handwriting is represented as a unique relative position with respect to the handwriting of a small selection of writers, the *vantage writers*.

The last chapter, Chapter 7, concludes the dissertation. It summarizes the main results and provides new research directions. It is followed by Appendix A, in which GIWIS will be introduced, a graphical computer application for writer identification. It serves as a proof-of-concept that demonstrates the feasibility of robust and applicable handwriting biometrics.

As a preliminary to the two parts, the remainder of this section discusses fundamentals of handwriting biometrics. It will introduce the theoretical basis and it will explain how a straightforward approach based on statistical pattern recognition works.

1.3 Individuality in handwriting

Individuality in handwriting is the result of two factors: the physical properties of the body and the way it is controlled by the brain. The *body* can be seen as a heavy contraction of bones, joints and energy-effective, dampening muscles. The physiological properties of these components such as bone lengths, muscular strength and fatigability are probably different for every individual [84]. The unique way of controlling the body is stored in a unique *motor program* in the brain that controls the body during writing [48, 84]. This program is formed by learning at school and further shaped by observation of writings of other persons and personal preference. Together, these physiological and mental constituents result in individual writing behavior.

This shows in several individual characteristics that can be categorized into eight aspects of individuality *letter form, curve style, slant, pressure, pen grip, proportions, spacing, contents, and consistency*. See Table 1.1 for examples of characteristics for these aspects. Note that some of the characteristics are influenced by the individual pen grip, which determines how human dynamics are translated to pen tip dynamics.

Writer-specific characteristics vary among different writers; this variability is called *between-writer variability*. A complicating factor is that there is also *within-writer variability*: handwriting of any individual is never the same. Humans are subject to variability and cannot copy their own movements exactly, unlike a robot. This is caused by several factors, such as:

- Neuro-biomechanical variability: the neuromotor system is a complex feedback system that results in chaotic behavior;
- Mental state: mood, stress, sleepiness, alcohol level, etc.;
- Aging: over a period of years, a person's physical properties and mental motor program change;
- Injuries;
- External causes: temperature, properties of the writing instrument and support (paper).

It is a challenge to identify features of handwriting that have high between-writer variability and low within-writer variability.

Aspect of individuality	Characteristics in handwriting
Letter form	Allograph usage (the type of copybook letters used) [53]; placement of i-dots, t-crosses etc. [66]; “unusual” letter form (letters that have a non-standard shape) [48]; cursiveness (as opposed to hand-printing) [22, 48]; embellishments [53]; tick marks (short pointed marks) [66].
Curve style	Construction [53]; stroke sequence order; domination of clockwise, counterclockwise, or straight motions [53, 66]; curvature (changes of direction); [19, 48, 66]; line quality (fluency, speed) [48, 53]; legibility or writing quality [53]; stroke direction and length at beginning and ending strokes [8, 53, 66, 87]; compression at line ends [53]; signs of unusual pen grip [53].
Slant	Dominant stroke angle [16, 22, 26, 48, 53, 63]; angle of the long strokes of the letters <i>h</i> , <i>k</i> , <i>p</i> , etc.
Pressure	Pressure variation [48, 53, 66, 89], degree of tapering at beginning and ending strokes [8, 53, 66, 87]. (indirectly measurable)
Proportions	Dimensions [53]; character proportions (height, width, and width/height ratio of within-character parts); [22, 48]; word proportions [53].
Spacing	Connectedness [48] and position of connections [53]; line continuity [53]; alignment with respect to the baseline [53]; spacing and shape of margins [48, 53, 66]; spacing and shape of text lines [48, 53, 66]; spacing between words and parts of words [48, 53]; character, position and frequency of interlineations [53]; depth of indentions [53]; paragraphing [53]; location of headings, signature, address, etc. [53].
Contents	Spelling; use of diacritics and punctuation [53, 66]; use of numerals and symbols in monetary amounts [53]; abbreviations [53]; vocabulary.
Consistency	Consistency of the characteristics mentioned above [48, 53].

Table 1.1: Aspects of individuality in handwriting and characteristics for each aspect as defined in literature. Notice that some characteristics are not directly measurable, such as pressure and pen grip.

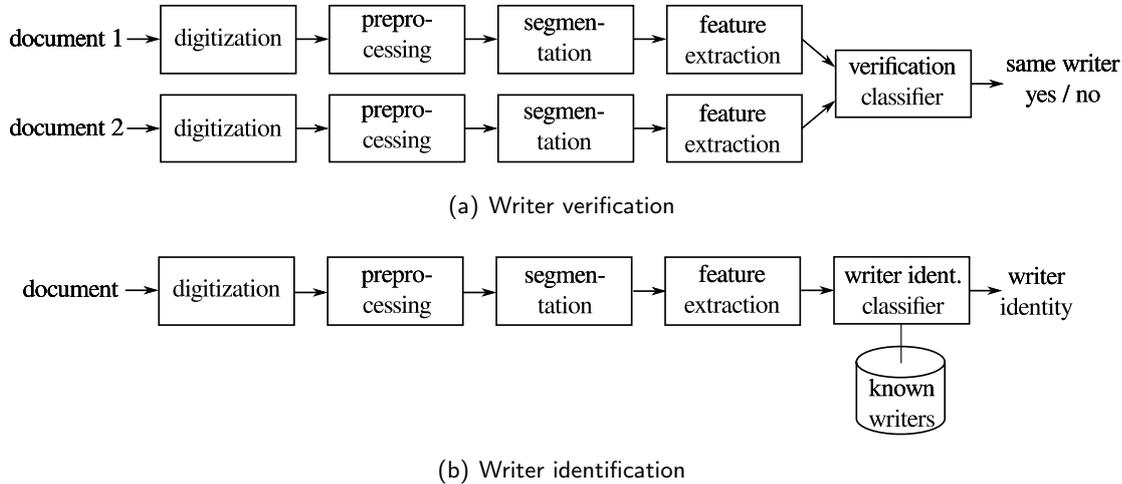


Figure 1.3: Components in straightforward systems for writer verification and identification. A system for writer verification takes two documents and decides whether the documents have been written by the same person with a yes/no result; a system for writer identification takes only one document and returns the possible identity of the writer. These systems are largely identical: only the classifier is different.

1.4 Handwriting biometrics

Straightforward systems for handwriting biometrics consist of several components [33, 81], or steps: digitization, preprocessing, segmentation, feature extraction and classification. These steps are exactly the same for writer verification and identification, except the classifiers are different. This is visualized in Figure 1.3. The order of these steps may be altered to fit the chosen methods. In a robust system, the components may be executed more than once, with different parameter values, as more knowledge is derived in the process. The next sections explain a basic approach of each of these steps individually, followed by common methods of performance evaluation. Later chapters build upon this setup.

1.5 Digitization

The first step of handwriting biometrics is digitization. Documents can be digitized using a scanner, which ensures constant illumination, a flat and well-aligned original, high resolution and it does not suffer from lens distortion. However, a scanner cannot be used for delicate documents, such as old handwriting, as the scanner might damage the document. In such cases, a photo camera setup can be used, consisting of a container for the original, a high-quality digital photo camera, a stand to hold the camera, and one or more light sources to ensure almost-constant illumination. The image resolution must be high enough to capture writer-specific details, such as 300 dpi [44]. The resulting images should be stored in a lossless file format such as .PNG to avoid image degradation.

1.6 Preprocessing

In the *preprocessing* step, the pixel intensities of the image are altered to suit feature extraction and to make the distinction between handwriting text and background as clear as possible. Ideally, the handwritten text turns dark while everything else turns white. Feature extraction requires grayscale or black/white images; the rationale is that text written in ink is usually darker than the background, irrespective of the hue. Thus, color images must be converted to *grayscale*. The simplest method is by averaging the *R*, *G*, and *B* values of each pixel.

Many feature extraction methods require a pure black-and-white image. This is done by *binarization* of the image. A good and frequently used method is *Otsu's thresholding method* [75]; this method applies a global threshold based on a statistic of the histogram of pixel intensities. This is effective on documents with uniform illumination and uniform background intensity.

Real-world documents will require additional, more elaborate preprocessing steps [39]. In chapter 5 we will show how extra steps were used to adapt to handwriting in medieval documents: perspective correction, automatic scaling and high-pass filtering. If such elaborate steps are used, care must be taken to keep the process explainable for admission in a court of law. This can be facilitated by keeping track of all applied preprocessing steps [100].

1.7 Segmentation

In the *segmentation* step, the coordinates and boundaries are determined of the units of interest in the image: single characters, words, text lines, or the complete block of text (the region of interest, ROI). For each unit of interest in one document, features will be extracted and compared to similar units in the other document in later steps of the system. The required type of segmentation depends on the desired type of comparison: text-dependent or text-independent comparison [78].

Text-dependent comparison is a detailed one-to-one comparison of single words or characters. For example, the word “the” in one document will be compared to the same word in the other document. This type of comparison is similar to the human approach and it facilitates high performance as it can draw on knowledge of *what* is being compared. *Text-independent comparison*, on the other hand, comprises the comparison of big regions of text as a unit, disregarding the textual contents. The unit can be an individual text line or the entire block of text. The rationale of comparison on these levels is that the unit is so big that it should contain a representative sample of characters, providing a stable basis for statistical features. It requires text-line or text-block-level segmentation.

The smaller the unit of interest, the more human intervention is required, since segmentation cannot be fully automatized yet (except in special cases). Particularly word-level and character-level segmentation are labor-intensive. This is a major disadvantage of text-dependent comparison. In addition, text-dependent comparison requires that the textual content of each character or word is manually entered, and the textual contents of the compared documents must be equal or at least overlap significantly. These disadvantages do not hold for text-independent comparison. Therefore we will categorize it as “automatic”. Furthermore, it has been shown that text-independent methods can reach high performance (see Table 1.2 on page 17). For these reasons, this dissertation focuses on text-independent methods. In Chapter 2, the minimum required amount of text for such methods will be determined.

1.8 Feature extraction

The heart of handwriting biometrics is *feature extraction*: measuring writer-specific characteristics in each segmented unit. The measured values are reflected in a list of numbers, the *feature vector*. Eventually, not the handwriting is compared, but the handwriting’s

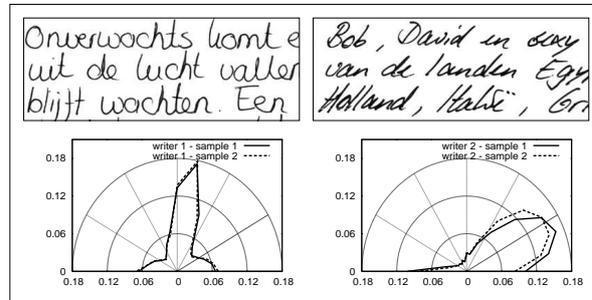


Figure 1.5: Directions feature. (Figure by Marius Bulacu, reprinted with permission.)

this case it was trained with a pre-computed code-book based on modern handwriting: all pages of the first 100 subjects of the Firemaker dataset [88]. *Fraglets* is one of the best features currently available. See Table 1.2.

- **Grapheme occurrence** [6] is similar to *Fraglets*: a dataset of handwritten text is segmented and clustered into graphemes; a binary list indicates occurrence of graphemes in the text and forms the feature vector. See Table 1.2.
- **On-line allograph matching** [73, 71] relies on an off-line to on-line conversion: inferring the writing movement of each character. Based on matching of these on-line characters, the frequency of each character is determined and weighted to global frequencies.

1.8.2 Features encoding curve style

Curve style is determined by traversing the boundaries of the (supposed) characters while accumulating directional statistics. Examples are:

- **Chain code histogram** encodes direction usage frequencies. Each contour in the text is represented as a *chain code*: a sequence of numbers indicating the direction between two consecutive contour pixels. In one approach, [94] the method counts the number of “vertical, negative, positive, [and] horizontal” components. In a more recent approach [90], a histogram of all seven chain code directions is used as feature vector.
- **Chain-code differentials** [90] encode curve style. These features are histograms of the first-order or second-order differential of the chain codes.

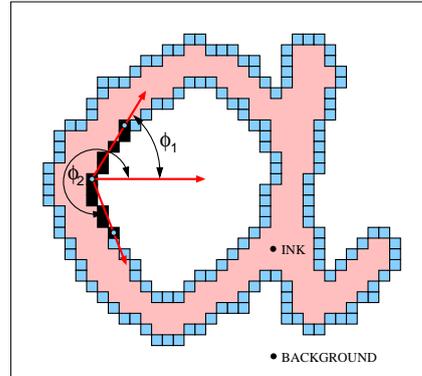


Figure 1.6: The *Hinge* feature determines slant and curve style by measuring angle combinations at the ink boundary. (Figure by Marius Bulacu, reprinted with permission.)

- **Edge-direction histogram** (or “Directions”) [16] encodes direction usage frequencies. It is a probability distribution of ink directions of longer oriented edge fragments along the contours, quantized into a limited number of histogram bins. See Figure 1.5.
- **Hinge** [19, 21] encodes slant and curve style. *Curve style* refers to frequency, direction and degree of bends in the ink trace. Intuitively, it expresses the appearance of curves, corners, round parts, and straight parts. *Hinge* is a probability distribution of angle combinations that are measured on the boundaries of the ink. It is one of the best features available. Figure 1.6 shows a visualization.
- **Micro-features** [94] encode curve style and structure of each character. It is computed by image gradients, gradient combinations and concavity computed in a 4x4 grid per character.
- **Curvature index histogram** [90] encodes curvature.

1.8.3 Features encoding slant

- **Angle frequencies** [94]. The average writing slant is computed as a weighted average angle of near-vertical contour fragments, where the vertical dimension of the fragment determines the weight. Variations include computing an edge-direction histogram and finding the maximum or mode in it [16] or the peak that is closest to 90° [26]. Another variation computes the average angle in rectangular

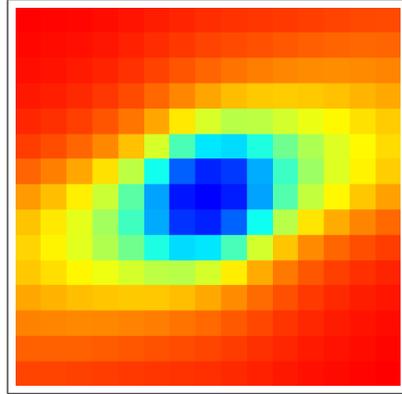


Figure 1.7: The *Brush* feature is a 2D probability distribution of ink intensities at stroke endings.

sub-areas containing vertical structures [9].

- **Repeated shearing** [57, 103]. This method determines the slant angle based on repeatedly shearing images of individual text lines, varying the shear angle, and optimizing a criterion on the vertical projection histogram.

1.8.4 Features encoding pressure

- **Gray-level threshold** [94] encodes average pen pressure. This approach uses the threshold value found using Otsu's method [75] as a feature.
- **Black pixel count** [94] encodes average pen pressure, thickness of strokes, and size of writing.
- **Gray-level distribution** [94]. This feature measures the variation of pen pressure. It is computed by the entropy of the histogram of gray levels.
- **Brush** [87] encodes habitual pressure and stroke direction during pen landing and lifting. It is a probability distribution of ink intensities at stroke endings. See Figure 1.7.

1.8.5 Features encoding proportions

- **Text line height** [94] encodes the average height of the text lines. This can be computed by considering the average distance between maxima and minima in the

upper contour of the text lines.

- **Paragraph aspect ratio and indentation** [94]. This can be computed after semi-automatic selection of the paragraph regions.
- **Word proportions** [94] encodes the upper zone ratio, lower zone ratio, and length of words. These can be computed after semi-automatic selection of a specific word in each input document.

1.8.6 Features encoding spacing

- **Contour counts** [94] are measures of the connectivity of handwriting. It is computed by contour tracing followed by counting the number of inner and outer contours. The ratio of these two numbers is different for isolated characters than for connected characters.
- **White runs** [2, 87] (also named “HrunW”) encodes within- and between-letter spacing. It is a probability distribution of white run-lengths. The best approach counts the run-lengths horizontally, but a vertical approach can also be used, or a combination of the two.

1.8.7 Features encoding contents

Analysis of textual contents to determine the author is the field of stylometry [97, 61]. It is not used in conjunction with automatic handwriting comparison yet, since it requires automatic recognition of the textual contents, which is not good enough for practical application yet.

1.8.8 Features encoding texture

A new possibility of computer methods is to see the handwriting purely as a textured pattern and derive features describing the appearance of the texture. This appearance is determined by writer-specific traits. Although it can be argued that some of the features mentioned above also treat handwriting as a texture, the following features are purely textural:

- **Autocorrelation** [19] describes regularities in the horizontal direction, for example regularly-spaced vertical strokes. It is computed by the correlation between the

image pixels and a horizontally shifted version. It can be seen as Fourier analysis along the pixel rows.

- **Autoregression** [44] (*f16*). The handwriting is treated as a purely textural pattern; the coefficients of 2D autoregression are used as features.
- **Gabor filtering** [80] is based on visual processing in the human visual cortex.

1.8.9 Features encoding consistency

Elaborate approaches model the variability (or consistency) of the features using Gaussian mixture models (GMMs) [83] or other methods that require learning. In straightforward implementations, however, consistency is not determined; the measured features are simply averaged.

1.9 Pairwise comparison

Comparison of feature vectors comprises a mapping from two input vectors to a single value, signifying the degree of (dis)similarity. This can be done using trained and untrained methods. Trained methods include Gaussian mixture models [83] and neural networks. In this dissertation, untrained (dis)similarity measures have the focus because they are simple yet powerful. Untrained methods include (dis)similarity measures such as the following three (where \mathbf{x} and \mathbf{y} refer to the two compared feature vectors). For a more elaborate overview, see [25].

- The Manhattan or City-block distance:

$$d_M(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{x}|} |\mathbf{x}_i - \mathbf{y}_i| \quad (1.1)$$

A simple distance measure. It is equal to the L_1 -norm of the difference of the feature vectors: $\|\mathbf{x} - \mathbf{y}\|_1$.

- The Euclidean distance:

$$d_E(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{|\mathbf{x}|} (\mathbf{x}_i - \mathbf{y}_i)^2} \quad (1.2)$$

This distance measure is frequently used. It is equal to the L_2 -norm of the difference of the feature vectors: $\|\mathbf{x} - \mathbf{y}\|_2$.

Publication Features		Wri- ters	Text- indep.	Auto- matic	Ident. top1	Ident. top10	Verif.
Schomaker et al. [86]	Fraglets / fCO ³	150	yes	yes	97%	100%	
Schlapbach et al. [82]	HMM-based	100	yes	no	97%	98%	98%
Bulacu [21]	Fraglets + Hinge + White runs	900	yes	yes	87%	96%	97%
Siddiqi et al. [90]	Chain code directions + Chain-code-based curva- ture + Curvature index	650	yes	yes	86%	97%	97%
Srihari et al. [94]	Micro-level character-based features + Slant + Direc- tion freq. + Gray-level distr. + Gray-level thresh. + Black pixel count + Text line height + Contour counts	900	no	no	88%		96%
Bensefia et al. [6]	Grapheme-occurrence	150	yes	yes	87%	99%	96%
Garain et al. [44]	2D autoregression	422	yes	yes	62%	96%	

Table 1.2: Performance of leading systems for handwriting biometrics in related work, tested on handwriting created in laboratory conditions. The performance figures cannot be compared because the used datasets differ in size and type of handwritten material. Some systems require a high degree of human interference, in the form of manual character selection or training optimization (marked with “no” in column “Automatic”). In particular, systems based on a hidden Markov model (HMM) require in-depth statistical modeling per writer. Other systems assume that the pages in the corpus contain the same text (indicated with “no” in column “Text-indep.”).

- The χ^2 distance:

$$d_{\chi^2}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{|\mathbf{x}|} \frac{(\mathbf{x}_i - \mathbf{y}_i)^2}{\mathbf{x}_i + \mathbf{y}_i} \quad (1.3)$$

This distance measure emphasizes relative differences in small feature values. It is frequently used in this dissertation because it has been shown to be effective on probability distributions of writer-specific features [85], such as *Hinge* and *Fraglets*.

1.10 Classification for writer verification

Based on a feature extraction method and a distance measure, handwriting can be classified with a classifier for writer verification or writer identification. A classifier for *writer verification* performs a one-to-one comparison of two feature vectors and decides whether the corresponding documents have probably been written by the same person. See also Figure 1.3 on page 8. In the most basic form, a writer verification classifier has to make the difficult decision whether two documents have been written by the same person. Such a classifier is a dichotomizer, which yields *true/false* or *yes/no*, based on a threshold θ :

$$h(\mathbf{x}, \mathbf{y}) = \begin{cases} true & \text{if } d(\mathbf{x}, \mathbf{y}) < \theta \\ false & \text{otherwise} \end{cases} \quad (1.4)$$

The threshold represents a zone of tolerance on the feature distance domain for attributing two documents to the same writer, which is needed because of within-writer variability [81]. This implies that false matches and/or false rejections can occur. The value for θ that minimizes such errors can be learned by training.

The best value for θ can be learned by modeling the feature distances between pairs of documents in the two classes: the class of pairs of documents written by the same writer, and the class of those written by different writers. An example of such models is shown in Figure 1.8. To build the models, a dataset of reference handwriting is needed. This dataset will be regarded as being representative for all future classifications. After preprocessing and feature extraction, pairs of documents are drawn from the dataset in all combinations. The distributions of feature distance in the two classes form the basis for the models. The models can be made using any parametric or nonparametric density estimation method; see [33] for an extensive overview. Approaches using parametric methods often assume a Gaussian distribution, which is suitable for the different-writer

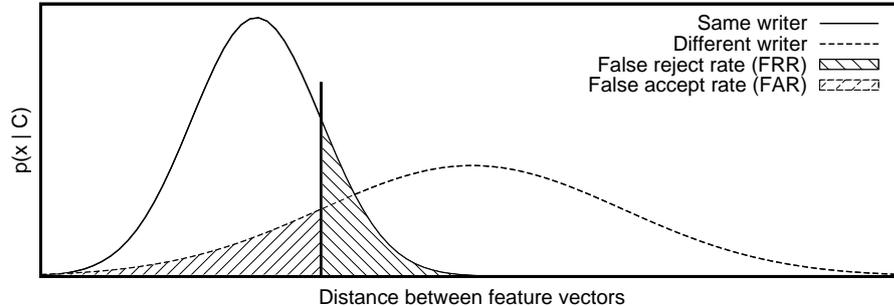


Figure 1.8: Probability densities of distances between feature vectors for two classes: the same-writer class (left curve) and different-writer class (right curve). In writer verification, a threshold on the feature distance determines if two documents are attributed to the same writer (“accepted”). Since the densities usually overlap, errors are expected: a false accept rate (FAR) and a false reject rate (FRR).

distances but less accurate for the same-writer distances, because these are frequently near-zero positive values.

Using these models, the threshold θ must be fixed such that optimal expected performance is acquired. Given this threshold, two kinds of errors can occur: Type-I errors and Type-II errors. A Type-I error refers to a *false accept*, in this case meaning the false attribution of two documents to the same writer. Conversely, a Type-II error refers to a *false rejection*: falsely attributing the documents to different writers. As in practice the two bumps always overlap because handwriting biometrics is not perfect, errors will occur. The expected rate of Type-I errors must be balanced with the rate of Type-II errors according to the user’s preference. See Figure 1.8. A straightforward approach is to set the threshold such that both error rates are expected to be equal; the *equal-error rate* (EER). This can be done by integrating the probability density models into cumulative density models and finding the crossing point. See Figure 1.9.

1.10.1 Remarks

The most important advantage of a classifier for writer verification is that it embodies inherent objectivity because it is not influenced by prejudice, additional clues, or witness testimonies. However, it is inappropriate to provide a yes/no classification, for several reasons:

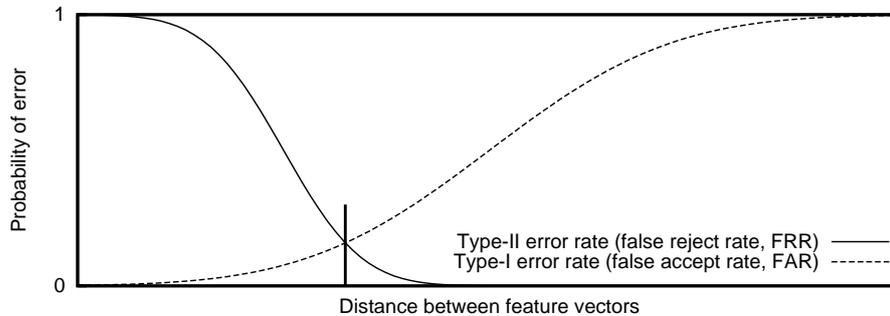


Figure 1.9: Cumulative probability density functions. The y value of the crossing point indicates the expected equal-error rate (EER); the x value of this point is often used as the verification threshold value θ .

- The classifier does not take context information into account.
- No systems exist yet that are robust for forgeries and disguise, which are common in forensic cases.
- It is impossible to make a representative dataset for training and testing the classifier, because the prior probabilities are unknown and hard to obtain: it is not known how frequently two presented documents are actually written by the same writer. In particular, it is not known how frequently forgeries and disguised writings occur, and certain authorship information of forged documents is not available, unless statistics of confessions are kept.

These issues transcend handwriting biometrics. Human document examiners face the same problems. Moreover, the third issue is problematic for any biometric verification with overlapping class distributions. Therefore, it seems appropriate for an applicable system for writer verification to just provide the degree of similarity between the compared documents, presented in an explainable way. This enables a final judgment (by a judge) in which an estimate of the prior probability is taken into account and other evidence such as DNA, footprints, testimonies, etc., which could contradict the judgment suggested by the handwriting alone. Nevertheless, in this dissertation (and in most related work) a simple dichotomizer is still used because it enables straightforward performance testing and comparison to other systems.

1.11 Classification for writer identification

A classifier for *writer identification* takes a single feature vector of one document and yields a list of closely matching feature vectors in the dataset. The most simple and commonly used classifier for writer identification is a *nearest-neighbor*-classifier. In this case, writer identification is done by repeated pairwise comparison: the feature vector of the query document is compared to the feature vectors of all documents in the dataset. The result is a *hit list*: a sorted list containing the s nearest neighbors of the questioned document, plus the author identities. s usually has one of the values 1, 10, or 100.

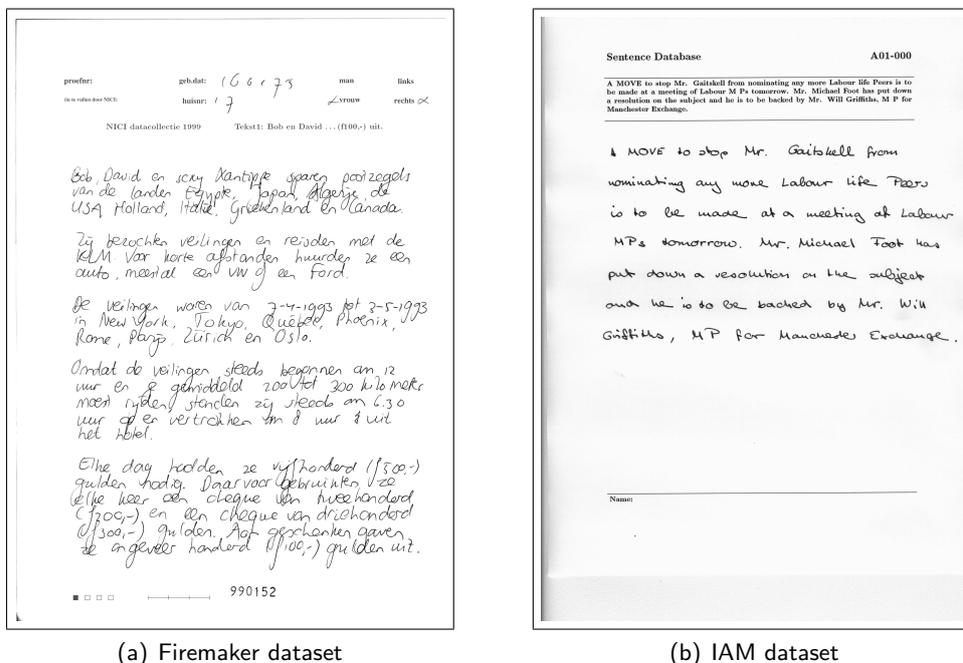
Training a nearest-neighbor classifier for writer identification consists of simply storing the feature vectors of all dataset documents. No modeling is required. However, training a complete system can include the optimization of the parameters of other system components such as feature extraction.

Many other classification methods could be used instead, but there are a few reasons for choosing nearest-neighbor classification: it is simple, effective and it does not require training or human intervention. Whereas trained classifiers allow for generalization of observations into a broad class, biometrics is concerned with the identification of individual samples. Nearest-neighbor search fits that goal very well and does not require evolved training. Furthermore, all common parametric classifiers require large amounts of training examples, a requirement that cannot be easily fulfilled in practical settings. Previously, several other popular classifiers such as MLPs and SVMs have been tried on writer identification tasks in unpublished studies at ALICE, but the nearest-neighbor classifier was never surpassed.

1.12 Performance evaluation

The standard approach to evaluate the power of a system for handwriting biometrics is to use idealized datasets of handwriting in which the authorship of every page is known and represented by a code or number. The handwritten text in these datasets is produced in laboratory conditions, following a strict format to facilitate automatic processing. It is common to instruct a group of subjects to copy a fixed text on provided forms. The documents lack many aspects of variability found in the real world; the resulting performance represents the best-case scenario.

An example of such a dataset is *Firemaker* [88]; see Figure 1.10(a). This dataset contains 1004 pages of contemporary Dutch handwriting, written by 251 students. Each



(a) Firemaker dataset

(b) IAM dataset

Figure 1.10: Example of contemporary handwriting in laboratory datasets.

student wrote 4 pages. The pages were lined with a color that vanished during scanning. For each subject, page 1 contains natural handwriting, which is a copy of a fixed text consisting of 612 alphanumeric characters; page 2 contains text in capital letters, page 3 contains free forged handwriting and page 4 contains natural free text of 252 characters on average. All pages were scanned at 300 dpi. This dataset is frequently used in this dissertation and in other studies as well. In this dissertation, only page 1 and page 4 are used, since these contain unconstrained connected cursive handwriting.

Another widely used laboratory dataset is the *IAM dataset* [65] (Figure 1.10(b)). It contains contemporary English text written by 657 subjects; the number of pages per subject varies. The pages were written in natural handwriting, using different writing instruments, and scanned at 300 dpi. As a last well-known example, *Srihari's dataset* [94] contains 4500 full pages of English text written by 1500 subjects; each subject wrote three pages with the same content. The subjects are a representative sample of the U.S. population.

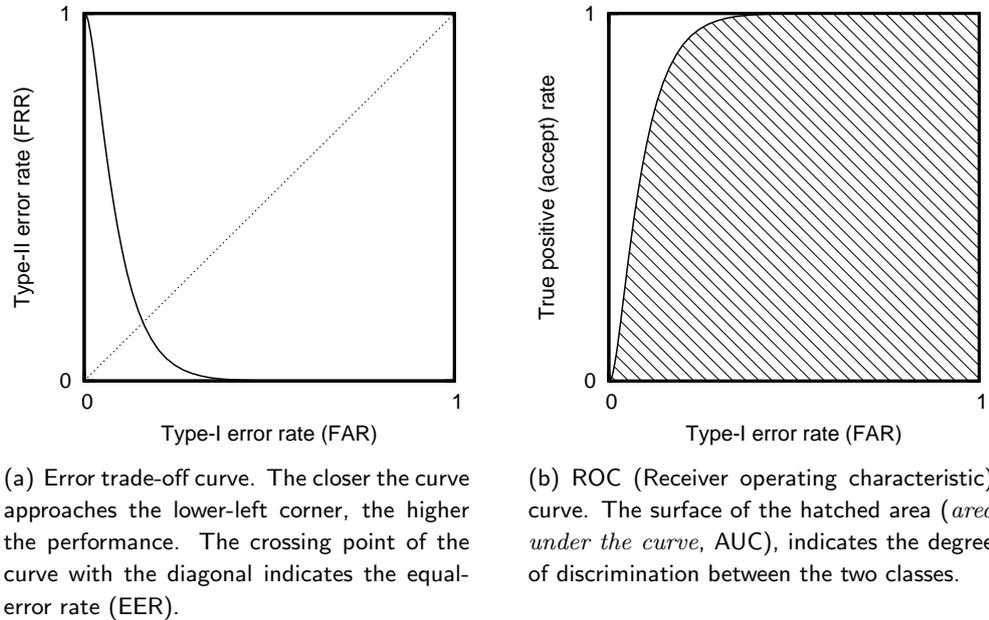


Figure 1.11: Error trade-off curve and ROC. The graphs are equivalent, except the vertical axis is inverted.

A *writer verification* classifier can be evaluated on such a separate laboratory dataset. All pairs of documents are drawn from the dataset and the threshold θ is used to make a decision, as described in Equation 1.4. The classification result is checked based on the identity information in the dataset. The performance is expressed as the percentage of errors or the percentage of correct classifications. Another way to assess the performance is by using an error trade-off curve ROC plot [34], which show the performance for any threshold value. See Figure 1.11.

The performance of a *writer identification* classifier can be evaluated by treating every dataset document once as a query document, and checking if another document from the same writer appears in the resulting hit list. If this is the case, the classification is called “correct”, disregarding other instances in the hit list that may be written by someone else. The percentage of correct classifications can then be used as a performance measure.

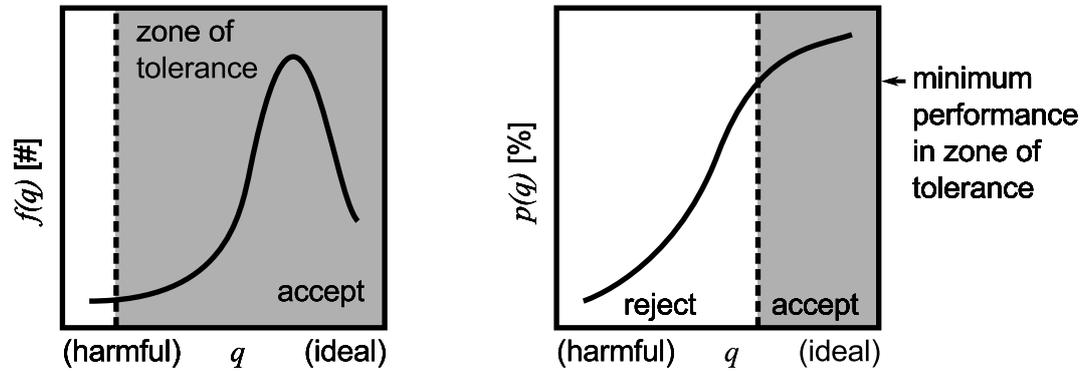
Performance figures are not a complete description of a system’s capabilities. In ad-

dition, it matters how difficult the classification task was and how much user interaction is required. Among other things, the difficulty of the classification task is influenced by the size of the dataset, the number of writers, the handwriting style of the writers, the textual content and the graphical quality of the input material. The amount of user interaction is influenced by the level of segmentation, the amount of human-directed system training, and the regularity of the handwriting. Therefore, the performance figures in the overview of current systems in Table 1.2 (page 17) cannot be truly compared.

A robust and applicable system should perform well on a large corpus of realistic handwriting, is text-independent and requires little user interaction. Such a system is presented in Appendix A of this dissertation. Its most important component is the *Quill* feature, which competes with leading existing methods. This feature will be introduced in part II, followed by a method to make systems more explainable. Before this, aspects of the robustness of current systems are discussed in part I.

Part I

Robustness analysis of handwriting biometrics



(a) Frequency of occurrence (f) of input quality (q). A threshold on input quality could be defined such that most realistic documents are accepted.

(b) Performance (p) as a function of input quality (q). A threshold on input quality could be defined such that only a subset is accepted on which high performance can be achieved.

Figure 1.12: Determining a zone of tolerance by setting a threshold on input quality.

Realistic input documents are often of suboptimal quality. Typical issues affecting quality include factors of imaging quality (resolution, contrast, illumination, perspective, lossy compression), material quality (age degradation, preprinted patterns or text, irregular ink absorption, holes, wrinkles, tears) and handwriting quality (sloppiness, visible effects of abnormal writing conditions, disguise, forgery). More factors can be found in [42]. Such quality factors affect performance: as quality decreases, system performance decreases as well [5]. Naive systems will process any input material, irrespective of quality, and will produce erroneous results on artefacts (“garbage in, garbage out” [60]). A system for handwriting biometrics must be resilient for such realistic performance influencing factors: it has to be *robust*.

Robustness means that system performance degrades gradually as input quality decreases, instead of suddenly (*brittleness* [23]). In a strict definition, this should hold on the entire theoretical range of input conditions, including conditions that are highly unlikely. This type of robustness will be called *intrinsic robustness*. A system incorporating intrinsic robustness would require training and testing on unrealistically difficult data, resulting in low tested performance, thus the applicability of such a system is limited.

In a less strict definition, a sudden performance drop is allowed outside the realistic range of input conditions. This requires the determination of a threshold on the input

quality, bounding the *zone of tolerance*. Obviously, input quality is not a one-dimensional quantity: each performance influencing factor is a dimension of input quality, and input quality is a point in multi-dimensional space, thus the zone of tolerance is a multi-dimensional box. For each dimension of input quality, the threshold could be established using a quality frequency graph such as in Figure 1.12(a). However, such a graph is difficult to obtain and it is not feasible to obtain good performance on the full range.

Therefore, in this dissertation the focus is on *practical robustness*: high system performance within a broad zone of tolerance. For any document within this zone, system performance will be high. Other documents have to be rejected, just like unclear fingerprints cannot be admitted to forensic analysis. To determine the threshold for the zone of tolerance, a graph is needed that describes the relation between input quality and system performance such as in Figure 1.12(b).

In the three chapters in this part it is determined to what extent realistic documents fall within the zone of tolerance; each chapter focuses on a single dimension of input quality. Chapter 2 focuses on text scarcity; the relation between the amount of text and system performance is investigated and the zone of tolerance is determined. In Chapter 3 it is determined whether document that contain a realistic fraction of crossed-out text fall within the zone of tolerance. In Chapter 4, a simple method is used to counter disguise by slant manipulation, effectively increasing the zone of tolerance for this dimension of input quality.

Chapter 2

Robustness for text scarcity

A modified version of this chapter was previously published in *Proc. of the 19th International Conference on Pattern Recognition (ICPR)*. [11]

The reliability of writer verification and writer identification depends on the amount of text in the handwritten documents: the more text is present, the more evidence is available. This is relevant for the forensic application domain, where the amount of text varies from snippets to complete letters. Statistical feature extraction methods, such as *Hinge* and *Fraglets*, build a model of writer specific features that can only be accurate when the number of measurements is large, thus they need a sufficient amount of text. This has been shown in previous studies: In [55], a writer verification experiment yielded a performance of 63% using a single line of text per page, while the performance on half a page was 95%. In [20], a writer identification experiment yielded 53%, 83% and 88% on a single line, half a page and a full page, respectively, using the best feature. It showed a similar pattern for two other features. These scores are illustrated in Figure 2.1. In another study, on-line characters were used, and a minimum requirement of 160 characters was found [101].

The objective of this study is twofold. The first objective is to gain insight in the relation between text amount and the performance of current writer verification and identification systems into more detail. This facilitates judging the reliability of the outcome of automatic classification based on the amount of text. The second objective is to derive a rule of thumb for the *minimum* amount of text that is required for reliable classification using the best features. Such a minimum could be used as an acceptance criterion in input images. It is also useful for the design of new datasets for automatic writer verification and identification. Such a minimum cannot be firm, because performance does not only depend on the amount of text but also on other factors such as feature quality, text quality and database size.

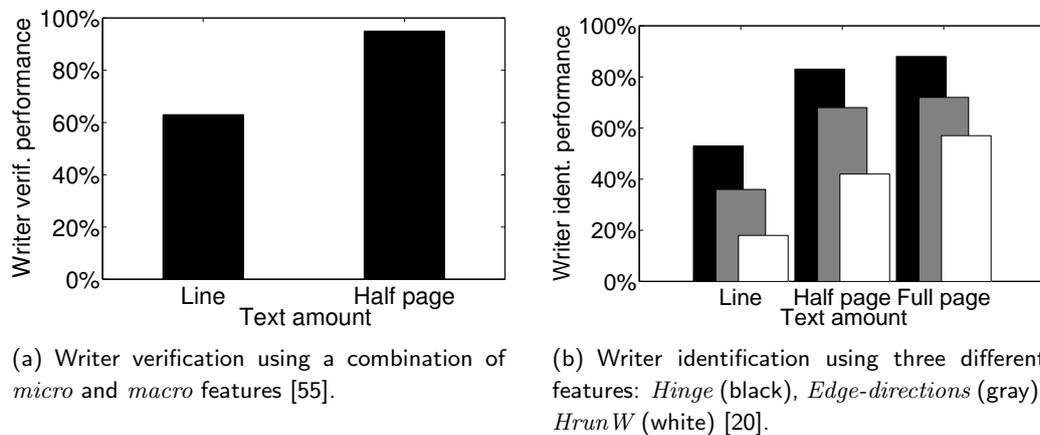


Figure 2.1: Previous studies show that classification performance increases when the amount of text increases.

2.1 Method

The relation between text amount and performance was examined by repeatedly conducting writer verification and identification performance tests while gradually increasing the amount of text on the pages. The amount of text on the pages was controlled by *wiping* text from complete pages of existing datasets. It was also determined what happens when one of the compared documents contains little text (a partially wiped page) while the other contains a lot (a full page). The next subsections describe the preparation of datasets and the text wiping methodology. The last subsection shows how these were used in a series of performance experiments.

2.1.1 Datasets

Two handwriting datasets were used: Firemaker [88] and IAM [65], which are described at page 21. Both datasets were split in two parts: part Q represents questioned documents and part S represents sample documents. This mimics the realistic forensic scenario where the authorship of questioned documents is to be determined while a database of documents of known authorship exists. Not all documents in the original datasets were put in Q or S : pages with less than 200 characters were discarded and the same happened to pages of writers that wrote only one page. Of the remaining pages,

for every writer, the page containing the most text was added to S and the next page was added to Q . Additional pages were discarded as well. For the Firemaker dataset, the resulting sets Q and S both contained 192 scans (118 pages were discarded). For the IAM dataset, Q and S both contained 298 scans (378 pages were discarded).

2.1.2 Text wiping

We present `line_wipen`, a method to wipe handwritten text on a line-by-line basis until an estimated n characters remain. It is natural to use a line-based approach because it is straightforward and it keeps the appearance of the handwriting almost intact. The general idea is to wipe some lines completely and one line partially. This approach requires that the text lines can be separated well. This is not feasible in general free text but it is relatively easy in the Firemaker and IAM dataset, because the text lines are quite horizontal and the overlap between descenders and ascenders of consecutive lines is minimal. The locations of the text lines can be found with projection histograms. For the Firemaker and IAM datasets, this has already been done in [16] and [65] respectively. From these publications, the coordinates of boxes around the text lines were re-used for this experiment.

Based on these boxes and a desired number of characters n , the handwritten text can be wiped. Ideally, after wiping, exactly n characters would remain. In practice, this is not possible, because it is not yet possible to segment free text into characters automatically. Instead, n is converted to a desired number of ink pixels: $d = \alpha \cdot n$. The variable α denotes the average number of ink pixels per character on the page, which can be computed by dividing the number of dark pixels in the image by the number of characters in the transcription. As an example, Figure 2.2 illustrates the distribution of α on the scans of the Firemaker dataset.

The next step is to remove full text lines by filling the bounding boxes with white pixels. The lines are wiped in random order while maintaining a variable r , the number of remaining ink pixels on the page. When wiping the next line would decrease r below d , the line contains too much ink to be removed entirely. That line is wiped partly as follows. First, the vertical projection histogram of the line is created and then smoothed using a Gaussian window. The valley that is closest to the point where $r = d$ is designated as the cutting point. The right part of the line is wiped starting from this point. This approach makes it likely that the cut is made between characters and not through characters, thus lowering the risk of damaging the letter shapes. See Figure 2.3 for an example.

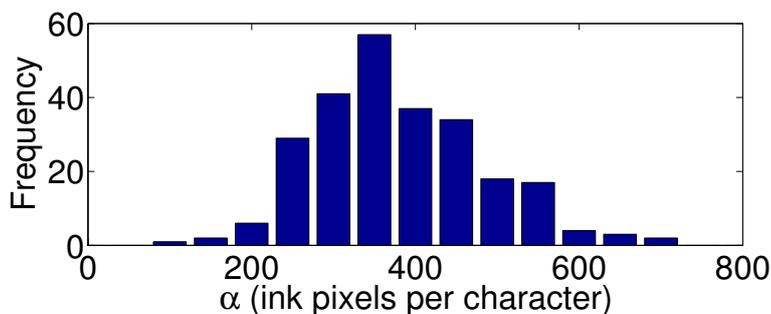


Figure 2.2: Distribution of α (number of ink pixels per character) on 251 pages of the Firemaker dataset: page 1 of every writer. The distribution illustrates that the writers deposited different amounts of ink per character. A document's α value is required for wiping the approximately correct number of characters. Total number of characters in this experiment: 153k.

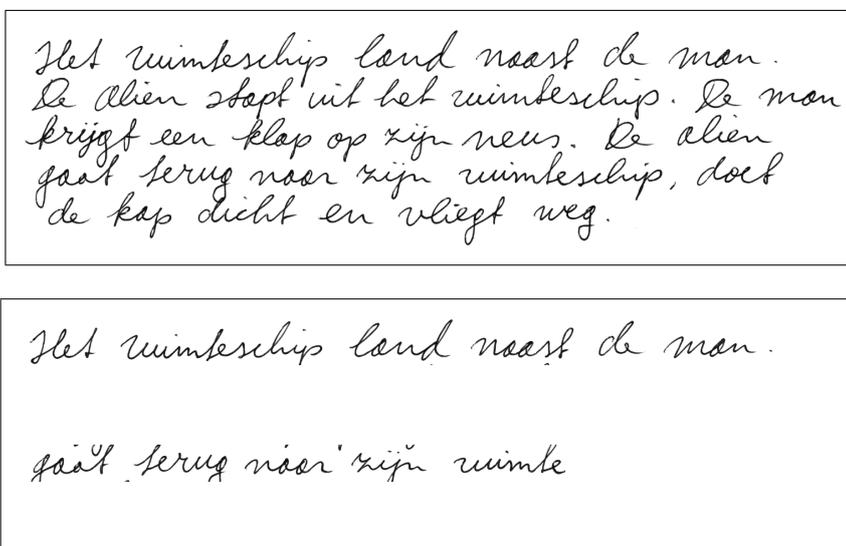


Figure 2.3: Text before (top) and after (bottom) `line_wipen=50` was applied. In this example, actually 51 letters remained.

2.1.3 Experiment

In the experiment, the text wiping methodology was used to create graphs that show the relation between text amount and performance: while varying the number of characters n , the performance of both writer verification and writer identification were computed. This was done in two variants: in one variant, the documents in part Q and S of the dataset were both wiped such that they contained the same number of characters n . In the other variant, only the pages in Q were wiped, while the documents in S remained intact. This distinction respects the difference in text amount between the compared documents in forensic practice. Algorithm 1 outlines the experiment in pseudocode. The algorithm was run on both the Firemaker and IAM dataset using four different features: *Hinge*, *Fraglets*, *Brush* and *HrunW* (see Section 1.8).

Algorithm 1 Complete experiment for a dataset D and a feature extraction function \mathbf{f} . Compute performance of writer verification and identification, with equal and unequal amounts of text. `split_dataset` refers to 2.1.1; WV and WI indicate writer verification/identification performance.

```

 $Q, S = \text{split\_dataset}(D)$ 
for all  $n$  in  $[4, 9, 16, 25, \dots, 196, 200]$  do
   $QW = \text{line\_wipe}_n(Q)$ 
   $F_{QW} = \mathbf{f}(QW)$  (compute features)
   $SW = \text{line\_wipe}_n(S)$  (for equal text amount*)
   $F_{SW} = \mathbf{f}(SW)$  (compute features)
   $\text{verif\_perf}(n) = \text{WV}(F_{QW}, F_{SW})$ 
   $\text{ident\_perf}(n) = \text{WI}(F_{QW}, F_{SW})$ 
end for

```

* For unequal text amount: $SW = S$

2.2 Results

Figure 2.4 shows the results of the experiment. The eight graphs each represent a combination of a feature and a dataset. In each graph, the Top-1 and Top-10 performance and the verification EER are shown; each in a variant of equal-text amount and unequal-text amount. The graphs confirm that *Hinge* is a strong feature extraction method, closely followed by *Fraglets*. They also clearly show that increasing the amount of text also

increases writer verification and identification performance. From these graphs, a rule of thumb for the minimum text amount was derived by visual inspection. Alternatively, the threshold could be determined numerically, provided an exact definition of the threshold is given. As a rule of thumb, the following minimum amounts of text can be considered:

Feature	Min. text amount (chars)
Hinge	100
Fraglets	150
Brush	200
HrunW	200

These minima were drawn into the graphs as vertical lines. The graphs also consistently show better performance for comparison of texts that have an unequal amount of text. Thus, it is always better to increase the amount of text in the database documents, even when the amount of text in the query documents remains the same.

2.3 Conclusion

As a rule of thumb, a document of 100 characters contains a good minimum text amount for text-independent writer verification and identification when using a strong feature such as Hinge. Any text above this minimum does not significantly increase performance any further; the feature itself becomes the limiting factor. For features that are less powerful, such as Brush, a reasonable minimum is 200 characters. These numbers are not absolute because the performance also depends on other factors such as the size of the database. In general, the more difficult the classification task, the more text is needed. In any case, more text is always better, even when the amount of text is increased in only one of the documents in pairwise comparisons. A follow-up to this study should consider text-dependent methods, since those are more similar to the manual methods used in forensic practice, where text samples can be very small.

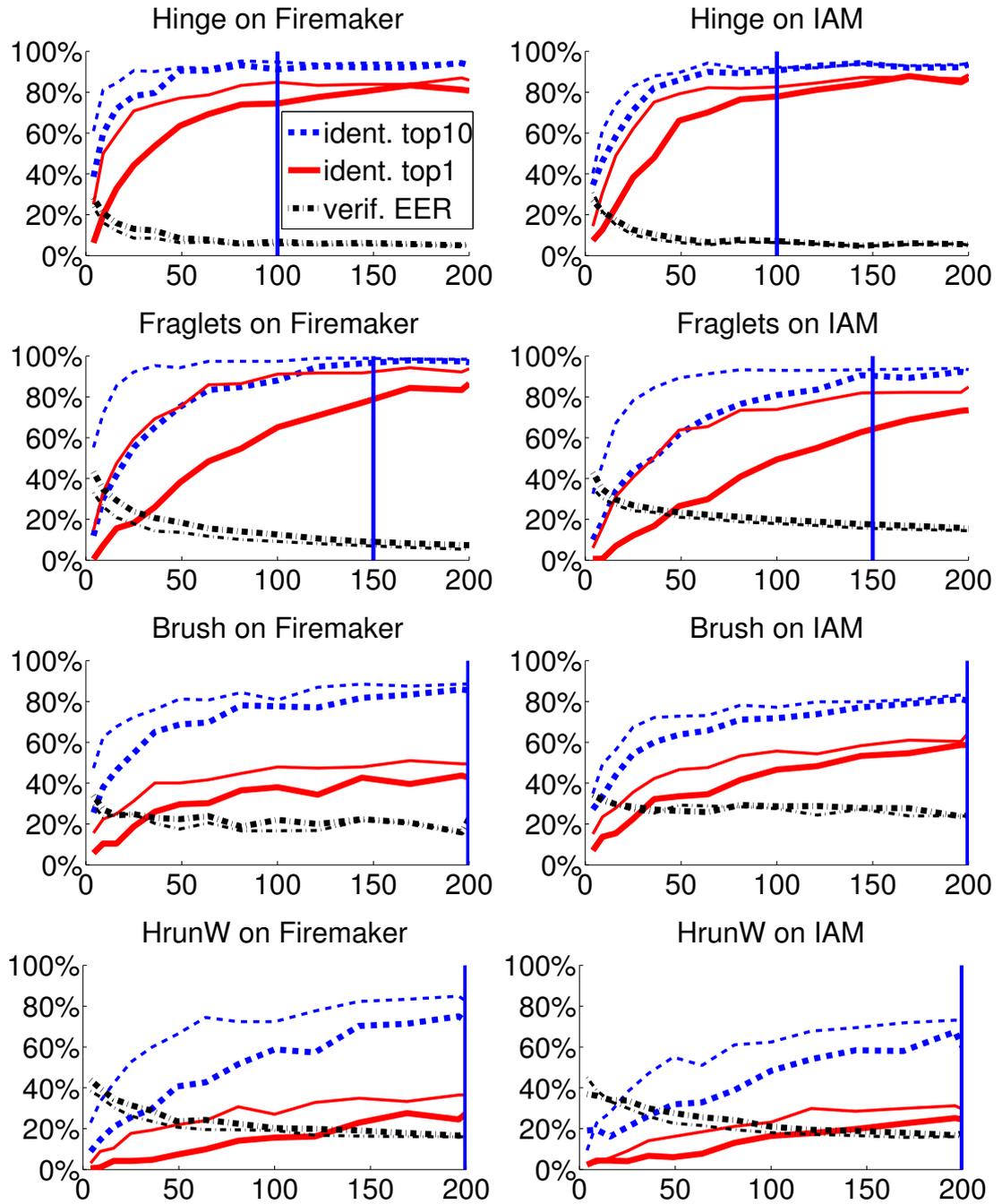


Figure 2.4: Relation between text amount (in characters; horizontal axis) and performance of writer verification and identification. Thick lines indicate equal text amount; thin lines indicate unequal text amounts (text is wiped in only one of the two pages in each comparison). The vertical lines indicate a suitable minimum text amount. The *Hinge* feature requires only 100 characters for for near-optimal performance; other features need more.

Robustness for crossed-out words

A modified version of this chapter was previously published in *Proceedings of Document Recognition and Retrieval XV, IS&T/SPIE International Symposium on Electronic Imaging*. [15]

Another challenge for handwriting biometrics is crossed-out text: without countermeasures, such text will be processed like any other ink blob, which may give incorrect results. This is problematic for automatic handwriting recognition [59] and it is often assumed that crossed-out text also impedes computation of writer specific features of the handwriting, because it is irregular [13]. It is conceivable that feature extraction methods find many bogus features in the crossed-out text that may not be present in another text of the same writer, decreasing the apparent similarity. Therefore, it seems appropriate to attempt to remove crossed-out text prior to automatic writer verification and identification.

Identifying crossed-out text has not attracted much attention yet. One approach focused on separate characters and distinguished characters from noise including crossed-out characters [3]. In a more recent approach, Markov Random Fields were used to identify crossed-out words in very challenging documents [69]. A special property of that particular method is that it seems robust against connections between crossed-out words and normal words. The results look promising, but the performance has not been quantified.

In this chapter a method for identifying crossed-out words in offline handwriting is proposed. It works on the level of connected components and classifies them based on two features of the skeleton: the *branching* feature and the *size* feature. The system is trained and tested on a part of a real forensic dataset, called the NFI dataset. This dataset was first introduced in [13]. It consists of 3500 handwritten samples taken from suspects in criminal cases; these samples have previously been studied manually by the NFI, the Dutch National Forensic Institute. Apart from the textual content, the

handwriting is unconstrained and contains many crossed-out words. See Figure 3.1 for an example. This dataset seems to be somewhat similar to the kind of data used in [102], which consists of spontaneous handwriting. The pages were each cut in two parts, as will be described later.

Training and testing was performed in three stages. In the first stage, the classification performance was assessed on the level of connected components in the first 250 pages of the NFI dataset. In the second and third stage, this classification is applied to assess the effect on writer verification and identification, respectively, on 2374 pages. This is described in the next sections.

3.1 Recognizing crossed-out words

3.1.1 Preprocessing and segmentation

The first 250 pages of the NFI dataset were thresholded using Otsu’s thresholding method [75]. From the result, the black connected components were extracted based on 8-connectivity. Two kinds of components were discarded: very small components with a width or height smaller than 7 pixels, and very big components with a width or height bigger than half the page. The small components can be considered to be noise or dots; the big components were caused by page border effects.

This resulted in a set of 86537 connected components. These were manually labeled into three categories: “normal”, “crossed-out” and “other”. The category “other” consisted of connected components that are noise or textual elements that could not be clearly categorized into one of the other categories. For examples and the number of components in each class, see Table 3.1. This categorization proved to be not straightforward during the manual labeling process, which indicates that the problem of detection of crossed-out text may actually be ill-posed.

3.1.2 Branching feature

When handwritten words are crossed-out, one or more strokes are written over existing strokes. The result can be seen as a high number of strokes with many crossings. The *branching* feature takes the number of crossings into account, where each crossing is called a branching point. To find the branching points of the connected components, they were first thinned using a recent method [52]. In the resulting skeleton, the branching points

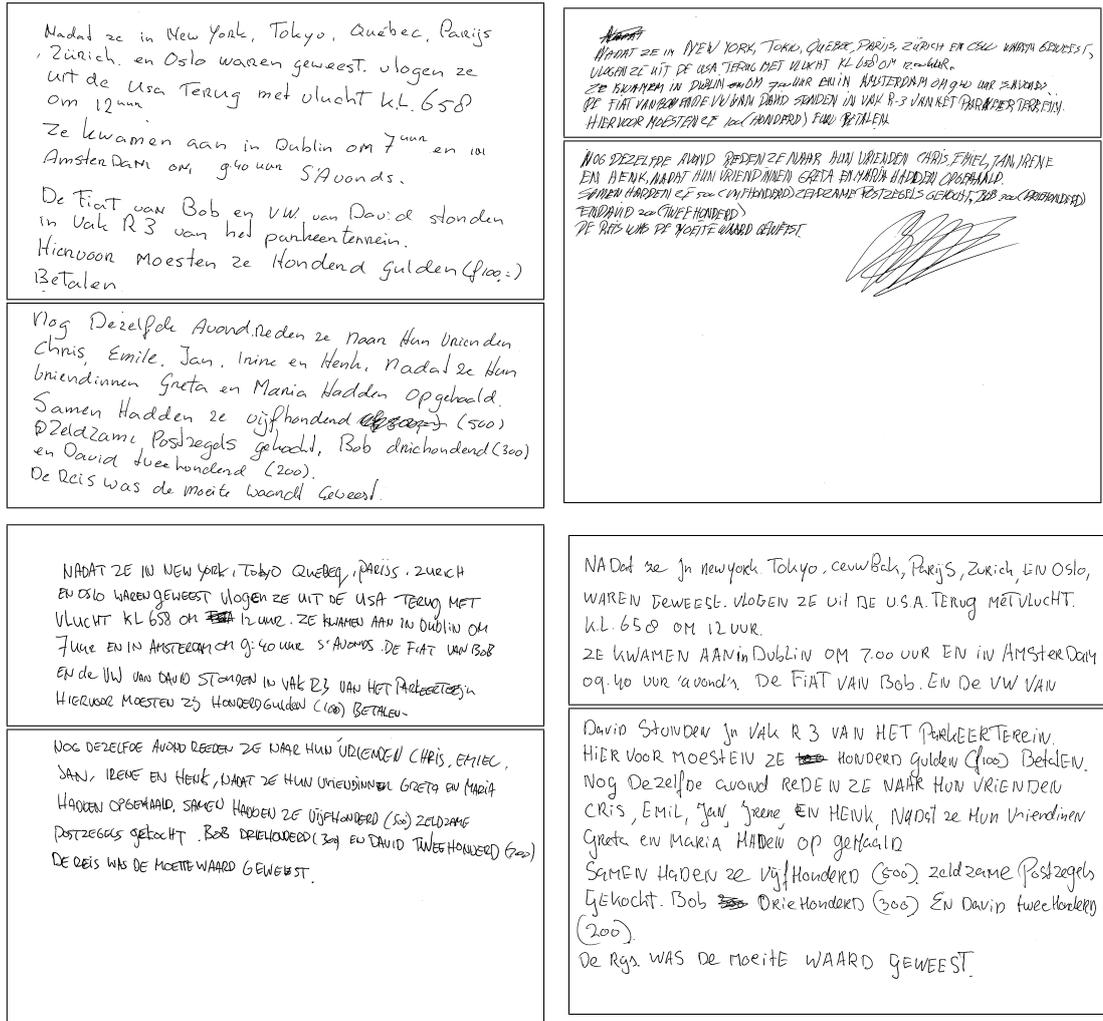


Figure 3.1: Example documents in the NFI dataset, each one cut in two parts.

Table 3.1: Classes of labeled connected components.

	normal	crossed-out	other
train set	43745	403	202
test set	41640	221	326
examples			

were identified as the black pixels that have more than two 8-connected black neighbors. This usually results in more than one branching point per actual crossing, but this is not important for the quality of the feature. The resulting number of branching points was normalized by dividing by the width of the connected component.

3.1.3 Size feature

The second feature exploits the fact that crossed-out text is usually a big object, because the crossing strokes add ink and usually connect individual letters or parts of a word. The size of the object is measured by counting the number of pixels in the skeleton image. Another approach would be to count the number of pixels in the original connected component; our tests indicated that that does not make much of a difference.

3.1.4 Training

The resulting branching feature and size feature were both normalized by dividing the values by the standard deviation within each page. This ensures that feature values

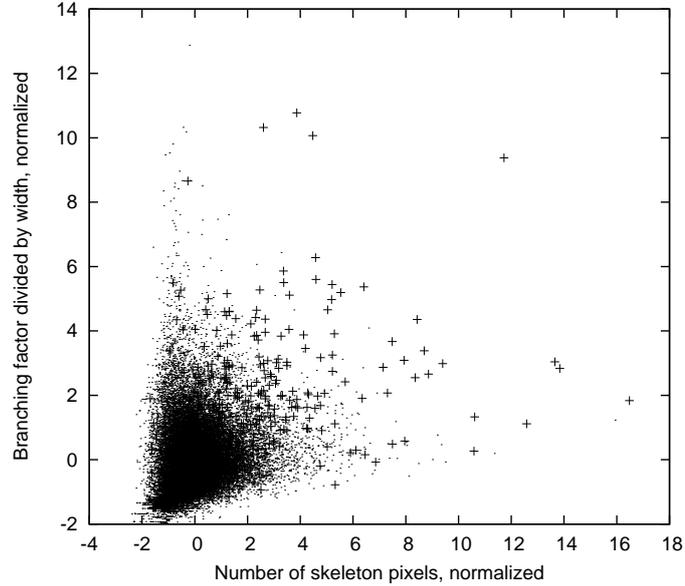


Figure 3.2: Feature values of connected components labeled as normal text (‘.’) and crossed-out text (‘+’) in page 1–125. Many pluses are inside the cloud of dots.

that are not common within a page can be identified as relevant extreme values. The final result is a set of labeled two-dimensional data. This data was split in two parts: a *train* set containing the connected components from page 1–125 and a *test* set involving page 126–250. The feature values of the classes “normal” and “crossed-out” in the train set are plotted in Figure 3.2. The feature values of the class “other” were not plotted, since they are not relevant for determining a decision boundary between the features of normal words and crossed-out words.

The figure shows that the classes “normal” and “crossed-out” mainly overlap, but not totally. It also shows that there are many more instances in the “normal” class. The classes can be separated up to a certain degree by a decision tree, which is a very simple classifier. Several other classifiers have been tried as well, including k-nearest neighbor, a linear support vector machine [54] and a neural network, but since their performance was not better and a decision tree is simple, the latter was used.

The decision tree was implemented by setting thresholds on each of the two normalized feature values: θ_s is the threshold on the size feature; θ_b is the threshold on the

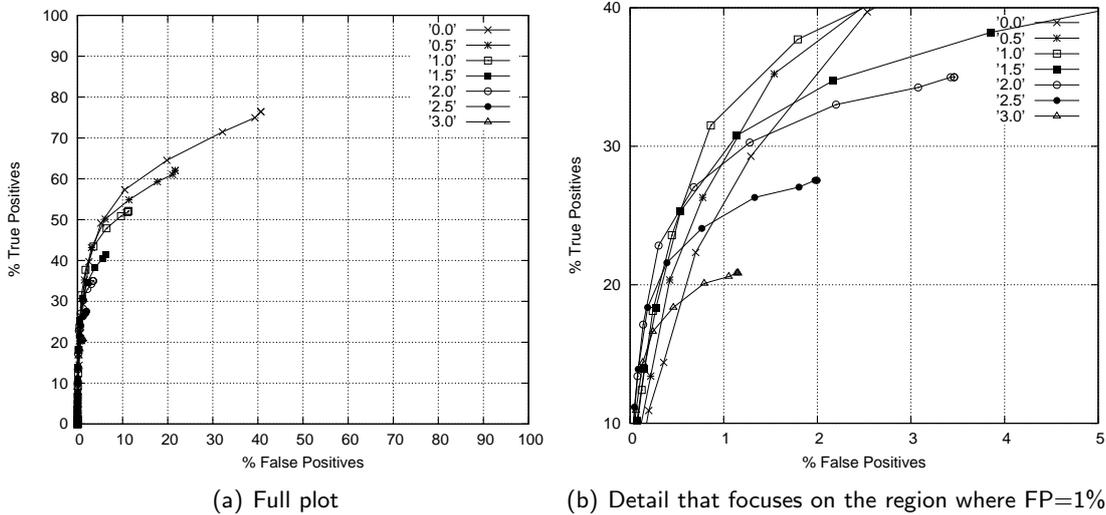


Figure 3.3: Multiple ROCs. Each ROC has a fixed θ_s ; see legend. Along each curve, θ_b varies. The curves do not reach the upper right corner because given the selections of θ_s , θ_b could not be positioned such that all data points would fall within the decision boundaries.

branching feature. Values above both of the thresholds were seen as positive examples, or crossed-out words. By positioning the thresholds, the ratio of true positives (TP) and true negatives (TN) can be balanced. This can be done using ROC plots which are created on the train set; see figure 3.3 and 3.4.

The optimal balance between TP and TN depends on the application, but at least it is desired that most of the normal text is not seen as crossed-out and thus remains intact. For illustrative purposes, it is now assumed that TN should be at least 99%. In other words, the number of false positives (FP) should be less than 1%. Using plots 3.3(b) and 3.4, it can be derived that usable thresholds would be $\theta_s = 1$; $\theta_b = 1.5$.

3.1.5 Results

The thresholds $\theta_s = 1$ and $\theta_b = 1.5$ were applied to the test set, which was completely fresh: it had not been used for training or testing before. The results are: $TP = 47.5\%$ and $TN = 99.1\%$. That means that almost half of the crossed-out words can be automatically removed while preserving 99% of the normal text. Figure 3.5 shows what the result would be on the images from Figure 3.1. Figure 3.6 shows the results using

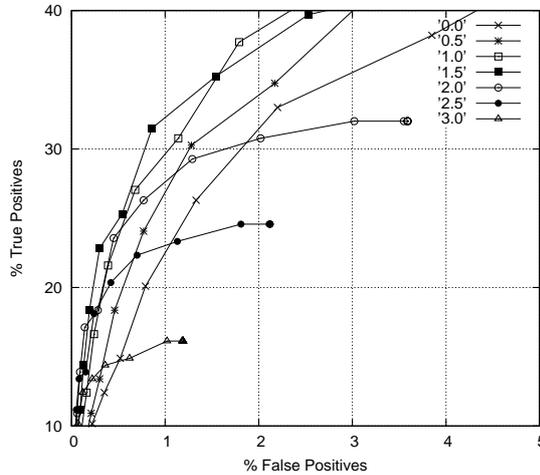


Figure 3.4: Same zoomed figure as Figure 3.3(b), but now every ROC curve has a fixed θ_b (see legend); along each ROC θ_s varies.

other thresholds. In these examples all of the crossed-out words have been successfully removed. It is clear that some of the components of normal words are removed as well, particularly bigger components, but most components of normal words remain.

To illustrate how the method scales to very big scratches, a small experiment was also performed on semi-artificial data: the four pages of which the cut versions are shown in Figure 3.1 have been overlaid with pages containing big scratches. For this test the condition that the crossed-out components should not be bigger than half the page was relaxed. Figure 3.7 shows what the result would be on such pages.

3.2 Application to writer verification

The proposed technique to automatically remove crossed-out words was applied in a writer verification experiment to determine whether it affects performance. In this experiment, the powerful *Hinge* [19] feature was used. This technique captures the orientation and curvature of the ink trace, encoded in a 528-dimensional feature vector. As a distance measure, the χ^2 measure [85] was used.

The experiment was performed as follows. First, like in [13], all pages of the NFI dataset were split in an upper part and a lower part. 1127 pages had to be discarded

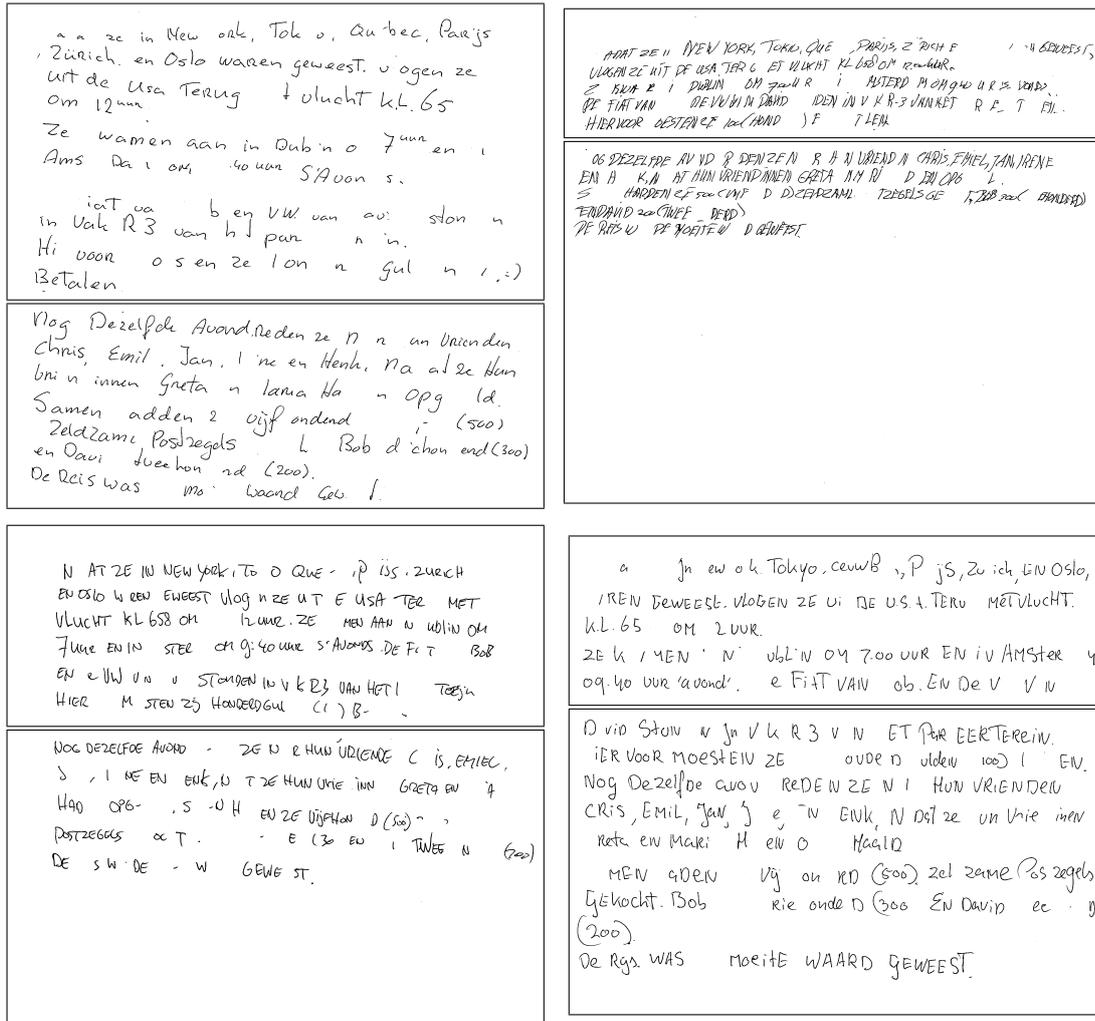


Figure 3.5: Images from Figure 3.1; thresholds $\theta_s = 1$ and $\theta_b = 1.5$ applied. All crossed-out components have been removed at the expense of some normal text.

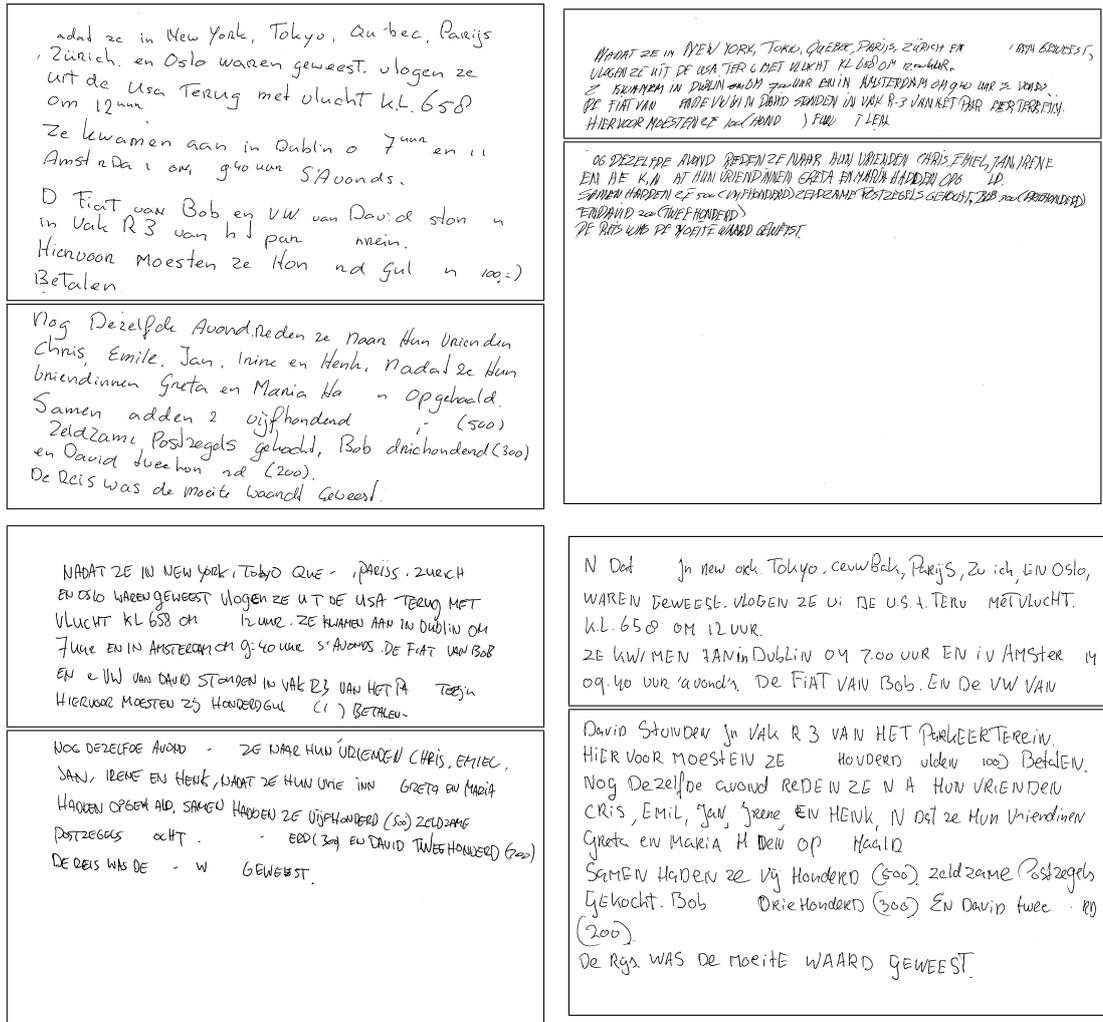


Figure 3.6: Images from Figure 3.1; thresholds $\theta_s = 2.5$ and $\theta_b = 2.5$ applied. With less strict thresholds, more of the normal text remains. Crossed-out text is also more likely to remain, but that does not occur in this example.

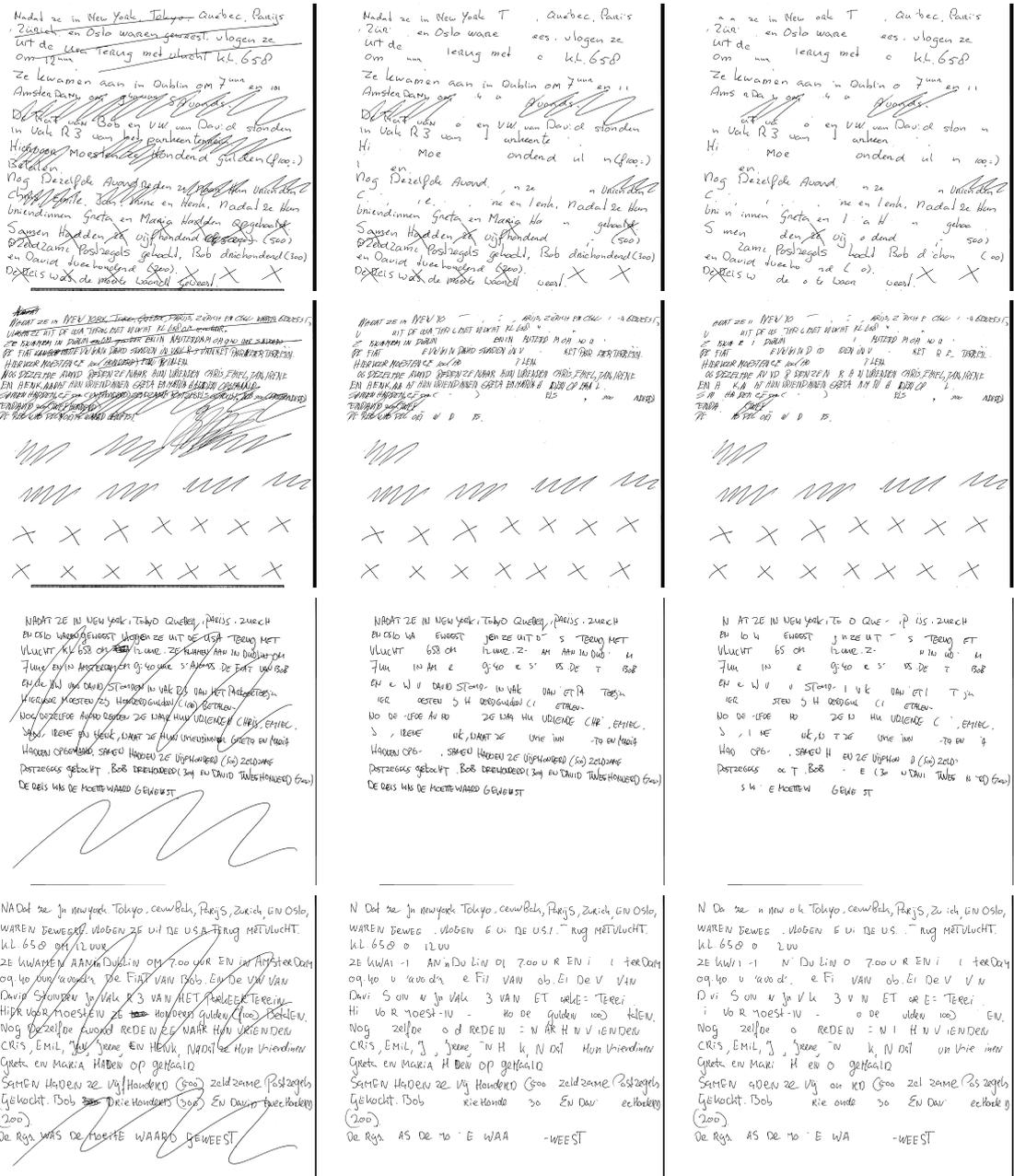


Figure 3.7: Result of removing big scratched-out elements. Column 1: original overlaid with scratches, column 2: result with $\theta_s = 2.5, \theta_b = 2.5$, column 3: result with $\theta_s = 1.0, \theta_b = 1.5$.

Table 3.2: Number of pages and writers in the dataset for verification and identification.

	train set	test set
original pages (writers)	250	3250
selected pages (writers)	181 (87)	2193 (988)
selected parts (writers)	362 (87)	4386 (988)

because curved or sloped baselines made a good cut impossible. After splitting, 4748 page parts remained, written by 1074 persons. The page parts were converted to monochrome using Otsu thresholding because this is required by the method to remove crossed-out words. The page parts were divided into two sets: a train set and a test set. The train set consisted of the parts of pages 1–250; the test set consisted of the parts from the other 3250 pages. See Table 3.2 for details. To compute the baseline performance, a verification threshold was learned from the training data by modeling the distances in the “same writer” and “different writer” classes using Parzen windowing. The threshold was selected such that the expected ratio of true positives (TP) was equal to the expected ratio of true negatives (TN); the equal-error rate (EER). This threshold was applied to the test set, yielding the experimental TP and TN.

To assess the effect of crossed-out text, the same steps were taken on the same page parts after cleaning by the crossed-out text removal method. Several values of θ_s and θ_b were tried while testing on the train set. The results are shown in Table 3.3. The table shows that the values for θ_s and θ_b have no big implications on writer verification, but the best result was used to determine the final values: $\theta_s = 2$ and $\theta_b = 2$.

3.2.1 Results

The values $\theta_s = 2$ and $\theta_b = 2$ were used to remove crossed-out text in the test set (4386 half pages) of the NFI dataset. Table 3.4 shows the result together with the baseline performance. It is clear that automatically removing crossed-out text using the proposed method has no substantial influence on writer verification performance.

3.3 Application to writer identification

The same kind of experiment was performed to determine the effect of automatically removing crossed-out text on writer identification. Writer identification means returning

Table 3.3: Writer verification results on 362 thresholded half pages extracted from the train set (the first 250 pages) of the NFI dataset; 87 writers.

θ_s	θ_b	TP	TN
1	1	78.1%	81.9%
1	1.5	79.3%	81.5%
1.5	1	78.7%	81.3%
1.5	1.5	79.7%	80.7%
1.5	2	80.4%	80.3%
2	1.5	80.3%	80.5%
2	2	79.0%	82.8%
2.5	2.5	79.8%	81.3%
3	3	80.5%	81.1%

Table 3.4: Verification results on 4386 thresholded half pages; 988 writers.

	TP	TN
Baseline	76.6%	84.0%
$\theta_s = 2, \theta_b = 2$	77.1%	83.6%

a *hit list*, a sorted list of documents of which the handwriting is similar to that of a questioned document. In this experiment, similarity was again determined by the *Hinge* feature and χ^2 -distance. The pages of the NFI dataset were split in parts, thresholded, and divided into a train set and a test set as described in section 3.2. The baseline performance was computed by treating every document in the test set as a questioned document, then yielding the hit list and finally counting how often a matching document appeared in the top-1, top-10 or top-100.

This was also done with pages of which crossed-out text was automatically removed using several values of θ_s and θ_b in the train set. The performance using these thresholds on the train set is shown in Table 3.5. Although the differences are again very small, the best selection of thresholds could be identified: $\theta_s = 1.5$ and $\theta_b = 2$.

3.3.1 Results

The thresholds $\theta_s = 1.5$ and $\theta_b = 2$ were applied to remove crossed-out text in the test set. On the resulting documents, writer identification was performed. The results are shown

Table 3.5: Identification results on 362 thresholded half pages extracted from the first 250 pages of NFI dataset; 87 writers.

θ_s	θ_b	Top-1	Top-10	Top-100
1	1	85.4%	95.6%	98.9%
1	1.5	87.9%	96.1%	99.2%
1.5	1	86.7%	95.0%	99.2%
1.5	1.5	87.3%	96.1%	99.2%
1.5	2	88.4%	95.0%	99.2%
2	1.5	87.0%	96.1%	99.2%
2	2	88.1%	95.6%	99.2%
2.5	2.5	88.1%	95.0%	98.9%
3	3	87.9%	95.0%	99.2%

Table 3.6: Identification results on 4386 thresholded half pages; 988 writers.

	Top-1	Top-10	Top-100
Baseline	76.5%	88.1%	95.0%
$\theta_s = 1.5, \theta_b = 2$	75.5%	87.7%	94.8%

in Table 3.6, together with the baseline performance. The table shows that automatically removing crossed-out text does not improve writer-identification performance.

3.4 Conclusion

In this chapter a simple method to identify and remove crossed-out text was presented. It can remove 47% of crossed-out text while 99% of the normal text is preserved. There is no important effect on writer verification or identification based on the *Hinge* feature [19]. This is an indication that the effect of crossed-out text on writer verification and identification may be overestimated. It has not yet been tested what fraction of the text could be crossed-out without disturbing automatic verification or identification but it is now clear that realistic documents containing crossed-out text can be admitted to systems for handwriting biometrics based on the *Hinge* feature.

Although this result suggests that removing moderate crossed-out text may not be worth the effort, there are options to make this statement more firm. It is conceiv-

able that the *Hinge* feature that was used for the writer verification and identification experiment, is just quite robust for crossed-out text. Therefore, other features should be tried for this too, for example the *Fraglets* feature (also called fCO3) [86]. It is also possible that the automatic method to remove crossed-out words does improve verification or identification performance, but at the same time reduces performance because also some good text is removed. Therefore the next step should be to improve the method to detect crossed-out words.

One way to improve the method could be to use textural features such as *Hinge* on the level of connected components. Alternate thinning methods could be tried for the branching feature because artefacts in the skeleton have a big influence on the performance. The method could also be adapted to work with grayscale images, which would make the method more versatile. That would slightly improve writer verification and identification performance, since the *Hinge* feature was designed for grayscale and performs slightly worse on black and white images. The best values for θ_s and θ_b could be determined in a more thorough way by using steepest descent or genetic algorithms. Furthermore, other classifiers can be tried, and line segmentation should be applied to disconnect connected components that are big because they consist of intersecting text from multiple text lines.

Chapter 4

Robustness for disguise by slant manipulation

A modified version of this chapter was previously published in *Pattern Recognition Letters*. [12]

A salient property of Western handwriting is *slant*: the dominant angle of near-straight downstrokes with respect to the horizontal. Slant is caused by the choice of pen grip and the relative contributions of wrist and finger movements. It has been modeled as the effect of locally using a single actuator (muscle) in a two-dimensional neuromuscular apparatus [31]. Slant seems to be a key feature for writer verification: it plays an important role in biometric systems, as it is a major constituent of angular features [16, 26, 63]. For example, the state-of-the-art *Hinge* feature [19] is based on angular frequencies; it is influenced by both curvature and slant. Furthermore, forensic document examiners and paleographers use slant as a discriminatory characteristic [22, 48]. These facts suggest that slant is a key feature for writer verification. However, it is not known to what extent slant contributes as an isolated factor to the performance of biometric systems for handwriting and its value may be overestimated.

In particular, slant is not a valuable feature in (possibly) disguised handwriting. In such a case, the handwriting was produced in a deliberately modified style, with the intention to avoid recognition of the writer's identity. Disguised handwriting is often used in threatening or stalking letters. In some cases, the mutilation of shapes successfully disturbs handwriting examination by forensic experts [36]. Moreover, disguised handwriting cannot be handled by state-of-the-art systems for handwriting biometrics: computational features that are invariant to disguise do not exist. This is one of the reasons why systems for handwriting biometrics are not fully suitable for application in the forensic domain yet.

A strategy to handle disguise is by applying an image operation to undo the effect of disguise, resulting in handwriting that is close to natural. This seems possible for the most frequently used kind of handwriting disguise: a change of slant. It is not surprising

that slant modification is the most frequently used kind of disguise [49, 58, 66, 68], since humans can easily modify the slant during writing, and the effect on the visual appearance is dramatic [58]. Therefore, an important step in making biometric systems robust for disguise is by correcting the slant. An obvious approach is to perform the correction by transforming the image with the *shear* operation, possibly resulting in the writer's natural handwriting.

The objective of this study is twofold. The first objective is to determine how much information about the writer's identity is contained in the slant characteristic of natural handwriting. This will be tested in the first experiment by eliminating the slant in natural handwriting (*slant elimination*) and measuring to what extent the performance of automatic writer verification degrades. This experiment contributes to the theoretical basis of computational writer features based on directionality, such as the Hinge feature [19]. The result may direct the design of future features.

The second objective is to determine the effectiveness of the shear transform in correcting handwriting disguised by slant change, when used as a preprocessing step before applying feature extraction methods such as *Hinge* [19] and *Fraglets* [86].

At the same time, the underlying question will be answered: to what extent is a change of slant during human production of handwriting functionally equivalent to a shear transform? Slant change may result in more than just a shear effect, since it requires a non-habitual movement of the finger-wrist system, which may affect curvature. It has been suggested that there must also be an effect on writing speed, pressure, connecting strokes, style, construction, and size [66]. Furthermore, disguised handwriting is less consistent [49, 58, 66]. In the second experiment, it will be quantitatively determined to what extent such other effects occur. This will be done by shearing slanted text back to the supposed writer's natural slant angle (*slant correction*), and determining the performance of writer verification using state-of-the-art features. This is a first step in designing new biometric systems that are robust to disguise. To the best of our knowledge, no similar experiment has been performed before.

The experiments will be performed on a newly created public dataset: the *TriGraph Slant Dataset*, containing both natural and slanted handwriting of 47 subjects. It is described into more detail in the next section. In sections 4.2–4.3, methods for slant estimation and feature extraction are described; these are preliminaries for the experiments. Experiment 1 will show that slant is not as informative as is usually assumed; it is described in section 4.4. Experiment 2 will show that deliberate slant change is not

equal to a simple shear transform; it is described in section 4.5. Section 4.6 summarizes the conclusions.

4.1 TriGraph Slant Dataset

A new dataset was created, the *TriGraph Slant Dataset*: a unique collection of clean, deliberately slanted handwriting in conjunction with each writer's natural handwriting. It consists of 188 scanned images of handwritten pages, written by 47 untrained Dutch subjects, aged 27 on average. This dataset is relatively small compared to other datasets such as Firemaker [88] (251 writers), IAM [65] (657) and Srihari's dataset [94] (1500). However, the dataset proved to be large enough to analyze the effect of slant. It can be obtained from <http://www.unipen.org/trigraphslant.html>. The dataset can be used for both handwriting comparison experiments and handwriting recognition experiments.

The dataset was assembled as follows. The subjects were provided two printed Dutch texts, *text A* and *text B*. Both texts contained approximately 200 characters, including all lowercase letters and many capitals; the distribution of the letters among the two texts was similar. Each subject wrote four pages, such as the one shown in Fig. 4.1, following these instructions:

1. [AN] Copy text A in your natural handwriting.
2. [BN] Copy text B in your natural handwriting.
3. [BL] Copy text B and slant your handwriting to the *left* as much as possible.
4. [BR] Copy text B and slant your handwriting to the *right* as much as possible.

See Fig. 4.2 for a close look at fragments of the four pages written by one writer. The codes AN, BN, BL and BR refer to subsets into which the collected pages of the writers were subdivided. *AN* represents a collection of authentic documents; *BN*, *BL* and *BR* can be seen as collections of questioned documents. To avoid structural effects of fatigue, the order of item 3 and 4 was randomized; half of the subjects wrote the BR page before the BL page.

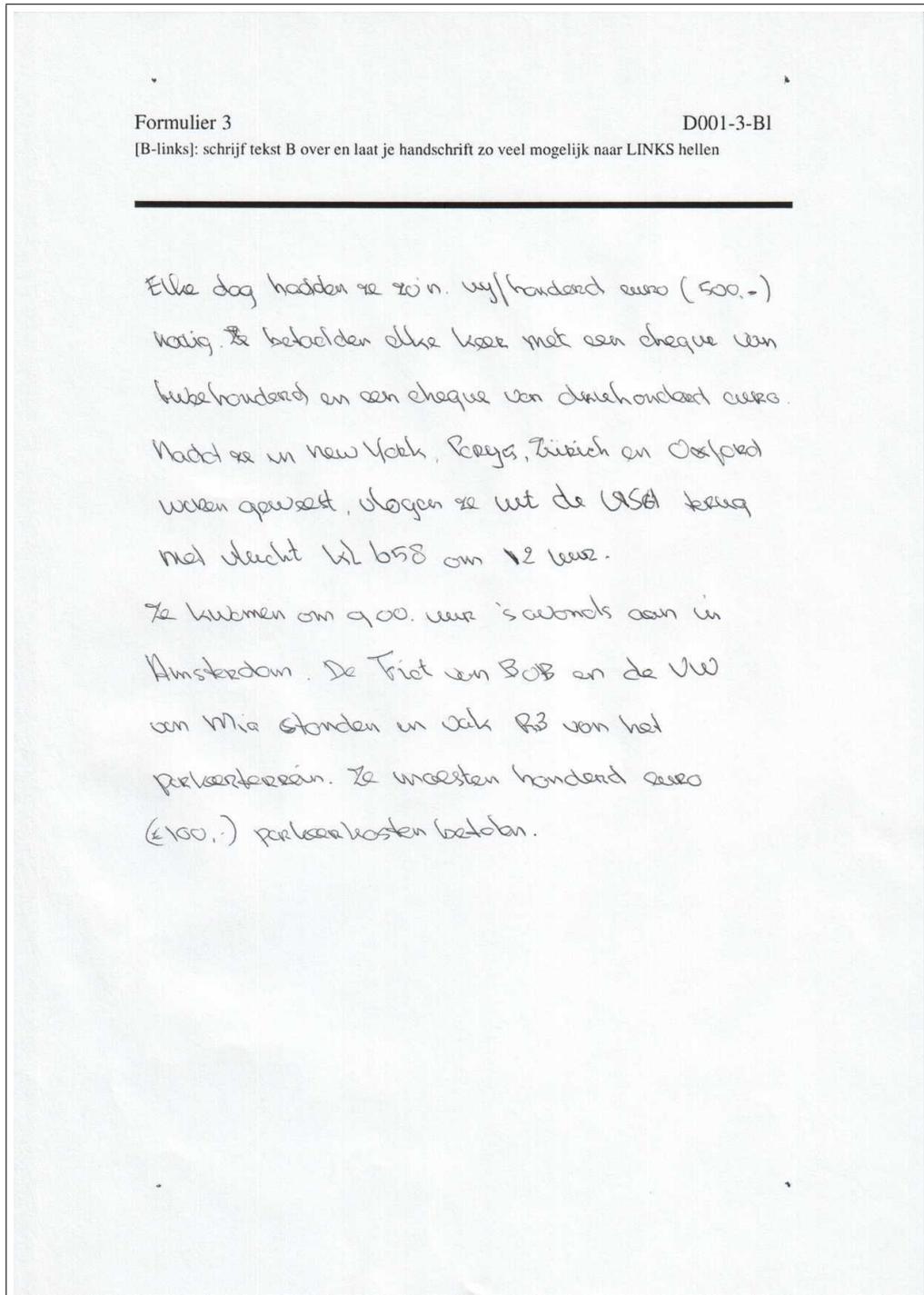
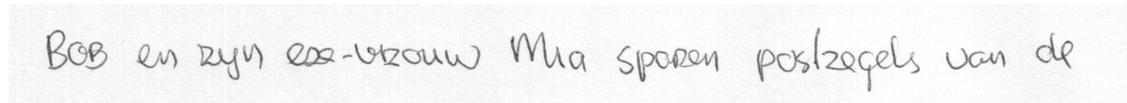
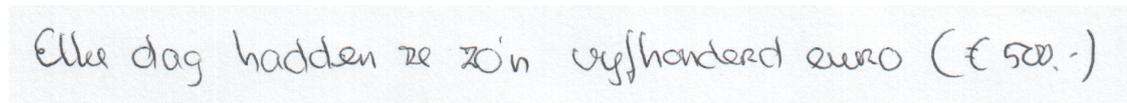


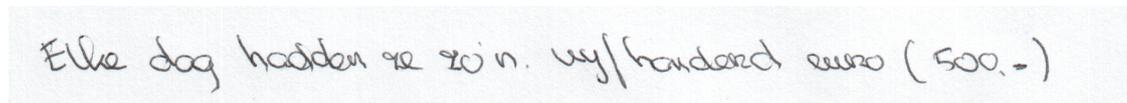
Figure 4.1: Example page from the TriGraph Slant Dataset: page 3 of writer D001. It contains text B, slanted to the left (BL).



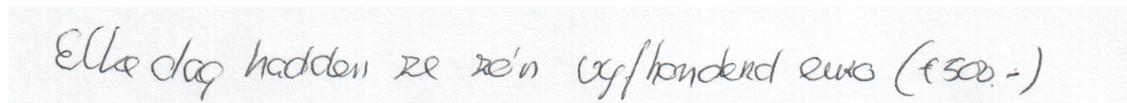
(a) Text A, natural handwriting (AN)



(b) Text B, natural handwriting (BN)



(c) Text B, slanted to the left (BL)



(d) Text B, slanted to the right (BR)

Figure 4.2: Example of the four pages written by writer D001, the first subject in the dataset. For each page only the first line is shown, manually cut out for the purpose of illustration.

4.2 Slant estimation

Since Experiment 1 and 2 both require a reliable technique to estimate slant, a limited comparison of techniques is included here. A variety of slant estimation methods exists, based on different definitions of ‘slant’. For example, it has been defined as the average direction of near-straight or long downstrokes [64], or “the angle between the vertical direction and the direction of the strokes that, in an ideal model of handwriting, are supposed to be vertical” [103]. The methods can be roughly subdivided into two general approaches which could be called the *angle-frequency approach* (AF) and the *repeated-shearing approach* (RS). In AF, which is most popular [57], downstrokes are first located based on a criterion such as a minimal vertical extent or velocity. Next, the angle of the local ink direction is measured at those locations; the resulting angles are agglomerated in a histogram. From this histogram, the slant angle is determined. This general

algorithm is shown in Algorithm 2. Variations include computing an edge-direction histogram and finding the maximum or mode in it [16] or the peak that is closest to 90° [26]. Another variation computes the average angle in rectangular sub-areas containing vertical structures [9].

Algorithm 2 AF: Compute the slant angle using the angle-frequency approach. INPUT: image I . OUTPUT: slant angle a .

```

 $h \leftarrow \text{empty\_histogram}()$ 
for all pixel  $p$  in  $I$  do
  if  $\text{criterion}(p)$  then
     $a_p \leftarrow \text{local\_angle}(p)$ 
     $h.\text{add}(a_p)$ 
  end if
end for
return  $\text{best\_freq}(h)$  {maximum or mode}

```

RS is based on the assumption that the projection of dark pixels contains maximal peaks when projected along an axis parallel to the slant angle. The basic principle is to repeatedly shear images of individual text lines, varying the shear angle, and optimizing a criterion on the vertical projection of dark pixels [57, 103]. Such a criterion involves the maximization of peaks in the projection. The range of shear angles extends to hypothetical extreme slant angles such as $30^\circ \cdots 150^\circ$. This approach is shown in Algorithm 3. It requires that the text has been split into individual text lines, which can be done by using smoothed projection histograms if the text lines do not overlap much. Obviously, RS is much slower than AF, but that is of no importance for this experiment.

To determine a usable technique for slant estimation, a limited comparison of implementations of AF and RS was performed. AF was implemented by calculating the angle at near-straight parts of the ink boundary and yielding the mode of the smoothed histogram; RS was implemented with an algorithm that maximized the density of the 10 highest peaks in the vertical projection histogram of each text line. For testing purposes, ground truth data for the first 24 pages of the dataset was generated by averaging 10 manually measured downstroke angles per page; these were compared to the angle estimations of the automatic methods. Table 4.1 shows that the angles computed with RS are closer to the ground truth, although this difference is not significant at the 5% confidence level ($t = -1.12$; determined using the one-sample t test: $t = \frac{\bar{x}}{s/\sqrt{n}}$, where $n = 24$,

Algorithm 3 RS: Compute the slant angle using the repeated-shearing approach. INPUT: image I . OUTPUT: slant angle a .

```

a ← empty_list()
for all textline  $L_i$  in  $I$  do
   $s^* = 0$ 
  for all  $a$  in  $30^\circ \dots 150^\circ$  do
     $p \leftarrow \text{ver\_project\_ink}(\text{shear}(L, a))$ 
     $s \leftarrow \text{score}(p)$ 
    if  $s > s^*$  then
       $s^* \leftarrow s$ 
       $a_i \leftarrow a$ 
    end if
  end for
end for
return median(a)

```

\bar{x} is the average of x_i , which are the differences of squared errors, and s is the standard deviation of x .) Still, in the following experiments, RS was used to automatically determine the slant angle.

4.3 Feature extraction and comparison

The effect of slant on features of handwriting was evaluated using three well-performing automatic feature extraction methods: *Directions* [16], *Fraglets* [86] and *Hinge* [19]. See Section 1.8 for a description of these methods.

The distance $d(\cdot, \cdot)$ between any two feature vectors was computed with the χ^2 distance measure [25] (see Section 1.9). It was used for this experiment because it is specifically effective on feature vectors that represent a probability distribution [85], such as the three features described above. In the following, $d(P, Q)$ will denote the distance between the feature vectors of the images P and Q .

Table 4.1: Slant angle in the first 24 pages determined using three methods: manual (Man), angle frequencies (AF), and repeated shearing (RS). The manual measurements represent averages of at least 10 measurements per page. RS yielded the lowest RMSE (root of the mean of squared errors) with respect to the manually determined angle.

Page	Man	AF	RS	Page	Man	AF	RS
D001-1-AN	99	96	97	D004-1-AN	95	95	96
D001-2-BN	103	101	100	D004-2-BN	95	94	95
D001-3-BL	122	122	121	D004-3-BR	76	74	76
D001-4-BR	68	68	68	D004-4-BL	112	112	111
D002-1-AN	75	76	77	D005-1-AN	72	75	72
D002-2-BN	67	72	70	D005-2-BN	73	73	73
D002-3-BL	97	97	97	D005-3-BR	64	35	56
D002-4-BR	54	44	50	D005-4-BL	107	102	104
D003-1-AN	79	78	79	D006-1-AN	82	80	78
D003-2-BN	71	76	78	D006-2-BN	76	76	78
D003-3-BR	59	59	58	D006-3-BR	54	50	53
D003-4-BL	97	99	100	D006-4-BL	98	99	98
RMSE						7	3

4.4 Experiment 1: information in slant

The first experiment focused on determining how informative the slant value in natural handwriting is. This was determined by computing the performance of writer verification on unmodified handwriting (denoted 'AN vs BN'), and comparing it to the performance on handwriting of which the slant was eliminated ('AN vs BN, elim.'). This is explained in the next subsections.

4.4.1 Unmodified handwriting

For the performance on unmodified handwriting, documents were drawn from the dataset in pairs: one was drawn from the AN subset, the other from BN. After computing the distance between their feature vectors using one of the feature extraction methods, two cases were distinguished: the documents were written by the same person, or by different persons. The same-writer distances formed a multiset of distances D_s and the different-writer distances formed the multiset D_d . Thus, D_s and D_d are defined as follows:

$$D_s = \{\forall i :: d(AN_i, BN_i)\} \quad (4.1)$$

$$D_d = \{\forall i, j : i \neq j : d(AN_i, BN_j)\} \quad (4.2)$$

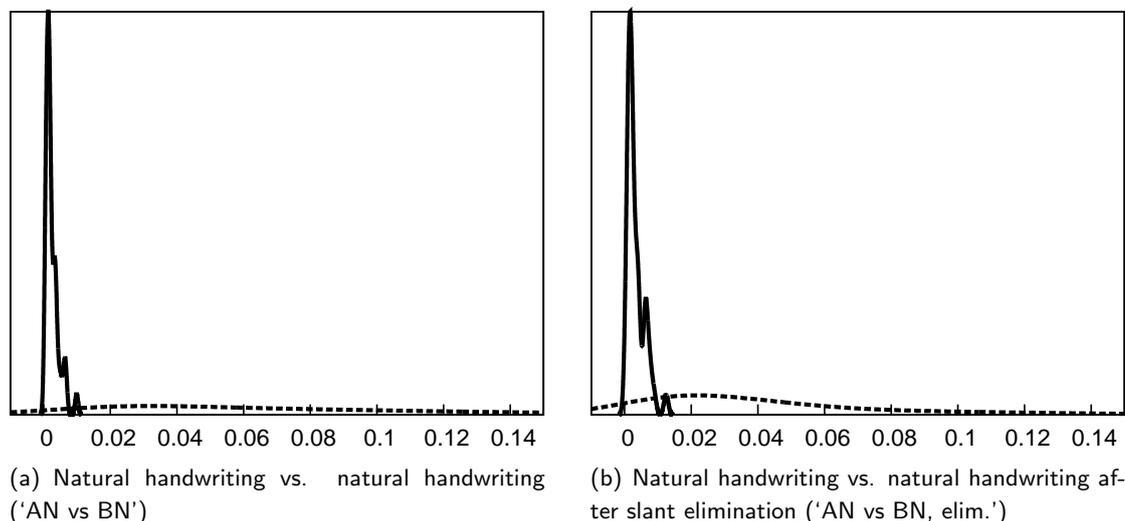


Figure 4.3: Densities of distances between documents with natural handwriting, based on the *Directions* feature. The continuous curve represents D_s , distances between documents written by the same writer; the dashed curve represents D_d , distances between documents by different writers. The two classes can be separated quite easily, either without (a) or with (b) slant elimination. The densities were rendered smooth using Parzen windowing.

Fig. 4.3 shows an example of D_s and D_d , visualized as densities.

Based on these multisets of distances, a writer verification classifier was implemented by setting a threshold. The position of the threshold determines the trade-off between Type-I error rate (false accept rate; falsely assigning two pages to the same writer) and the Type-II error rate (false reject rate; falsely assigning two pages to different writers). It was put on the position where the Type-I error rate was equal to the Type-II error rate, or the equal-error rate (EER). The performance was estimated by $100\% \cdot (1 - \text{EER})$.

4.4.2 Slant-eliminated handwriting

To determine the contribution of slant to writer-specific features, the experiment was repeated after applying *slant elimination* on all pages: shearing the text such that its apparent slant becomes equal to 90° . It is a standard step in handwriting recognition systems, which use it to optimize recognition of the textual contents [7, 57]. It has also been called “deslanting”, “slant removal”, and “slant correction”. In this way, the

Table 4.2: Decrease of performance after slant elimination. Writer verification performance for three different features as the percentage of correct classifications. The value of slant seems very limited: only the Directions feature suffered somewhat, while the performances of the other features did not decrease significantly.

Subsets	Directions	Fraglets	Hinge
AN vs BN	97	100	99
AN vs BN, elim.	92	99	98

absolute slant information is lost. Slant elimination $E(\cdot)$ can be expressed as follows:

$$E(P) = \text{shear}(P, 90^\circ - a(P)) \quad (4.3)$$

where $\text{shear}(P, \alpha)$ is the image processing operation that shears a page image P with α degrees and $a(P)$ denotes the slant angle of the handwriting in P . $a(P)$ was estimated by the RS algorithm described in section 4.2. Figure 4.4 shows partial examples of slant-eliminated pages. Using slant elimination, the new definition of D_s and D_d is:

$$D_s = \{\forall i :: d(E(AN_i), E(BN_i))\} \quad (4.4)$$

$$D_d = \{\forall i, j : i \neq j : d(E(AN_i), E(BN_j))\} \quad (4.5)$$

The resulting performance $100\% \cdot (1 - EER)$ will be denoted ‘AN vs BN, elim.’

Notice that the pages were sheared entirely. An alternative option is to shear text lines or words individually, but this is less reliable and breaks ink traces at region boundaries. It is also possible to eliminate slant non-uniformly within each text line, but this seems to add little or no improvement, despite the added complexity [7]. The page-level approach is simple, fast and keeps the signal structurally intact.

4.4.3 Results

The results of Experiment 1 are shown in Table 4.2. The first row shows performances of the three features on natural handwriting, ‘AN vs BN’. The performances of all features are 97% or higher, which confirms the discriminative power of these features. These performances have an optimistic bias since they were not obtained on a separate test set, but the absolute performance is not relevant here. The second line of the table, where slant elimination was applied, shows a slight decrease of performance: for the

AN'_1 Bob en zijn ex-vrouw Mia sparen postzegels van de
 BN'_1 Elke dag hadden ze zo'n vijfhonderd euro (€500,-)
 BL'_1 Elke dag hadden ze zo'n vijfhonderd euro (500,-)
 BR'_1 Elke dag hadden ze zo'n vijfhonderd euro (€500,-)

(a) Writer D001

AN'_2 Bob en zijn ex-vrouw Mia sparen postzegels van
 BN'_2 Elke dag hadden ze zo'n vijfhonderd euro (€500,-)
 BL'_2 Elke dag hadden ze zo'n vijfhonderd euro (€500) nodig. Ze hebben
 BR'_2 Elke dag hadden ze zo'n vijfhonderd euro (€500) nodig.

(b) Writer D002

AN'_3 Bob en zijn ex-vrouw Mia sparen postzegels
 BN'_3 Elke dag hadden ze zo'n vijfhonderd euro
 BL'_3 Elke dag hadden ze zo'n vijfhonderd euro (€500,-)
 BR'_3 Elke dag hadden ze zo'n vijfhonderd euro (€500,-)

(c) Writer D003

Figure 4.4: Slant elimination in four pages by three writers; only the first text line of each page is shown. For each writer, four example text lines are shown: AN'_i , BN'_i , BL'_i and BR'_i . These are transformed images, obtained by executing slant elimination on the original pages. The originals written by subject D001 are shown in Fig. 4.2.

Table 4.3: Comparison of writer verification classifications using the *Directions* feature on natural handwriting in two conditions: with and without slant elimination. This contingency table elaborates on the found differences of performance shown in Table 4.2. The off-diagonal entries show the number of unequal classifications in the two conditions. In this case, the difference of the classifications in the two conditions is significant ($p \ll 0.001$, χ^2 test of significance on this table); the *Directions* feature significantly performs slightly worse after slant elimination. The other features do not show a significant effect.

		slant eliminated (‘AN vs BN, elim.’)	
		correct	wrong
original (‘AN vs BN’)	correct	2015	145
	wrong	6	43

Directions feature, it decreased with 5 percentage points. This is significant ($p \ll 0.001$, determined using the χ^2 test on the contingency table shown in Table 4.3) but small. The performance of the other features decreased with only 1 percentage point, a non-significant difference.

4.4.4 Discussion

Contrary to common assumptions, the results of Experiment 1 show that slant is not an important aspect of writer-specific features. Slant elimination did not significantly affect *Fraglets* and *Hinge*, while the performance of *Directions* decreased with only 5%. The latter relies heavily on angular information; it is a distribution of angles in which the position of the mode (peak) indicates the average slant angle. The small decrease in its performance shows that the shape of the distribution is more important than its position. This also indicates that the shear transform can be used to counter slant changes in disguised handwriting. However, the next experiment shows that this is not completely effective.

4.5 Experiment 2: is deliberate slant change an affine transform?

The aim of the second experiment is to determine whether deliberate slant change is functionally equivalent to a simple affine transform: *shear*. In this experiment, apart from

natural handwriting (BN), the disguised handwriting (BL, BR) from the dataset was included as well. Thus the experiment was performed three times, each time comparing documents from AN with those from either BN, BL or BR. Furthermore, in an attempt to restore the handwriting, *slant correction* was used instead of slant elimination. This is explained into more detail in the next subsections.

4.5.1 Unmodified handwriting

The baseline performance for this experiment was computed similar to Experiment 1 (see 4.4.1). In this case, three baseline performances were computed: ‘AN vs BN’, ‘AN vs BL’ and ‘AN vs BR’.

4.5.2 Slant-corrected handwriting

If deliberate slant change is functionally a shear transform, then the manipulated handwriting could be transformed back to natural handwriting by shearing it such that the apparent slant becomes equal to the writer’s natural slant angle. We define *slant correction* $C(\cdot, \cdot)$ as follows:

$$C(P, Q) = \text{shear}(P, a(Q) - a(P)) \quad (4.6)$$

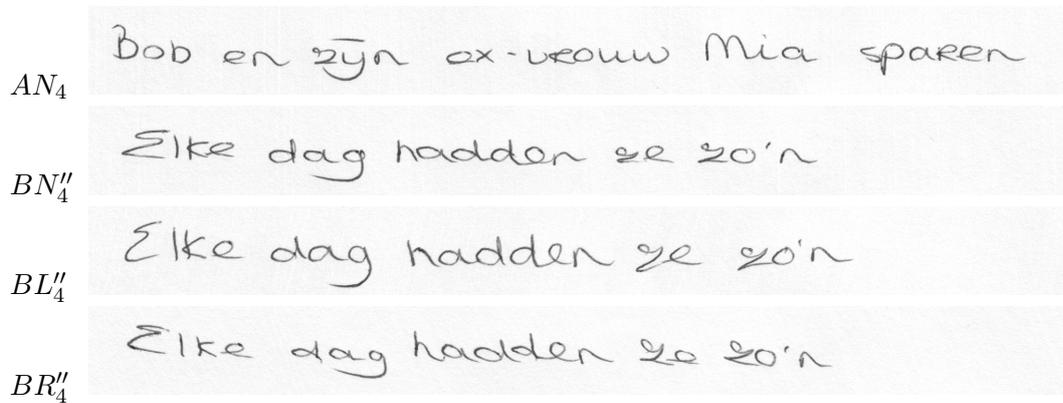
It is similar to slant elimination, but this approach attempts to restore the original handwriting. It can be used if the handwriting in Q is known to be genuine but P may have been disguised by slant manipulation. This is illustrated in Figure 4.5, which shows fragments of pages after slant correction.

In this condition, the distances were computed as follows:

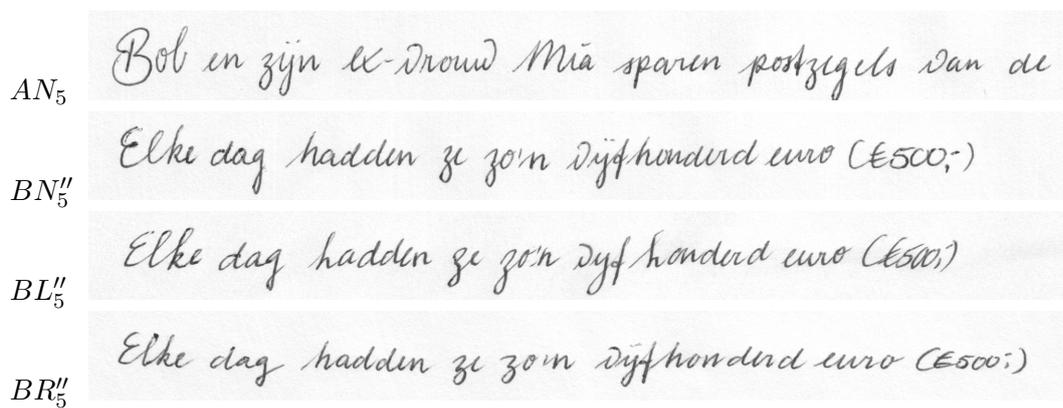
$$D_s = \{\forall i :: d(AN_i, C(B_i, AN_i))\} \quad (4.7)$$

$$D_d = \{\forall i, j : i \neq j : d(AN_i, C(B_j, AN_i))\} \quad (4.8)$$

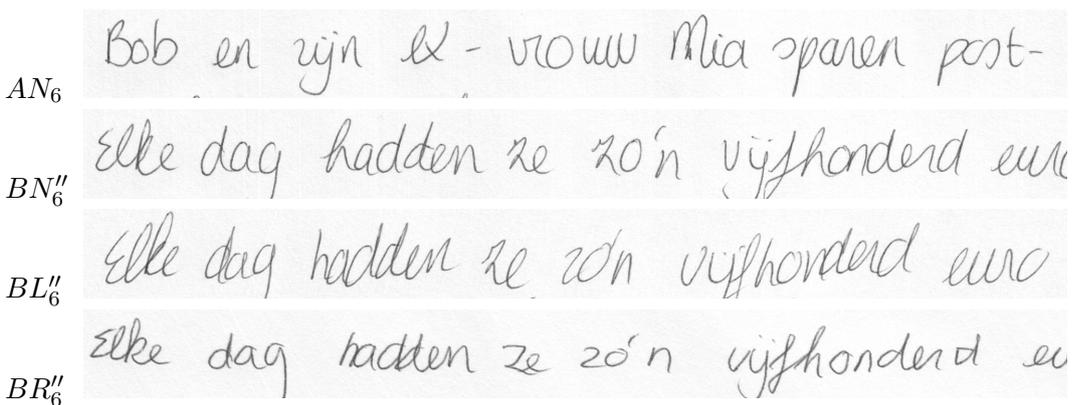
where B is either BN, BL or BR. Examples of the distributions of D_s and D_d are visualized in Fig. 4.6. The resulting performances are denoted ‘AN vs BN, corr.’, ‘AN vs BL, corr.’ and ‘AN vs BR, corr.’ If the hypothesis is true, then these performances should be equal to the performance on natural handwriting (‘AN vs BN’).



(a) Writer D004



(b) Writer D005



(c) Writer D006

Figure 4.5: Slant correction. For each of three writers, four example text lines are shown: AN_i , BN_i'' , BL_i'' and BR_i'' . The latter three are transformed images, obtained by slant-correcting BN, BL and BR, respectively, with their slant matching that of the first line (AN).

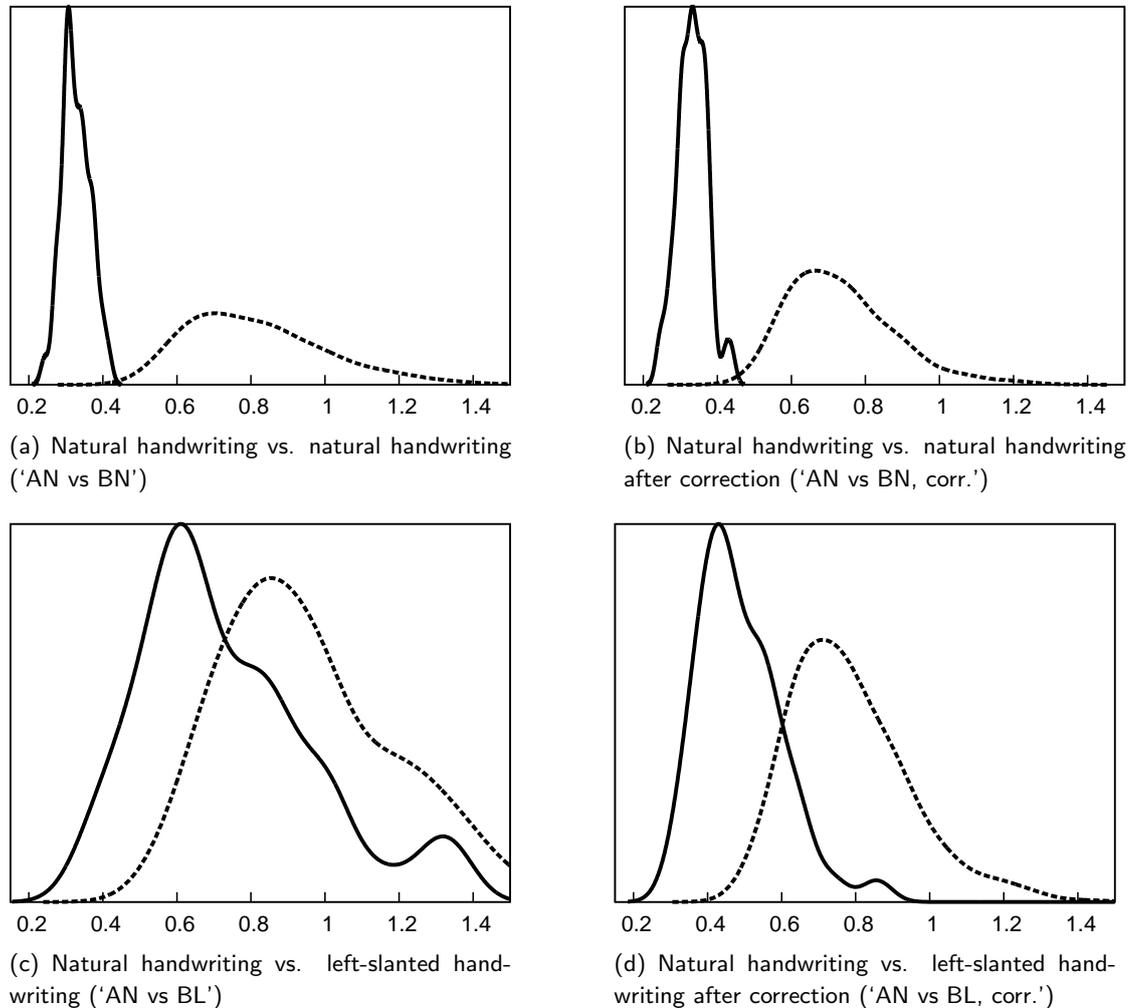


Figure 4.6: Distribution of distances between documents with natural and slanted handwriting, based on the *Fraglets* feature. The continuous curve represents D_s , distances between documents written by the same writer; the dashed curve represents D_d , distances between documents by different writers. In the case of unmodified natural handwriting (a), the two classes can be separated easily. This changes when the writers disguise their handwriting by a slant change to the left (c). This is partly solved by slant correction (d), which does little harm to natural handwriting (b). For visualization purposes, the distributions were rendered smooth using Parzen windowing.

Table 4.4: Quantitative effect of slant manipulation on three writer-specific features. Writer verification performance on original images (first three lines) and slant-corrected images (last three lines). Percentages of correct classifications.

Subsets	Directions	Fraglets	Hinge
AN vs BN	97	100	99
AN vs BL	57	68	62
AN vs BR	53	66	57
AN vs BN, corr.	92	99	99
AN vs BL, corr.	64	86	75
AN vs BR, corr.	66	90	72

4.5.3 Results

The results of Experiment 2 are shown in Table 4.4. The first row is a copy of the first row in Table 4.2. The best writer verification is obtained on natural handwriting (‘AN vs BN’): for the tested features, performances are in the range 97%–100%. The performances on natural vs. slanted handwriting (‘AN vs BL’ and ‘AN vs BR’) are obviously lower: a drop to 53%–68%. These figures are all significantly differing from the corresponding performances on natural handwriting (χ^2 test, $p \ll 0.001$). “Correcting” the slant in *natural* text, which should not need correction (‘AN vs BN, corr’), had only little negative influence on writer verification: Hinge and Fraglets remained stable, but the performance of the Directions feature dropped from 97% to 92% because it relies more on absolute slant information. But the most important result is that the performance on slant-corrected, originally slanted handwriting (‘AN vs BL, corr’ and ‘AN vs BR, corr’) is significantly *lower* than the performance on natural handwriting ($p \ll 0.001$); the figures are only in the range 64%–90%.

The reported performances focus on the equal-error rate (EER), where the Type-I and Type-II error rates are equal. To explore the trade-off between the Type-I and Type-II error into more detail, Figure 4.7 shows the errors as a result of varying the classification threshold. In the graphs, the EER values can be found at the intersection of any curve with the diagonal (shown as a dashed gray line).

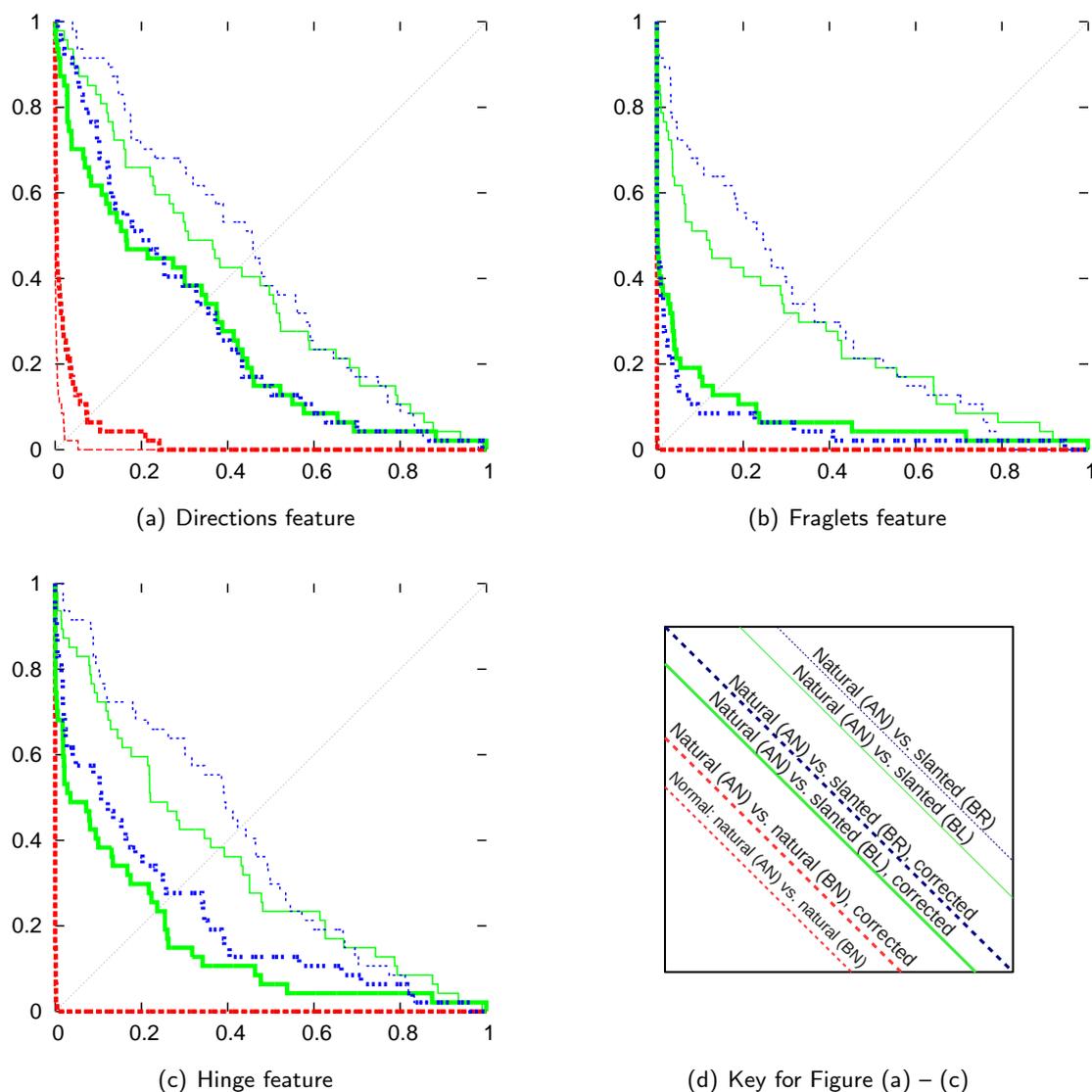


Figure 4.7: Effectiveness of correcting deliberate slant change in handwriting; the key is in Figure 4.7(d). Verification error plots for three features. The curves show the trade-off of the Type-I error rate on the horizontal axis and Type-II error on the vertical axis. The performances reported in Table 4.4 appear as the intersections of the curves with the diagonals. The more a curve approaches the lower left corner, the better the performance. Since *Fraglets* and *Hinge* perform 99%–100% on natural handwriting ('AN vs BN') and on natural handwriting after correction ('AN vs BN, corr'), the corresponding curves are in the lower-left corner of the figure and cannot be discerned. The graphs show a consistent increase of performance after correcting slant, but it does not get as good as on natural handwriting.

4.5.4 Discussion

The results of experiment 2 show that handwriting disguise by changing slant lowers writer verification performance, if no correction is applied. This is obvious, but the effect is not the same on all tested features: the *Fraglets* feature seems to be most resilient against disguise by slant manipulation. After automatically correcting the slant with a shear operator, the performance improved for all tested features, which means that the distortion caused by slant change can be partly undone by slant correction. This result suggests to apply slant correction always before handwriting comparison takes place as a preprocessing step. The best performing feature after slant correction was *Fraglets*. But in spite of the improved performance, correcting the slant in slanted handwriting did not restore writer verification performance *fully* for any feature. This means that using the shear transform is not a complete solution against the problem of slant manipulation in disguised handwriting. In other words, slant correction did not result in the original handwriting, and deliberate slant change is *not* functionally equivalent to the affine transform which is the shear operation.

This raises the question what else changes in the handwriting during slant change. We know that disguise is usually inconsistent [49, 58, 66], thus a greater variation of slant is expected. This is confirmed by the observation that *Hinge* and particularly *Directions*, which heavily rely on slant information, suffered most from the slant manipulation. Furthermore, it can be expected that the non-habitual movement of the finger-wrist motor system introduces artifacts. The dataset is not extensive enough for a conclusive analysis, therefore, we suggest a follow-up study with more data, in which the corrected handwriting is thoroughly analyzed by forensic experts.

A further improvement would be to automatically *detect* disguise and decide if slant correction should be applied. Forensic experts try to detect disguise based on experience, but to the best of our knowledge, automatic methods to detect disguise do not exist yet. In one study, a method has been devised to approach the related problem of forgery detection [24]. It exploits the fact that forged handwriting is often less fluently written; this principle may be applicable to disguise detection as well. In addition, we suggest to exploit the aspect of inconsistency, as it is known to be an important indicator.

Another challenge for the future is to develop features that are invariant to disguise. A new direction might be to make features that determine the way the letters are constructed, mimicking an approach used by forensic document examiners.

4.6 Conclusion

Slant is a salient feature of handwriting and it is an important factor of several statistical features, but as an isolated factor, it is not essential for good writer verification performance. It is not as informative for handwriting comparison as is usually assumed. This was found in a series of writer verification experiments using three state-of-the-art statistical features: *Directions*, *Fraglets*, and *Hinge*. Removing the absolute slant lowered writer verification performance by only 1–5 percentage points.

In disguised handwriting, slant is not valuable, but possibly deceptive because it is subject to deliberate modification. When a non-habitual slant angle is applied during writing, performance of the features obviously decreases. However, correcting the handwriting by shearing it to obtain a natural slant value did not restore performance fully. Thus, disguise by slant change has more effect on the handwriting than just a shear effect, and the shear transform is not a complete solution against it. However, it is useful as a partial solution: slant correction did improve performance on disguised handwriting. Since slant is not an important aspect of handwriting, all possibly disguised handwriting can and should be sheared to match the specimen handwriting. This may be an essential step in biometric systems and in manual comparisons as well.

Part II

New methods for robust and applicable handwriting biometrics

Knowing the boundaries of acceptable input is a first step toward robust and applicable systems. In the part I, the robustness of methods based on statistical pattern recognition was analyzed with respect to three performance influencing factors: text scarcity, crossed-out text and disguise were considered. It was shown that robust classification is possible if at least 100 handwritten characters are present in the input documents. Furthermore, a realistic fraction of crossed-out text can be admitted. However, classification of (possibly) disguised handwriting cannot be called robust yet, even if the disguise only constitutes a change of slant. In addition, chapters 3 and 4 also introduced methods to counter crossed-out text and disguise. It was shown that a robust system for handwriting biometrics should include a method for slant correction if disguised handwriting is expected as input.

In part II, a step further will be taken: the introduction of new methods to make handwriting biometrics more robust and applicable. This is the second objective of this dissertation. A robust system does not just rely on a single classification method, but contains several methods that are selected when needed [42, 43, 79]. In Chapter 5, a new and powerful feature for writer identification will be introduced. This adds another powerful method to the set of methods any robust system can select. The feature is inspired by modern paleography and exploits information in the angle and width of ink traces.

Apart from robustness, another factor contributes to applicability of handwriting biometrics: explainability. Current systems focus on achieving high performance, but the inner workings are not often easy to explain to field experts. In Chapter 6 a method is introduced that regains explainability by representing handwriting as a relative position with respect to typical handwritings, the handwritings of *vantage writers*.

Chapter 5

Using ink width and directionality as a feature

A modified version of this chapter was previously published in *Pattern Recognition*. [14]

Historical handwriting written with a quill has a salient calligraphic appearance. The variability of the trace width, as illustrated in Figure 5.1, is caused by physical properties of the writing instrument and the individual writing style of the writer. In *quantitative paleography* [29], a recent methodology in the manual study of such historical documents, writing hands are discerned based on measurable characteristics. Two of the characteristics form the motivation for this paper: *contrast*, which is the difference of width between the thinnest and thickest traces, and the *angle of writing*, which describes the habitual orientation of the pen tip, determined by the angle between the thinnest ink traces and the base line. These characteristics suggest that trace width is an important feature for writer identification and that it is relevant to relate trace width to the trace direction. Trace width and direction can both be determined automatically using simple contour-based image processing operations, and in this paper we will show that the combination of the two yields a powerful feature for automatic writer identification. It is not limited to historical handwriting since modern handwriting contains trace subtle width variations as well.

The value of directionality measurements for writer identification is known [16, 26, 90, 94], but the added value of width measurements is new. Only one remotely related feature for writer identification has been evaluated that is based on *run lengths* of black pixels [2, 87]. Apart from this approach, to the best of our knowledge, ink trace-width measurements have not been used for writer identification. However, such measurements have been evaluated for a few other applications. One study used width measurements for stroke detection and structural analysis [30]. A second experiment included a distribution of coarse trace-width measurements in a signature verification experiment [62]. In another study on signature verification, the trace width was used to

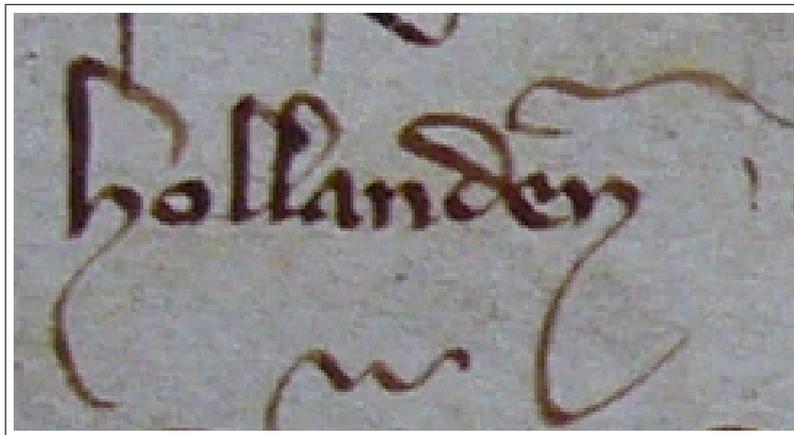


Figure 5.1: The word ‘hollanden’ in Dutch medieval handwriting. The width of the ink traces varies: vertical, southeast and northwest-bound strokes are thick, while southwest and northeast-bound strokes are thin.

express the mismatch between two signatures at corresponding locations [47].

Our approach is different in that width measurements are combined with direction measurements, and used for writer identification. The resulting statistic will be used as a text-independent writer identification feature, called *Quill*. It consists of a combination of simple methods, including trace-width computation using a method based on Bresenham’s well-known line-drawing algorithm [50]. *Quill* depends on pen properties and individual movement style. In this paper, the power of such computational measurements of ink trace widths relative to the writing direction as a feature for writer identification is explored on various handwriting datasets. We will show that its power to discern writers is comparable to that of *Hinge* [19] and *Fraglets* [86], which are among the world’s best features.

5.1 Datasets

The *Quill* feature will be evaluated on two datasets of medieval handwriting, the *Dutch charter dataset* and the *English diverse dataset*. It will also be evaluated on the contemporary datasets *Firemaker* (page 1 and 4 of each writer) and *IAM*, which are described in Section 1.12 on page 21. The medieval datasets are described in the next subsections.

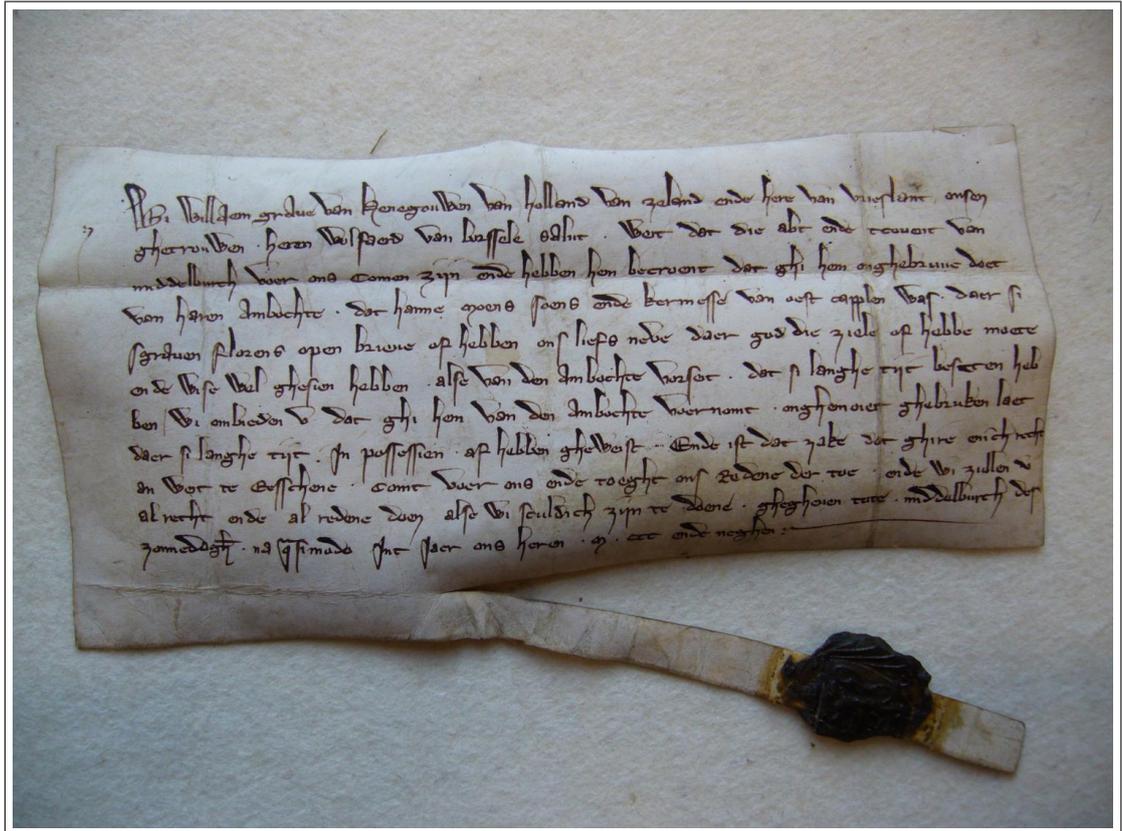


Figure 5.2: Document in the *Dutch charter dataset*: medieval handwriting in a charter, a legal administrative document (1309). Other charters written by the same writing hand can be found using writer identification. The original charter can be accessed via *Zeeuws Archief, Onze Lieve Vrouwe abdij te Middelburg*, access number 27, item number 80, regest number 119. Photo: Jinna Smit.

5.1.1 Dutch charter dataset

The *Dutch charter dataset* is a new dataset of 118 early 14th-century Dutch charters (1299-1328): administrative documents which served as evidence, written with quill (goose feather) on parchment. See Figure 5.2 for a sample photo. The charters are part of a collection that is studied at the University of Amsterdam in a research project called “Charters and Chancery of the Counts of Holland/Hainault, (1280)1299-1345”. The project is funded by NWO (the Netherlands Organization for Scientific Research) and part of the VNC (the Flemish-Dutch Committee for Dutch Language and Culture) program. As studies of medieval administrations rely heavily on writer identification, the different writing hands in this dataset were distinguished and code-named by the paleographers Jinna Smit and Jan Burgers. This was done independently and consistently, with the use of Burgers’s method [22], combining elements out of traditional and quantitative paleography.

The material in this dataset is graphically challenging in two ways. First, the originals contain difficulties such as pictorial letters, wrinkles, wax seals, and irregularly shaped parchment. Sometimes, parts of the ink are so faint that they are hard to distinguish. Many charters are in bad shape due to aging, tears or even fire damage. Second, the photos were taken using a 6 megapixel digital camera from different freehand positions and in different illumination conditions. Because of the free camera position, the images suffer from perspective distortions, non-uniform scaling, non-uniform illumination and occasionally blur. The photos also show document border shadows and sometimes document placement holders.

Most charters have been photographed several times; the dataset contains 248 photos in total. From this dataset, a variant was created, the *Dutch* charters dataset*, which does not include the “duplicate” photos. The duplicate photos of each original were removed by randomly selecting one photo to keep and removing the others. This procedure was repeated 25 times, averaging the results.

5.1.2 English diverse dataset

The *English diverse dataset* [27, 18] is a collection of 70 grayscale images of various late medieval texts (1375–1525), written by 10 scribes. The dataset was collected from library and archival resources by Professor Linne Mooney, expert codicologist-paleographer at University of York, UK. She also ascertained the authorship of each manuscript. This dataset was kindly provided by Dr John Daugman, University of Cambridge, UK.

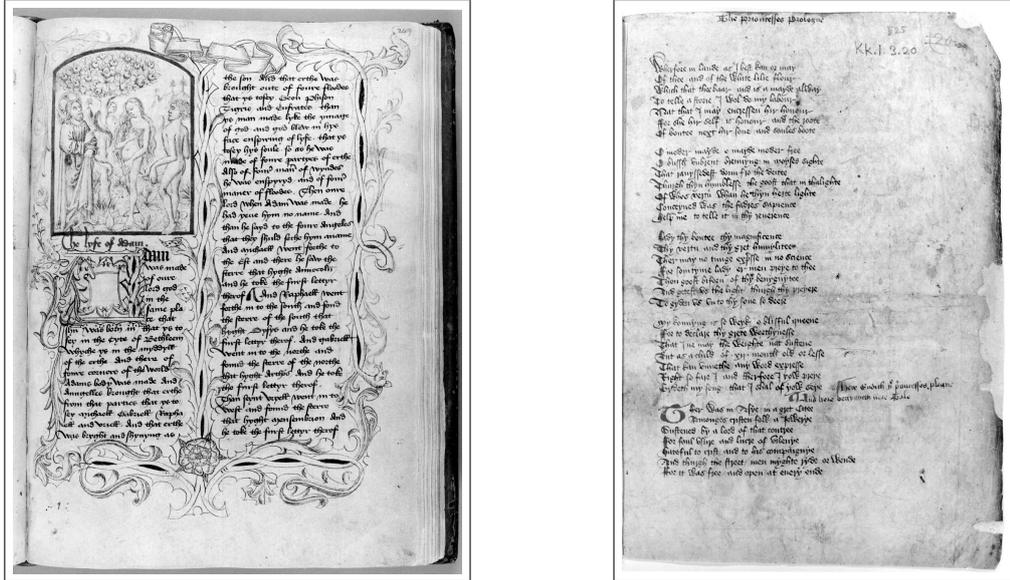


Figure 5.3: Example documents from the *English diverse dataset*: the left image shows part of a manuscript produced by the Trinity Anthologies Scribe, Cambridge - Trinity College R.3.21, folio 249 Middle English verse and prose.

The documents are graphically complex: most of them contain decorative frames, other decorations or pictorial letters. The layout also varies significantly. Frequently, the background is not uniform across the whole manuscript due to aging, stains and noise. They have been photographed from above, but probably from a variable distance thus the resolution of the digitization is not known and may not be constant. See Figure 5.3 for sample photos.

On each of the four datasets described above, the *Quill* feature will show to be effective for writer identification. The next section will discuss the theoretical background of this new feature.

5.2 Analysis of trace-width production

Since the *Quill* feature was inspired by paleographic methodology, it is instructive to understand how trace-width variations were produced in historical handwriting. Trace width is influenced by at least three factors: the writing instrument, the habitual tip

angle and individual movement style. These factors are described in the next subsections. Following, the basic principle of production of historical handwriting is modeled, and support for trace-width variation in modern handwriting is presented.

5.2.1 Writing instrument

Quills are capillary-action writing instruments that were used until the 19th century [51]. A quill was made from the feather of a bird, usually a goose's, by hardening it [93, p. 163] and cutting a *nib* (writing end). The feather's pen was first cut twice to create a sharp point and then topped, creating an oblong *tip* (contact surface). Finally, the nib was incised from this tip, splitting it in two flexible parts, the *tines* [67, p. 5–7]. Figure 5.4 shows such a quill nib. A few properties of this nib influence the trace width in a document [106, p. 72]:

- Length of the tip. This length depends on the radius of the pen and the position of the truncation on the nib. Scribes used to re-truncate the tip every so often during the process of writing, influencing the maxima in the ink trace width.
- Flexibility of the tines. The more flexible the tines are, the wider the ink traces could become. The flexibility is influenced by the length of the slit (incision) and degree of hardening. The flexibility also depends on properties of the used feather itself, as feathers vary naturally in pen stiffness, thickness and radius.
- Angle of truncation. The truncation was usually not exactly orthogonal, which had implications on the preferred pen grip.

5.2.2 Habitual tip angle

Scribes kept the orientation of the quill tip almost constant [45]. Since the tip of a quill is *oblong*, the effect on the produced ink trace is that it is thin where the tip was moved sideways and thick where it was dragged perpendicularly. Thus width variations were produced just by varying the writing direction. This principle is illustrated in Figure 5.5. The tip angle can be measured in the handwriting: it is generally parallel to the thinnest traces and perpendicular to the widest. This tip orientation is an individual influence on the handwriting [45].

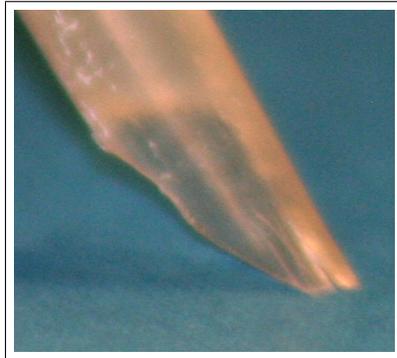


Figure 5.4: A quill nib. The slit (incision) partly splits the nib into two tines; this enables width variation by changing pressure.

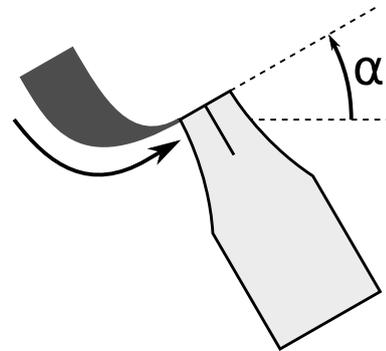


Figure 5.5: A near-fixed quill tip angle α caused by a near-fixed pen orientation determines the pattern of width variation.

5.2.3 Individual movement style

Individual movement style emerges partly deliberate and partly unconsciously. The following individual methods to influence the trace width are known [32, p. 81–84]:

- Force variation. The incision in the quill tip allowed the scribe to vary the trace width by varying force on the pen. An example of unconscious force variation is the individual way of creating tapered trace endings. This is visible in Figure 5.6 where the ‘z’, ‘e’ and ‘d’ have tapered trace endings, created by gradually decreasing the force on the nib. The force was also manipulated deliberately, for example when writing north or west-bound, which requires a “pushing” pen movement. In this case, to avoid damaging the parchment, only a very light force could be applied. The effect is possibly visible in Figure 5.6: in this example, the loops of the ‘l’ and ‘d’ are thinner than expected; these could have been drawn northwest-bound. It is also an established fact that force variation is very writer-specific in contemporary handwriting [89].
- Elevation (or pitch) variation. A lower pen elevation lowers the force needed to bend the nib’s tines and create a wider trace.
- Pen orientation (azimuth) variation. Even scribes with a very regular handwriting varied the orientation of the pen 2–3° in different documents [45]; it is conceivable that they also subconsciously varied the orientation within a document.



Figure 5.6: Width variation in the medieval Dutch word ‘zeland’. The superimposed contour (dashed) indicates the trace width to be expected when using a rigid oval pen tip. It illustrates individual influence on the trace width.

- Rotation (or roll). The quill could be rotated around its axis such that only one side of its surface touches the writing support; this enables creating curved thin lines. A slight pen rotation can be another explanation for the thin loops of the ‘l’ and ‘d’ in Figure 5.6. In some cases scribes *reversed* the pen (a rotation of 180°) to create thin traces.

The first three subsections of this section motivates that many pen-specific and individual behavioral properties have effects on the width of the ink traces. In the next subsection, these influences will be temporarily abandoned to study the basic principle of width production.

5.2.4 Trace-width production models

To aid the interpretation of the upcoming measurements, it is instructive to disregard individual movement style for the moment and make a model for the basic calligraphic effect: ink traces are thin where the tip is moved sideways and wide where it is dragged perpendicularly, as illustrated in Figure 5.5. This calligraphic effect is caused by the

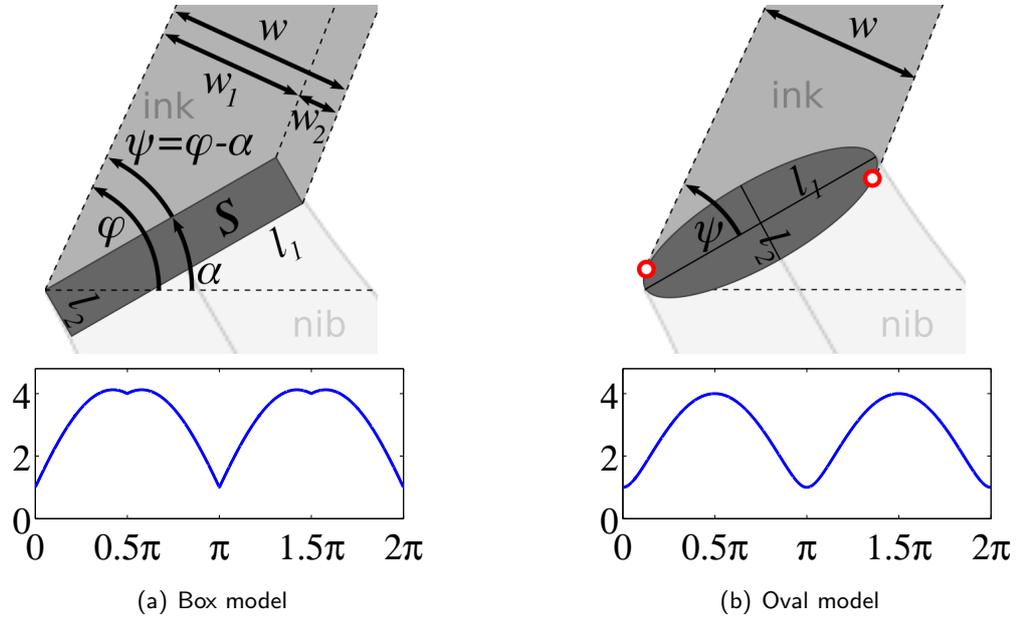


Figure 5.7: trace-width production in simplified conditions: the local ink trace width w depends on the tip shape and the relative orientation $\psi = \phi - \alpha$, where the tip is a rigid box or oval. Horizontal axis: ψ (radians); vertical axis: w (mm). w was plotted according equations (5.1) (left) and (5.2) (right). Tip dimensions $(l_1, l_2) = (4\text{mm}, 1\text{mm})$ are assumed.

oblong tip shape of historical pens. The exact effective shape of the contact surface during writing is not known, therefore the effect of two basic oblong shapes will be modeled here: a *box* and an *oval*. Both shapes will be modeled to be rigid and fixed at a rotation angle α , similar to usage of a *brush* in a graphics program. The oval shape may be more accurate than the box shape since cohesion and adhesion effects must make the ink at the quill tip round. The top row of Figure 5.7 shows these shapes plus the parameters that will be used for the models.

In these simplified conditions, the trace width w only depends on the *relative* orientation $\psi = \phi - \alpha$, where ϕ denotes the local trace direction and α denotes the tip orientation. In the bottom row of Figure 5.7, this relation is visualized in a graph for the two simple pen tip shapes, for any relative orientation. The graph for the box model was computed using Equation (5.1), which computes the predicted trace width w for any relative orientation ψ , assuming a box shape of dimensions (l_1, l_2) . The formula is

easily derived from Figure 5.7(a). The graph for the oval model was computed using Equation (5.2), which determines the height of a parametrized oval after rotation, where $t^* = \arctan \frac{l_2 \cos \psi}{l_1 \sin \psi}$ represents the parameter value at the top. Although the formulas for computing the graphs are not used for the *Quill* feature, these are mentioned for completeness and future reference.

$$w(\psi, l_1, l_2) = |l_1 \sin \psi| + |l_2 \cos \psi| \quad (5.1)$$

$$w(\psi, l_1, l_2) = |l_1 \cos t^* \sin \psi + l_2 \sin t^* \cos \psi| \quad (5.2)$$

The graphs demonstrate what the result of the *Quill* feature should look like if a rigid oblong tip shape can be assumed. Any deviation from these graphs will show the presence of individual influence on the trace width. Figure 5.6 shows a piece of real data and illustrates that the oval model does not explain all width variation: the ink does not follow the expected contour exactly, proving the existence of influence on the trace width by the personal writing habits of the scribes. This individual influence makes the trace width a valuable source of writer-specific features.

5.2.5 Trace-width production in modern handwriting

Modern handwriting contains trace-width variations as well, although not as pronounced as in historical handwriting. At least two types of individual writing style play a role. First, the trace width is directly influenced by the force applied to the pen tip [30, 47, 38], and different writers apply this force differently [89]. In particular, downstrokes are usually wider than upstrokes [30, 47]. Second, the width is altered by local retracings [30].

5.3 Quill feature

The *Quill* feature $p(\phi, w)$ captures the relation between local width w and direction ϕ of ink traces in a probability distribution. It expresses properties of the used pen and the writer's unique way of producing variations in the width of the ink trace. The feature consists of a few simple parts that together form a powerful method for writer identification: contour tracing, angle measurements, width measurements, and calculation of a probability distribution. These parts are further explained in the next paragraphs.

to reconstruct the actual direction [30], although not reliably. Instead, ϕ is defined to be the angle of a local tangent line with respect to the horizontal, where the ink is to the left-hand side (in image space) of the tangent line. This is illustrated in Figure 5.9. This systematic approach ensures robust measurements, rendering the actual writing direction unimportant: since the contours are circular, every measurement will generally be accompanied by an opposite measurement on the other side of the trace; both measurements counterbalance each other.

At contour pixel C_i , the i th pixel of contour C , ϕ can be estimated using two nearby contour pixels C_{i-r} and C_{i+r} which are the endpoints of imaginary ‘legs’ originating from C_i at a distance of r contour pixels. A straightforward approach to estimate the local orientation ϕ would be to simply calculate the angle between C_{i-r} and C_{i+r} . This approach has the disadvantage that it is not accurate at positions of greatly varying curvature, such as stroke endings. Therefore, a slightly more elaborate method was used.

The leg from C_{i-r} to C_i defines an inbound angle ϕ_1 ; the leg from C_i to C_{i+r} defines an outbound angle ϕ_2 . Since the pixel C_i is in the middle of these legs, ϕ can be estimated as the angle *between* ϕ_1 and ϕ_2 . Because of the periodicity of angles, special care has to be taken to ensure that the resulting direction keeps the ink on the left side (in image space). When the difference between ϕ_1 and ϕ_2 is smaller than π radians, by definition the ink must be in the region between these angles. But when this difference exceeds π , by definition the ink must be on the other side, thus the resulting angle flips. Summarizing, ϕ is computed as follows:

$$\phi = \begin{cases} (\phi_1 + \phi_2)/2 & \text{if } |\phi_1 - \phi_2| < \pi \\ (\phi_1 + \phi_2)/2 + \pi & \text{otherwise} \end{cases} \quad (5.3)$$

5.3.3 Width measurements

After computing the local angle ϕ at contour pixel C_i , the local width of the ink trace w is computed. Any robust method could be used as part of the *Quill* feature. A few methods to compute trace width have been proposed before, designed for application on signatures [30, 47]. These methods are based on traversing a line from C_i through the ink, perpendicular to ϕ , until the background is hit.

In this study, a variant of this principle was used because of its simplicity: a method based on Bresenham’s algorithm [50], which constructs an approximated (quantized)

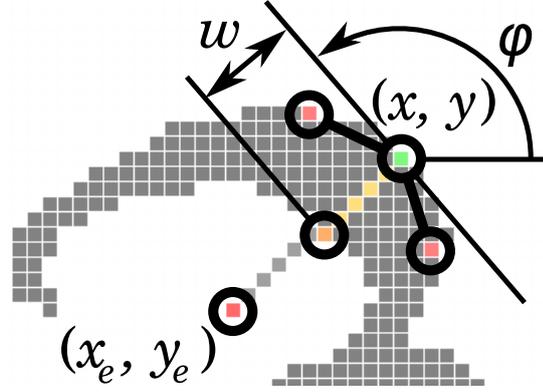


Figure 5.9: ϕ and w are determined at each contour pixel (x, y) . ϕ (trace direction) is measured by averaging the angles with two neighboring contour pixels at distance r . w (trace width) is computed using the so-called *Bresenham width*: the distance to the first background pixel that is hit when following a Bresenham path, perpendicular to ϕ , towards (x_e, y_e) .

linear path of pixels between two given pixel positions in an image. In this case, the starting pixel of this path is $C_i = (x, y)$. The end pixel (x_e, y_e) is a pixel that is on the line perpendicular to ϕ . This is illustrated in Figure 5.9. The precise position of (x_e, y_e) is determined by a parameter, m , which signifies the maximum measurable width, as shown in Equations (5.4) and (5.5):

$$x_e = x + m * \cos(\phi + \frac{1}{2}\pi) \quad (5.4)$$

$$y_e = y + m * \sin(\phi + \frac{1}{2}\pi) \quad (5.5)$$

The pixels on the Bresenham path are traversed from C_i to (x_e, y_e) and checked for color: the algorithm stops if a background (white) pixel is hit. The trace width w is then computed as the distance from C_i to this background pixel (x_b, y_b) using a simple Euclidean measure:

$$w = \sqrt{(x - x_b)^2 + (y - y_b)^2} \quad (5.6)$$

In the following, this method to compute the trace width will be called *Bresenham width*. In the next section, this method will be used together with angle measurements to form the *Quill* feature.

5.3.4 Probability distribution

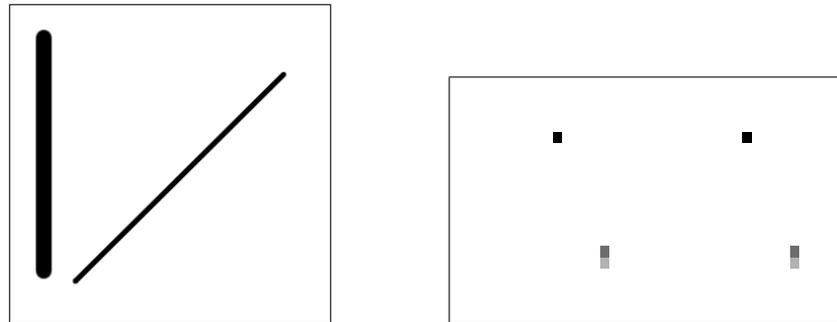
The locally measured direction ϕ and width w are agglomerated in a probability distribution $p(\phi, w)$. It will be referred to as the *Quill probability distribution* (QPD). It is created as follows. Every measurement (ϕ, w) is agglomerated in an interpolated $p \times q$ histogram, where p is the number of bins into which the measured width is quantized; q is the number of angle bins. In the following, p is equal to m (maximum width) for simplicity, so each width bin corresponds to one pixel of width in the ink. The histogram was built using bilinear interpolation, updating four bins at once for every measurement, to avoid distortions caused by measurements close to bin boundaries. This is relevant because ϕ is discrete and because of delicate rounding errors due to angle periodicity. The resulting histogram was converted into a probability distribution by normalization, which makes it independent of the amount of text and usable as a writer-specific feature vector.

Algorithm 4 *Quill* feature. INPUT: binary image I , leg length r , number of width bins p , number of angle bins q . OUTPUT: 2D probability distribution P .

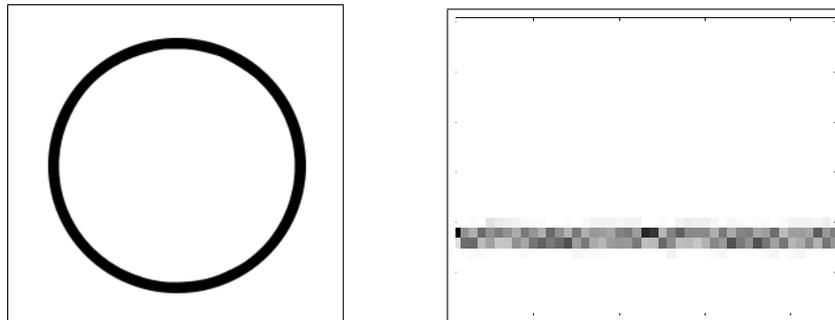
```

 $H \leftarrow$  empty_histogram( $q, p$ )
 $Cs \leftarrow$  contourtrace( $I$ )                                { $Cs$  is a list of contours}
for all  $C$  in  $Cs$  do
   $n \leftarrow$  len( $C$ )                                       { $n$  is the current contour's length}
  for  $i$  in  $[0$  to  $n - 1]$  do
     $\phi_1 \leftarrow$  angle( $C[(i - r) \bmod n], C[i]$ )
     $\phi_2 \leftarrow$  angle( $C[i], C[(i + r) \bmod n]$ )
     $\phi \leftarrow$   $\begin{cases} (\phi_1 + \phi_2)/2 & \text{if } |\phi_1 - \phi_2| < \pi \\ (\phi_1 + \phi_2)/2 + \pi & \text{otherwise} \end{cases}$ 
     $(x, y) \leftarrow C[i]$                                    {( $x, y$ ) is the current contour pixel}
     $x_e \leftarrow x + p * \cos(\phi + 1.5\pi)$ 
     $y_e \leftarrow y + p * \sin(\phi + 1.5\pi)$ 
     $w \leftarrow$  bresenham_width( $x, y, x_e, y_e$ )             {Compute width}
     $H$ .update( $\phi, w$ )                                       {Update histogram, interpolated}
  end for
end for
 $P \leftarrow$  normalize( $H$ )                                  {Make probability distribution}

```

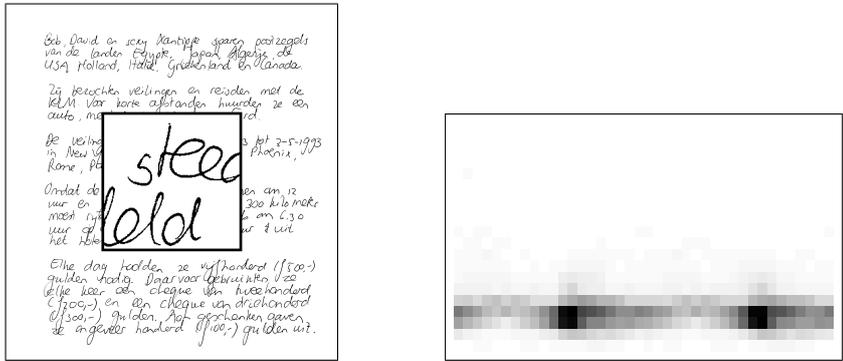


(a) Artificial lines. The upper dark spots correspond to the vertical stroke; the lower spots correspond to the diagonal line.

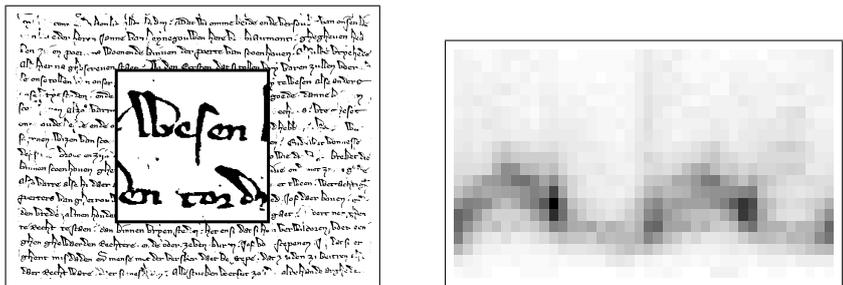


(b) Artificial circle. It has a constant trace width in all directions; it could have been produced using a stylus with a round tip and homogeneous ink deposition, possibly a fineliner.

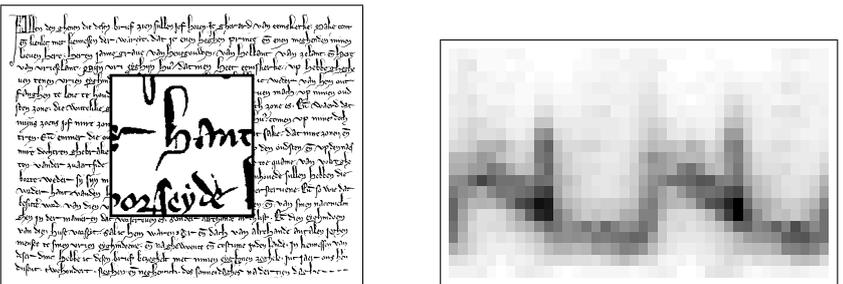
Figure 5.10: Artificial binary images (left) and their *Quill* probability distribution (QPD, right): a distribution of (ϕ, w) combinations. Dark regions indicate frequent combinations. The horizontal axis represents the trace direction ϕ ($0..2\pi$ radians), quantized in $q = 40$ bins; the vertical axis represents the trace width w ($1..20$ pixels) in $p = 20$ bins. Used parameters: $q = 40, p = 20, r = 10$.



(a) Contemporary handwriting in the *Firemaker* dataset. Text produced with a ballpoint; width variation is limited.



(b) Medieval Dutch text (1322) in the *Dutch* dataset. The QPD shows a wave shape, as predicted by the models in Figure 5.7, thus this is quite regular medieval writing.



(c) Medieval Dutch text (1300) in the *Dutch* dataset. The writer produced near-horizontal and near-vertical traces in a variety of widths, which show as vertical bands in the QPD.

Figure 5.11: Binarized handwritten documents (left) and their *Quill* probability distribution (QPD, right).

5.3.5 Interpretation

By visually inspecting the QPD a variety of properties of the handwriting can be derived. See Figures 5.10 and 5.11 for binarized example images and their QPD. Modern handwriting written with a ballpoint pen results in a near-horizontal structure, as the trace width is nearly the same for all ink directions. Dark (high-frequency) regions indicate frequently used angles, which are closely related to the handwriting's slant. For historical handwriting, the QPD reveals other properties of the handwriting as well:

- The most salient property of a QPD of historical handwriting is that it shows a wave shape, as predicted by the models in section 5.2.4. The shape repeats itself after a period of π radians (180°) because generally every measurement on one side of a trace has a counterpart on the other side, in the opposite direction. The presence of peaks and valleys indicates that a writing instrument with an oblong tip was used, held at a near-fixed orientation: the ink width depends on the direction.
- The valleys correspond to the thinnest traces. The w value of the valleys corresponds to the width of the thinnest strokes; the ϕ value of the valleys reveals the near-fixed pen-tip angle: $\alpha \approx \phi \pm \pi$. By fitting a model described in 5.2.4 to the raw measurements, α could be estimated automatically.
- Similarly, the peaks correspond to the widest traces. The w value of the peaks corresponds to the width of the thickest strokes and the ϕ value of the peaks reveals the near-fixed pen-tip angle: $\alpha \approx \phi + \frac{\pi}{2} \pm \pi$.
- The deviation from the models in section 5.2.4 indicates the influence of physical pen properties and individual movement style, as described in section 5.2.
- The ϕ value of the cell with the highest intensity (dark region in the figure) reveals the dominant stroke direction, which is closely related to the dominant *slant angle*. The corresponding w value reveals the dominant stroke width.

5.3.6 Variant: Quill-Hinge feature

Inspired by the success of the Hinge feature a modification of the *Quill* feature was developed: the *Quill-Hinge* feature $p(\phi_1, \phi_2, w)$. It records w in conjunction with ϕ_1 and ϕ_2 instead of ϕ , making the feature three-dimensional.

5.4 Performance experiment

The performance of the *Quill* feature was tested in a writer identification experiment on the four datasets that were introduced in section 5.1. Its performance was also compared with the performance of other features. This section discusses the experiment.

5.4.1 Preprocessing

The images in the datasets of modern handwriting were preprocessed using text region cropping based on known fixed coordinates followed by Otsu thresholding. The described medieval documents are graphically more challenging and required additional steps:

- *Manual text region selection:* A region of interest (ROI) was manually selected in a graphical interface (GIWIS) by placing a four-sided polygon, which enables the careful selection of a text area that is rectangular in reality but subjected to perspective transformation. This was essential for the *Dutch charter dataset*, where the camera position was free.
- *Perspective correction:* The perspective distortion in the *Dutch* and *Dutch** charter datasets was corrected by a reverse perspective projection. The parameters were derived from the positions of the four vertices of the ROI. The result was a rectangular image containing a version of the ROI, stretched using bilinear interpolation.
- *Automatic scaling:* The scale was estimated from the height of the text lines in the images, assuming that the true height of the text lines is equal in all documents. The text line height was determined by measuring the median width of the peaks in the smoothed horizontal projection profile of dark pixels. The images were then scaled to match a standard text line height of 50 pixels.
- *High-pass filtering:* Gradual intensity variations were canceled by applying a high-pass filter, after grayscale conversion. This was implemented by a straightforward approach: blurring the image and subtracting that from the original image.
- *Otsu thresholding:* The last preprocessing step was to binarize the image using Otsu thresholding [75], which is widely recognized as a good general-purpose binarization method.

5.4.2 Writer identification experiment

The preprocessed datasets were used to test the power of the feature in a writer identification experiment. Writer identification is the recognition of the writer of a query document by yielding a *hit list*: a list of database documents that are similar to the query document in feature space. Typical hit list sizes are 1 or 10, therefore the feature's performance will be expressed as its *top-1* and *top-10 writer-identification performance*, which means that a hit list will be counted as correct if at least one document of the query writer appears in it. Top-1 performance is also called nearest-neighbor accuracy.

In this experiment, any classifier could be used, but nearest-neighbor (instance-based) classification is used for reasons mentioned in Section 1.11. The top- x performance is simply computed by treating each dataset document as a query, sorting the other documents by similarity, and counting how often another document from the same writer appears among the x most similar documents. Similarity is based on two documents' feature vectors and a distance measure. The distance measure to determine the similarity of two documents' corresponding feature values was the χ^2 distance [25] (described in Section 1.9), because it has been shown to be effective on feature vectors of *Hinge* and *Fraglets* [85].

The following features were evaluated: *Directions*, *Brush*, *White runs*, *Hinge*, and *Fraglets*; these are described in Section 1.8. The code-book for *Fraglets* was pre-computed using modern handwriting: all four pages of the first 100 subjects of the *Firemaker* dataset. In addition, the new *Ink width* feature was evaluated: *Ink width* ($p(w)$) is a probability distribution (p.d.) of trace-width occurrence. It is a modified version of the *Quill* feature, where the number of angle bins is set to one ($q = 1$). The effect is that this feature only measures the distribution of ink widths, without regarding the corresponding direction. This feature was included to roughly evaluate the importance of the ink width in the *Quill* feature. Furthermore, four combinations of features were also tried: *Hinge & Fraglets*, *Quill & Hinge*, *Quill & Fraglets*, *Quill-Hinge & Fraglets*. These combinations were made by simply averaging the distance values.

For the *Dutch* charter dataset* the experiment was repeated 25 times; in each experiment one photo of each original was selected randomly. The results were averaged over the 25 runs.

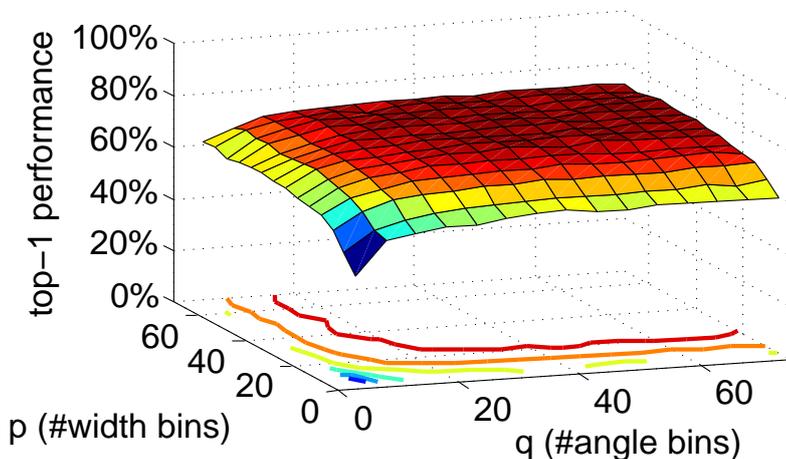


Figure 5.12: Sensitivity of Quill's top-1 performance for the parameters q (number of angle bins) and p (number of width bins) in the *Dutch* charter* dataset for $r = 20$. The graph shows that the choice of the parameters is almost arbitrary, as long as $q > 30, p > 30$.

5.4.3 Training

Instance-based classification does not require any training other than storing feature values, but the *Quill* feature contains three parameters (p , q and r) that need to be optimized. This was done using simple methods as will be described below. It will also be shown that the feature is not very sensitive to the actual parameter settings.

Optimizing the parameter values for p , q and r was done by evaluating 896 parameter combinations from a regularly spaced three-dimensional lattice on the *Dutch* charter dataset* using a 4x4-core PC with 128 GB of memory. It was not the maximum performance score in this evaluation that determined the final choice of the parameters. Instead, good parameter values were determined by visual inspection of the data plotted in figures such as Figures 5.12 and 5.13. This procedure minimizes the effect of overtraining that may exist since no separate training set was used.

Figures 5.12 and 5.13 show the sensitivity of *Quill* for its parameters. The three-dimensional evaluation cannot be fully visualized, therefore two 'slices' of the results are shown. Figure 5.12 shows how the performance relates to p and q , given a fixed $r = 20$. It shows that the performance is insensitive to the values of p and q , as long as they are at least about 30. Increasing the values much further does not increase the performance, but has a negative effect on memory usage and speed. The values $p = 40$

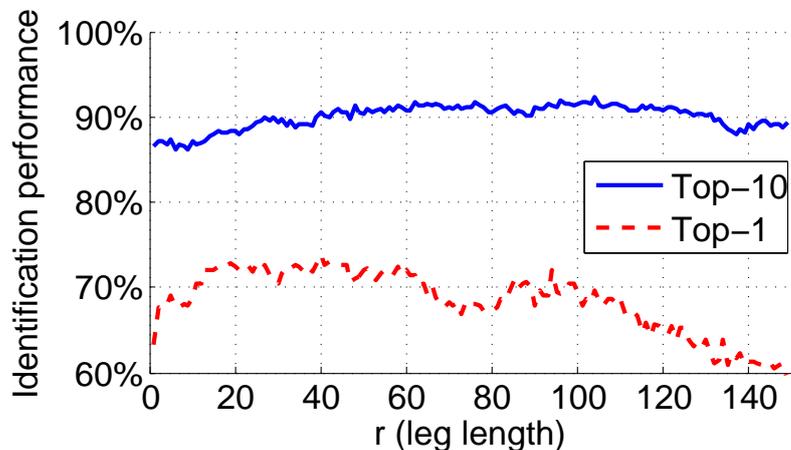


Figure 5.13: Sensitivity of *Quill*'s top-1 and top-10 performance for the parameter r (leg length) on the *Dutch** charter dataset; $q = 40, p = 40$. The graph shows that the influence of this parameter is marginal, as long as the value is between about 10 and 100.

and $q = 40$ were chosen as safe values. The effect of the parameter r (leg length) on the performance is shown in Figure 5.13. It shows that this performance is hardly affected by this parameter, as long as the value is between about 10 and 100. This is the same order of magnitude of the corpus height of the text (50 pixels). $r = 20$ was chosen because higher values are less suitable for small characters and less intuitive. Summarizing, the found parameters are $p = 40, q = 40$ and $r = 20$.

5.5 Results

Table 5.1 shows the results of all performance experiments. Top-1 performances of the well-known *Directions* feature are in the range 48–74%. The *Ink width* feature, on the other hand, performs 22–73%. It coarsely describes the informational value of ink trace width: it is low in *Firemaker*, where ballpoints were used, intermediary on the medieval datasets and high in *IAM*, where different types of pens were used. The contribution of ink width in the Firemaker dataset seems small, but the *Ink width* feature does not take structural dependency on the trace direction into account while *Quill* does.

Ink width awareness partly explains *Quill*'s performance on the datasets, most notably *IAM*, in which different types of pens were used. The *Quill* feature performs much

Table 5.1: Writer-identification performance of several features (top, middle) and feature combinations (bottom) on datasets of medieval and contemporary handwriting. The numbers represent recognition percentages.

	Medieval handwriting						Contemporary handwriting			
	Dutch*		Dutch		English		Firemaker		IAM	
	18 writers		18 writers		10 writers		251 writers		657 writers	
	118 images		248 images		70 images		502 images		1539 images	
25×112 queries		245 queries		70 queries		502 queries		1183 queries		
	Top1	Top10	Top1	Top10	Top1	Top10	Top1	Top10	Top1	Top10
Directions	57	86	69	93	56	93	48	79	74	90
Ink width	47	79	66	89	40	87	22	52	73	88
Quill	75	90	92	98	63	96	71	89	95	97
Quill-Hinge	75	89	92	98	70	96	86	97	97	98
<i>Comparison features</i>										
Brush	38	79	53	85	40	86	37	77	78	89
White runs	43	86	67	94	37	83	22	57	43	76
Hinge	71	92	86	97	76	93	83	92	94	96
Fraglets	73	94	92	98	89	100	72	90	97	98
<i>Combinations</i>										
Hinge & Fraglets	75	93	94	98	90	99	76	93	97	98
Quill & Hinge	73	91	93	98	74	96	83	95	96	97
Quill & Fraglets	77	94	94	98	86	99	78	95	97	98
Quill-Hinge & Fraglets	77	94	94	98	86	99	81	96	97	98

better than *Directions* or *Ink width* alone: top-1 performances are in the range 63–95%. This proves that the combination of directionality measurements with trace-width measurements is fruitful. It also supports the foundation of characteristics in quantitative paleography that are based on trace angle and width. The value of trace directions for writer identification was known, but the added value of width measurements for writer identification is new.

Even when compared to the other features, *Quill* proves to perform very well: it amply outperforms *Brush* and *White runs*, and on average, it performs as well as *Fraglets* and *Hinge*, which are among the world’s best features. The modified version of the feature, *Quill-Hinge*, seems to perform even better than *Quill*. Slightly better performances are achieved by combining features. Top-1 performances of *Quill & Fraglets* and *Quill-Hinge & Fraglets* are in the range 77–97%. The results were obtained by averaging the distance scores for both features in the combination. Table 5.2 puts the performance of *Quill-Hinge* in the context of leading results obtained by others. Only results on modern handwriting are shown since little similar work has been done on historical handwriting.

Note that the reported performances may be several percentage points off (except in the column ‘Dutch*’) due to the relatively low number of queries and the influence of randomness. For example, the 95% confidence interval for a recognition score of 92 with 245 queries (*Quill* in column Dutch) is the range 88–96 (based on the Binomial distribution).

Still, these performance figures suggest that *Quill* and *Quill-Hinge* can be used in a production environment. Researchers in application fields are currently evaluating the features [4, 74, 92] in our GIWIS software program, a user-friendly graphical user interface. A recognition score of 100% is not necessary: based on a top-10 hit list, the domain specialist can make the final decision. However, some improvements are still possible, as we will describe in the following.

5.6 Future work

A simple optimization approach is to try other distance measures. An excellent overview of available distance measures for probability distributions exists [25] which can serve as a starting point. However, the currently used χ^2 distance measure has proved to be effective in previous experiments [85].

The method of combining features could be improved as well. Simply averaging could

Publication	writers	top1	top10
Bensefia et al. [6]	150	87%	99%
Bulacu [21]	900	87%	96%
Garain et al. [44]	422	62%	96%
Schlapbach et al. [82]	100	97%	98%
Schomaker et al. [86]	150	97%	100%
Siddiqi et al. [90]	650	86%	97%
Srihari et al. [94]	900	88%	
Quill-Hinge	251	86%	97%
Quill-Hinge	657	97%	98%

Table 5.2: Performance of *Quill-Hinge* on modern handwriting compared to leading results obtained by others. Note that the numbers cannot be well compared because of differences in dataset material, required level of human interference, and number of writers.

be replaced by feature weighting, however little gain is expected since it is known that *Fraglets* and *Hinge* are best weighted by plain averaging [19]. This was confirmed by a small pilot experiment with *Quill* and *Fraglets*. The features could also be combined in different ways, for example, using Borda ranking.

Preprocessing could be improved. Although the preprocessing used in the performance experiment generally works quite well, it breaks the thinnest faint ink traces. The result is that measurements on those traces are underrepresented in the QPD, resulting in suboptimal performance. Preserving this weak signal is still a hard image processing challenge.

Quill and *Quill-Hinge* are pen-dependent features. This can be advantageous but it may not be so if a significant number of writers each use multiple pens. To partly cancel the pen dependence as described in 5.2.1, one of the models presented in 5.2.4 could be fitted to the measurement data, followed by a rescaling of the data based on the width range. This could also compensate for scaling differences due to varying camera distance

or resolution; these are now dealt with in an explicit preprocessing step. However, this model fitting is not trivial since the data contains structural noise.

For performance analysis, better medieval datasets could be collected or constructed. The results on the used datasets are not fully reliable since some writer labels may be wrong: the labels were manually determined based on skill and experience, not facts. Furthermore, the currently used datasets are relatively small. Larger datasets with known writer labels will make the results more reliable.

The presented methods can be used for other applications as well, including pen and script type estimation, by analyzing the QPD, and estimation of the modal pen-tip orientation α by fitting the models from 5.2.4 to the measurement data.

5.7 Conclusion

As suggested by modern paleographic methodology, the width of the ink trace is a powerful source of information for writer identification, particularly in combination with the trace direction. This is not only true for off-line writer identification on historical handwriting, which shows salient width differences because of usage of quill pens, but for modern handwriting as well. This was found in a series of writer identification experiments using a newly introduced feature: *Quill*. It is a 2D joint probability distribution of ink trace width and direction. The feature consists of simple, fast and accurate methods based on pixel contours. The feature was tested on two datasets of modern handwriting and two datasets of medieval handwriting: writer identification scores (nearest-neighbor classification accuracy) scores are in the range 63–95%. This is much higher than the individual performances of features based on either the ink width or direction. It even approaches the performances of *Hinge* (71–94%) and *Fraglets* (72–97%), which are among the world’s best features. A slightly more complex version of the feature involving curve measurements, *Quill-Hinge*, seems to perform even better. The performance of *Quill* and *Quill-Hinge* strengthens the foundation of related measurements in quantitative paleography. The features can be used as general-purpose writer identification features. Slightly higher performance can be achieved by combining the features with other features (77–97% with *Fraglets*). In GIWIS, a user-friendly user interface, the features are already helping historians fruitfully.

Chapter 6

Increasing explainability using vantage writers

A modified version of this chapter was previously published in *ICDAR*.
[13]

Despite the fact that several systems for handwriting biometrics have been implemented and impressive performances have been reported [78], still no system exists that convinces forensic experts. One of the reasons is that the output of such systems is often hard to interpret. Experts consider them as black boxes of which the inner workings are unclear. These systems usually yield numbers without an intuitive explanation. That issue is addressed in this chapter.

An adaptation to automatic off-line writer verification and identification is proposed, which allows to generate reports that are comprehensible. The principle is that a person's handwriting can be seen as a relative position with respect to handwritings of typical writers, the *vantage writers*. The degrees of dissimilarity between a document and each of the vantage writer's handwritings form a small list of numbers, called a *vantage profile*. This profile is used to discriminate writers. The vantage profile is comprehensible, because it represents a relative position with respect to documents that can be shown to the user.

A vantage profile based on vantage writers is an implementation of a *dissimilarity representation*, as introduced in [76]. In that publication, classes are discriminated by a Bayes classifier that assumes normal distributions of the dissimilarity values. We used a different approach for the classification, because there are very few samples per writer. A similar approach is described in [104], where hieroglyphic images are retrieved using *vantage objects*. In [105], on-line characters are hierarchically clustered, where each cluster is represented by a prototypical input sample, and each writer is assigned a style vector that indicates the usage of each of the prototypical characters. In [107], characters from several copybooks were clustered in order to determine the nationality of writers based on their character usage. In [72], writing styles are expressed as usage of inferred

copybooks.

To test how the vantage profile approach performs, both writer verification and identification experiments were performed based on vantage profiles. Two datasets were used separately; one of them is a collection of confidential forensic data. In the next section, the method and experiments are described in detail. In section 6.2, the verification and identification performance of the vantage writer method are presented. Finally, in section 6.3, the results are discussed and future work is suggested.

6.1 Method

For each image dataset, basic features of the images were extracted first, resulting in two feature datasets. These were each split into a *train set* and a *test set*. The train set was used to select the vantage writers. It was also used to determine the classification threshold for verification. The test set was used to assess the performance independently. A more detailed description follows in the next subsections.

6.1.1 Dataset preparation

Two datasets were used separately to provide the input patterns for the experiments: *Firemaker* [88] (see page 21) and *NFI*. The *NFI* dataset is a heterogeneous set of handwritten pages that have been collected by the NFI, the Dutch National Forensic Institute. It was used for the experiments in Chapter 3 as well. The collection consists of 3501 scanned forms with handwritten material, on demand written by 1311 suspects in criminal cases. Most of the suspects wrote two pages, but there are many exceptions. Most of them wrote a dictated standard text; the first part in connected cursive, the second part in capitals. In many cases the transition from cursive to capitals occurs within a page. The handwriting in this dataset gives an impression of great sloppiness. This is due to the facts that the forms were not lineated, the subjects have a lower level of education and some of the subjects may have not been cooperative. Also, many pages contain erasures or visible perforator holes. Altogether, one could conclude that this is a “dirty” dataset. For examples, see Figure 6.1.

For performance evaluation, the Firemaker dataset could be used as-is, but the NFI data required some preprocessing. First, border effects like visible pieces of form fields and the paper edge were removed by cutting along straight horizontal and vertical lines. The positions of the cutting lines were determined heuristically using a smoothed pro-

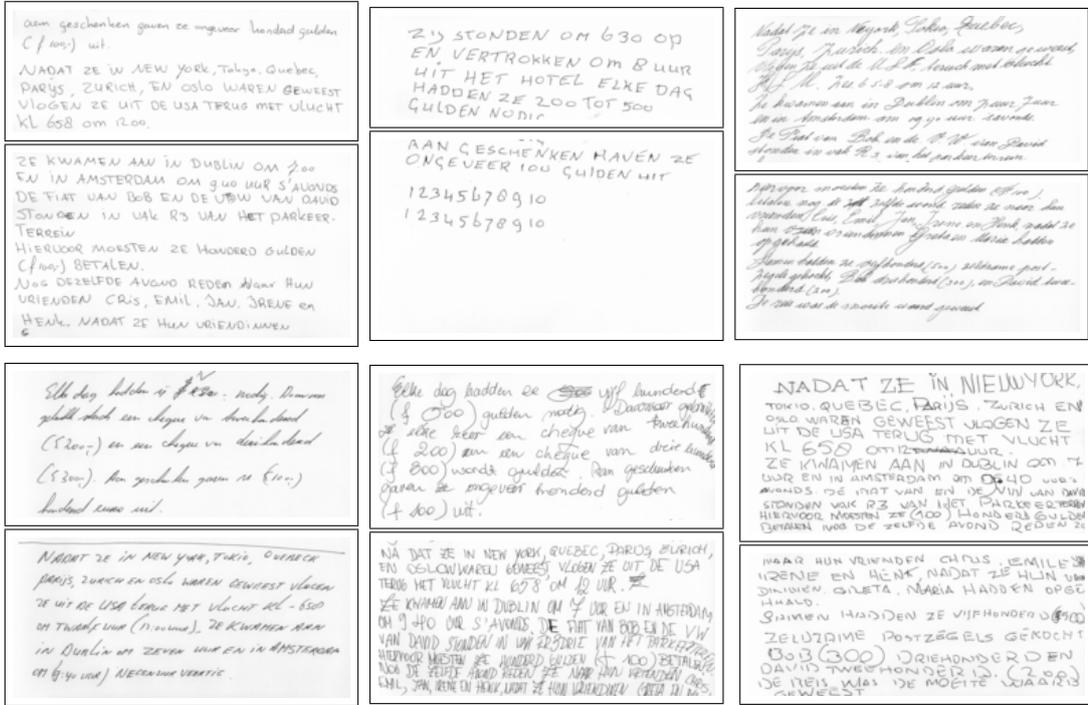


Figure 6.1: Example documents in the NFI dataset, each one cut in two parts.

jection histogram. The same kind of histogram was used to split the page into an upper and a lower part, to increase the probability that every writer has at least two pieces of similar text in the database; the parts were treated as separate documents. This method is relatively simple, and visual inspection proved that it works reasonably well. However, as much as 1127 pages had to be rejected because the text baseline was too curved or sloped, making a horizontal cut between two text lines in the middle of the text impossible. Still, after splitting, 4748 page parts remained, written by 1074 persons. The fact that the pages were cut in two parts that are treated as separate documents will introduce an optimistic bias in the verification and identification results. On the other hand, the NFI data has not been collected with automatic processing in mind, and is contaminated in many ways, which has a negative effect on the performance figures.

6.1.2 Basic feature extraction

Creating a vantage profile for an image requires that a basic feature vector has already been computed. This can be done using any statistical feature extraction method. In this experiment, feature vectors were computed for all input documents using a method that has proved to be very effective: the *Hinge* feature [19] (see page 13).

6.1.3 Selecting vantage writers

Vantage profiles represent writer information as relative position with respect to the handwriting style of typical writers, the vantage writers. To create these profiles, the set of vantage writers must be defined first. This can be done in various ways. For example, they could be manually selected to represent styles from different countries, sexes or ages. In our implementation, the vantage writers were represented by a sample of input documents in the training part of the dataset, the *vantage-writer sample*. This was done by random sampling a fixed number of times, and picking the best choice afterward. The number of random samples was set to 50 for the Firemaker dataset, and 25 for the NFI dataset, for reasons of computing time. For every sample, the performance was computed by creating vantage profiles and performing writer verification or identification on the train set as described in the next subsections. The sample yielding the highest performance on the train set was assigned to be the final set of vantage writers. Vantage writers were determined for verification and identification separately.

6.1.4 Creating vantage profiles

The vantage profile \mathbf{v} of a document is acquired by computing the *distance* between its basic feature vector \mathbf{x} and the basic feature vectors \mathbf{y}^j of the sample documents of each of the vantage writers j . The distance can be computed using various measures, such as Euclidean, Hamming, χ^2 , Bhattacharyya, etc. Since a hinge feature vector is essentially a probability distribution and the χ^2 measure has proved to be effective for this kind of feature vectors [85], this measure was used.

The distances together form a vector of distances \mathbf{v} , which we call a *vantage profile*. This can be written as:

$$\mathbf{v} = (d(\mathbf{x}, \mathbf{y}^1), \dots, d(\mathbf{x}, \mathbf{y}^n))$$

where d is a dissimilarity measure, \mathbf{y}^j are the feature vectors of the vantage writers and n is the number of vantage writers. It is treated as a new, indirect, feature vector that

is used to discriminate writers. As such, the computation of vantage profiles can be seen as a form of dimensionality reduction. See Figure 6.2 for an example.

6.1.5 Writer verification

A threshold for verification was learned from the documents in the train set that were not used in a vantage sample. This was done by modeling the distances between vantage profiles within the “same-writer” and “different-writer” classes. These distances were found by comparing the vantage profiles of each pair of documents and computing the Euclidean distance between them. From the distances in both classes, smoothed-probability densities were created using Parzen windowing with a Gaussian kernel. See Figure 6.3 for an example. Based on these probability densities, the threshold was positioned for the highest expected performance in terms of the *true positive* (TP) and *true negative* (TN) percentages (in other words, the lowest *false negative* and *false positive* percentages). The expected TP and TN could be balanced according to the desire of the end user, but as there was no such information available yet, the threshold was selected such that $TP = TN$: the *equal-error rate* (EER). The EER was also used as the performance measure for vantage writer selection.

6.1.6 Writer identification

In this experiment, each document in the train set was once treated as a questioned document. Its *hit list* was constructed, containing the s nearest neighbors of the questioned document based on the vantage profiles, with $s = 1, 10, 100$. During the selection of vantage writers, the writer-identification performance was simply assessed as the top-1 performance ($s = 1$).

6.2 Results

To test the performance of the vantage writer method, several runs of K-fold cross validation were carried out, varying the number of selected vantage writers: $n = 2, 4, 5, 50$. In each run, the cross validation iterated four times ($k = 4$), where in every iteration 25% of the data was available for training and 75% for testing. The data was split such that all pages of each writer were always in the same part. Testing was done for verification and identification separately.

	vantage 1 <i>Henk z aan ke Als het wezen Het ure Henk z</i>	vantage 2 <i>Piet is zeer Het is rone er een zee loopt naar op de grond</i>	vantage 3 <i>Een vlieger stapt nu hikken i nu, du Na ontde</i>	vantage 4 <i>Bob, Dav landen E Griekenlan Tij beao Bunke</i>	vantage 5 <i>Tied eijn og eer s Oto g Herke st a a n</i>
doc 1 <i>Bob, Dav landen E Italie</i>	0.153	0.157	0.321	0.104	0.339
doc 2 <i>Bob, Dav landen E Griekenlan</i>	0.286	0.062	0.591	0.187	0.608
doc 3 <i>Bob, Dav landen Italie</i>	0.294	0.123	0.799	0.373	0.579
...

Figure 6.2: Example vantage profiles based on five vantage writers in the Firemaker dataset.

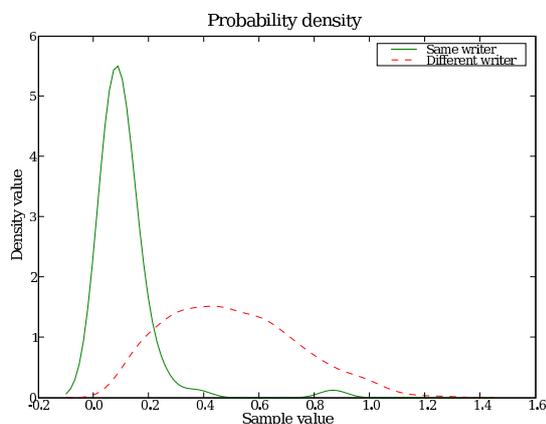


Figure 6.3: Model of hinge-5-vantage profile distances in Firemaker dataset, smoothed using Parzen windowing with a Gaussian kernel.

For writer *verification*, the train part of the dataset was used to select vantage writers, compute vantage profiles and determine a verification threshold as described in 6.1.3 – 6.1.5. Then, the found vantage writers and threshold were used to perform writer verification in the same way, but on the *test* set. The TP and TN were recorded. The results for the *Firemaker* and *NFI* dataset are shown in Table 6.1. For comparison, the table also includes a line with the performance of the bare *Hinge* feature vector, without the indirection induced by vantage writers. The tables show that the performance using vantage profiles is similar to the performance using the *Hinge* feature directly, and that the number of vantage writers does not have much influence. The ratio between TP and TN shifts slightly as the number of vantage writers increases; this is probably an effect of the Parzen smoothing. The effect of smoothing can still be optimized by changing its parameters.

Similarly, for writer *identification*, the train part of the dataset was used to select vantage writers and compute vantage profiles as described in 6.1.3, 6.1.4 and 6.1.6. Then, the found vantage writers were used to perform writer identification in the same way on the test set. Top-1, top-10 and top-100 performance were recorded. The results for the *Firemaker* and *NFI* dataset are shown in Table 6.2 and 6.3 respectively. For comparison, the table also includes a line with the performance of the bare *Hinge* feature vector, without the indirection induced by vantage writers. It is clear that the vantage writer

	Firemaker		NFI	
	TP	TN	TP	TN
hinge	96.1%	83.5%	79.0%	73.9%
hinge-2-vantage profile	91.4%	85.9%	74.5%	75.8%
hinge-4-vantage profile	92.0%	89.7%	75.1%	77.8%
hinge-5-vantage profile	90.8%	90.5%	75.1%	77.8%
hinge-50-vantage profile	88.7%	91.3%	75.0%	77.6%

Table 6.1: Average verification results for the Firemaker dataset (378 pages, 189 writers) and NFI dataset (3561 pages, 805 writers).

	Top-1	Top-10	Top-100
hinge	67.3%	89.0%	98.3%
hinge-2-vantage profile	12.4%	48.9%	94.8%
hinge-4-vantage profile	30.6%	72.3%	96.2%
hinge-5-vantage profile	36.5%	75.3%	97.3%
hinge-50-vantage profile	41.0%	76.8%	96.5%

Table 6.2: Average identification results for the Firemaker dataset (378 pages, 189 writers).

method does not work as well for identification as it does for verification.

6.3 Conclusion

In this paper a method was proposed that is a step toward explainable automatic writer verification and identification. This transforms the “black box” that automatic systems usually are into a more transparent one, since basic components can be shown to the user: a vantage profile and pieces of text written by the vantage writers on which the vantage profile is based. The output is currently a *same writer / different writer* verdict (for verification) or a hit list (for identification), accompanied by vantage profiles. These show the degree of dissimilarity to the handwriting of each of the vantage writers. The system could be made even more transparent when the dissimilarities are replaced by probabilities. This could be done when the measurements are repeated in multiple pieces

	Top-1	Top-10	Top-100
Hinge	53.5%	77.1%	92.2%
Hinge-2-vantage profile	2.8%	17.4%	61.2%
Hinge-4-vantage profile	14.1%	42.8%	80.6%
Hinge-5-vantage profile	18.2%	48.2%	82.6%
Hinge-50-vantage profile	28.3%	58.5%	85.6%

Table 6.3: Average identification results for the NFI dataset (3561 pages, 805 writers).

of text from the same writer. This is left as future work.

The results show that the method works very well for writer *verification*, even with as few as two vantage writers, yielding only a small performance drop relative to the original input *Hinge* feature vector, while representing writers in only two dimensions. This is an enormous dimensionality reduction, which by itself can be a reason to use this method. The performance results for *identification* are less encouraging, but our approach may still be useful because of its explainable basis. In any case, there is room for improvements. Better preprocessing should enable the use more of the authentic forensic NFI samples. Furthermore, the vantage profile might be constructed based on other basic features; the vantage writers may be selected more thoroughly using more extensive stochastic sampling and other distance measures may be evaluated. The set of vantage writers could also be manually selected by experts, representing specific groups of writers distinguished by criteria like sex, handedness, age, nationality and script type. We have already tried unsupervised clustering using a Kohonen self-organizing map (SOM), but that did not result in better performance than using stochastically sampled optimal vantage writers. Additionally, its clustering-based vantage centroids did not contribute to the explainability of results as was evident from discussions with a forensic expert. Applicability may further improve by transforming current writer vantage-distances to reliable probability estimates. Concluding, while there is still room for improvement, the proposed method introduces a feature transformation for increased explainability that could be part of a system for robust and applicable handwriting biometrics.

Two objectives were central to this dissertation:

1. Analysis of robustness of handwriting biometrics;
2. Introduction of methods to make handwriting biometrics more robust and applicable.

In the next two sections the main findings regarding these objectives will be summarized, followed by propositions for future work.

7.1 Robustness analysis

In part I, the robustness of well-performing methods for realistic input was tested. Three specific types of difficult input were distinguished, each in one chapter. Chapter 2 discussed the robustness for text scarcity. It was shown that robust classification is possible if at least 100 handwritten characters are present in the input documents. This was tested by gradually increasing the amount of handwritten text on the pages while monitoring the change of performance in writer verification and identification. Increasing the amount consistently increased performance, even when the amount of text was increased in only one of the two documents in the pairwise comparisons. The best tested feature (*Hinge*) performed near-optimal when the text contained 100 characters. Below this amount, the performance dropped. For features that are less powerful, a reasonable minimum is 200 characters.

In Chapter 3 it was shown that a realistic fraction of crossed-out text can be admitted for robust handwriting biometrics. A simple method to automatically identify and remove crossed-out text was applied, which was able to remove 47% of crossed-out text while preserving 99% of the normal text. After removal, writer verification and identification performance based on the *Hinge* feature remained nearly equal.

Chapter 4 discussed robustness for a frequent method of handwriting disguise: a change of slant. It was shown that handwriting biometrics is not fully robust for this kind of disguise, despite the facts that the slant angle is not an essential writer-specific feature and that slant change seems to be simple to correct. Change of slant has effect on other aspects of the handwriting, making it less identifiable than natural handwriting. This was found in a series of writer verification experiments using four statistical features.

7.2 New methods for robust and applicable handwriting biometrics

Part II constitutes objective 2: the introduction of methods to make handwriting biometrics robust and applicable. In chapter 5 two new, powerful features for writer identification were introduced, which are inspired by modern paleography: the *Quill* and *QuillHinge* features. These features measure variations in the width of the ink trace in relation to the trace direction. The features were tested on four datasets, including a challenging collection of 118 medieval writings, allegedly written by 18 scribes, captured on freehand photos. The methods perform very well, even on that challenging material: 75% top-1 writer-identification performance. On datasets of contemporary handwriting, top-1 accuracy for *QuillHinge* was 86–97%. This implies that the width of the ink trace contains discriminative information. The new feature extraction methods can be used as part of the collection of methods in a robust system for handwriting biometrics.

In Chapter 6 a method was introduced to make handwriting biometrics more explainable, which is particularly necessary for writer verification. The principle is that every handwriting can be expressed as a relative position with respect to a limited number of prototypical handwritings. This makes it possible to visualize and explain the found differences between the writings.

As a proof-of-concept, the feasibility of applicable handwriting biometrics is demonstrated in the form of a graphical computer application for robust and applicable writer identification: GIWIS. The program requires little user intervention because of the use of methods based on statistical pattern recognition and effective preprocessing steps. It contains the powerful *Quill* and *QuillHinge* features which were described in Chapter 5. It can be used by the police to quickly find suspects, and by historians to group documents likely written by the same writer. It is described in Appendix A.

Handwriting biometrics, in the form of writer *identification*, is now applicable for

historians and the police. In addition, a few steps were taken to make writer *verification* applicable too, for application in the forensic domain. Still, many directions for further improvements remain.

7.3 Future work

Robustness and applicability can be increased further in many ways. An obvious approach is to further increase recognition performance by continuing to invent good features, but current features such as *Hinge*, *Fraglets*, *Quill* and *QuillHinge* are already good and reaching 100% might be impossible. Therefore, we suggest to focus on other aspects.

First of all, it must become a standard to use realistic datasets for testing systems. Currently, laboratory datasets are the standard. These are useful to test the theoretical power of computational features in isolation, but not for testing complete systems, since such datasets lack the variability found in the real world. In the case of modern handwriting, datasets for testing must contain real-world forensic material, not only writing samples collected from suspects, but also heterogeneous real questioned documents. This should include a variety of materials, written using a variety of pens, and including disguised handwriting. A similar argument holds for testing on historical handwriting. In this dissertation a somewhat heterogeneous dataset was used, but the authorship of each document was not known for sure. We suggest to collect more heterogeneous datasets that include certain authorship labels. Testing on such datasets would give a better impression of real-world performance. It is a challenge to acquire such data in conjunction with reliable authorship information.

One of the problems that such real-world data imposes is foreground-background segmentation: it is common that not only the handwritten text is dark, but other objects as well, such as lineation, preprinted text or pictures, stains, and wrinkles. These must be removed automatically or at least semi-automatically to make automatic methods usable. Robustness for input quality has a limit, but this is acceptable if a clear threshold on input quality is known, and the zone of tolerance is large. Ideally, the system should automatically *reject* input that does not meet the quality criterion.

An issue that was not covered in this dissertation is within-writer variability modeling. We think it should be incorporated in a robust system for handwriting biometrics. Experts of handwriting use it implicitly, and others have implemented automatic ap-

proaches such as Gaussian mixture models (GMM) [83], hidden Markov models (HMM) [82], or simply by using weighted euclidean distance measures (WED). The difficulty with variability modeling is that it requires more training data: multiple input samples per known writer are needed to estimate the variability. Alternatively, it is possible to obtain multiple samples from a single document by splitting the document in smaller regions such as text lines, which can only work on neatly written text. A related issue is multi-stability: many people can stably write in more than one way. An obvious example is a distinction between cursive and capital letters, but other variations are also common. Simply averaging the features of all kinds of handwriting of each subject might result in false feature centroids.

The most problematic form of within-writer variation is disguise. A person can change his or her handwriting in an attempt to avoid identification, which is probably common in forensic cases. Computed features are not robust for this. The features may have some discriminating power in such cases, but thorough testing on disguised handwriting is needed before automatic writer verification can be trusted. In this dissertation we have investigated the frequent form of disguise by slant manipulation, but the effects of all other forms of disguise remain unknown. This also applies to an alternative method to change one's writing style: forgery, simulating someone else's handwriting. Recently a writer verification system was developed that showed an impressively low equal-error rate (EER) of only 4% when it was tested on a mix of genuine and forged handwriting [82].

The ultimate challenge for the distant future is the incorporation of handwriting *recognition*: automatic transcription of the textual contents. When this field has matured, a new window of opportunities can be opened. Most importantly, it would allow for word- and character-wise comparison. This is a specific and detailed analysis, which promises even higher performance. Handwriting recognition can also be used to automate preprocessing and ROI segmentation, by optimizing the parameters for readability of the processed image. A possible angle is to use Hidden Markov models (HMM) [82], which naturally combine writer identification with handwriting recognition. However, HMMs require that the text lines are well-separable and cannot cope with unconstrained handwriting.

For the near future, a different approach to support forensic document examiners is already possible: an application that presents a virtual *Oslo lineup*. Given a questioned document, the system should confront the FDE with a lineup of a limited number (for

example, 9) documents containing similar handwriting (“distractors”) plus the document written by the suspect. The task for the FDE is then to decide if the handwriting in one of the ten documents is remarkably similar to that in the questioned document. An open question is for how long handwriting will be used in the future, as digital communication is becoming more important. In any case, handwriting biometrics on contemporary handwriting is useful today and it is useful with historical handwriting as long as such handwriting is studied.

Appendix A

GIWIS: a robust and applicable writer identification tool

As a proof-of-concept of the feasibility of robust and applicable handwriting biometrics, a graphical computer program for writer identification has been developed: GIWIS (Groningen Intelligent Writer Identification System). It is a user-friendly tool for writer identification, intended to be used by non-technical handwriting researchers, including police members and historical researchers. The application enables these professionals to search for documents in a database based on handwriting similarity, possibly saving a considerable amount of time.

A few related graphical programs have been introduced before:

- CEDAR-FOX: a Windows program for handwriting recognition, writer verification and writer identification. It is intended for forensic experts. Developed by CEDAR/CedarTech, USA. [95]
- FISH (Forensic Identification System of Handwriting): a VMS/VWS/X11 program intended for forensic experts. Developed by BKA, Germany. [77, 88];
- GRAWIS (Groningen automatic writer identification system): a web-based writer identification demo based on precomputed results. Developed at ALICE in Groningen, the Netherlands. [17]
- Hand Analyser: a prototype system designed to aid in the identification of handwriting, particularly medieval. Developed at the University of Cambridge, UK. [98, 99]
- SCRIPT: a Windows program intended for non-expert users. Developed by NIFO/-TNO, the Netherlands. [28, 88];

- SPI (System for Paleographic Inspections): this system performs writer identification based on individual characters. The characters have to be segmented semi-automatically; the user has to point and click on them. A newer version (JSPI) is currently being developed. Developed in Italy. [1]
- Ulysse: A French windows application for word spotting in ancient documents. Developed in France. [46]
- WANDA: a framework for forensic handwriting examination and writer identification. It consists of a plug-in architecture with separate components for clients with a graphical user interface and servers for data processing. It can perform writer identification based on manual and automatic feature measurements [40].

In most of these programs, performing a search query is a laborious process. The programs require the user to answer many questions about the handwriting or to do the measurements manually using the mouse. GIWIS is unique in the sense that it is robust and applicable, as will be shown in the next sections.

A.1 Design principles

GIWIS was designed to be *robust* by combining different powerful techniques:

- Preprocessing consists of multiple steps, each compensating for a specific difficulty in the input material.
- Multiple powerful feature extraction methods are available. These features can be selected and combined to achieve a combination that works best, avoiding an overly strong dependence on a single information source.
- The well-performing feature extraction method *Quill-Hinge* is included, which is based on a sound principle and is not sensitive to parameter settings.
- The feature extraction methods are text-independent, which ensures that input documents with any textual content can be admitted.

In addition, GIWIS was designed to be *applicable* by minimizing required user interaction and by visualizing the inner workings:

- The feature extraction methods are based on statistical pattern recognition, therefore these do not require any user interaction.
- Preprocessing is almost fully automated. The only human input is needed for ROI selection: selecting the text region by dragging four corner marks, which is simple and takes little time. There is no need for manual image editing in an external image manipulation program, as image processing techniques are available in the tool itself. These techniques are applied automatically, thus the user is not required to tamper with image details, but the process can be controlled manually if desired.
- The inner workings of the system are explained: real-time visualizations of the result of preprocessing and feature extraction are available. (Visualization using *vantage writers* is not supported yet.)

The functionality of GIWIS is described in the next section.

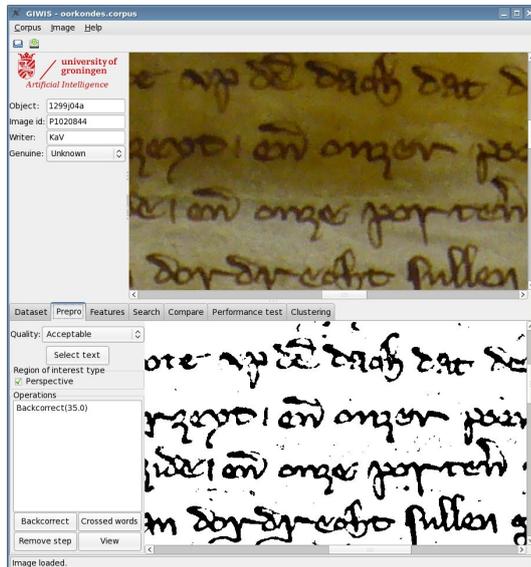
A.2 Functionality

The graphical interface of GIWIS consists of a top part and a bottom part; see Figure A.1. In the top part, the currently selected image is shown together with three properties: the object name or code, an image identifier (to distinguish between different images of the same object) and the writer name or code, if known. The menu bar enables loading and saving datasets and acquiring images by taking photos or scanning.

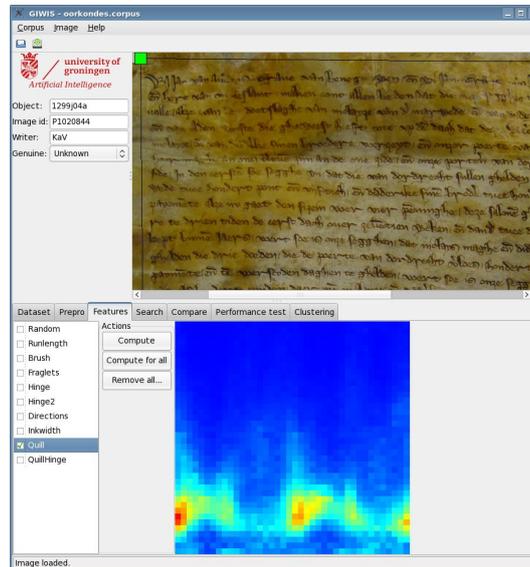
The bottom part consists of a series of tabs, which represent several steps of writer identification and other functionality. Clicking a tab results in the appearance of a dedicated sheet of controls and information; some of these are shown in Figure A.1. The next subsections each describe one of these tabs.

A.2.1 Tab 1: Dataset

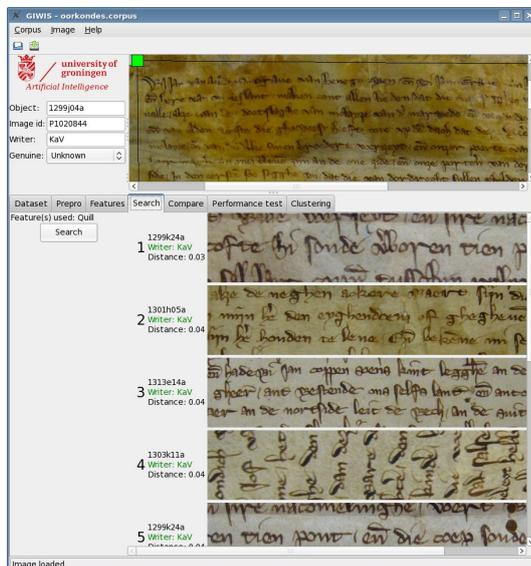
On the *Dataset* tab (not shown), the list of available images in the loaded dataset is shown. The list shows meta-information of each image, such as the writer name/code, if these have been entered beforehand. Also, the user can make selections to include or exclude categories of images based on meta-information. After selecting an image from the dataset, it will be shown in the top panel. The image can be panned by dragging using the mouse, and zoomed by using the mouse wheel. A four-sided polygon with corner



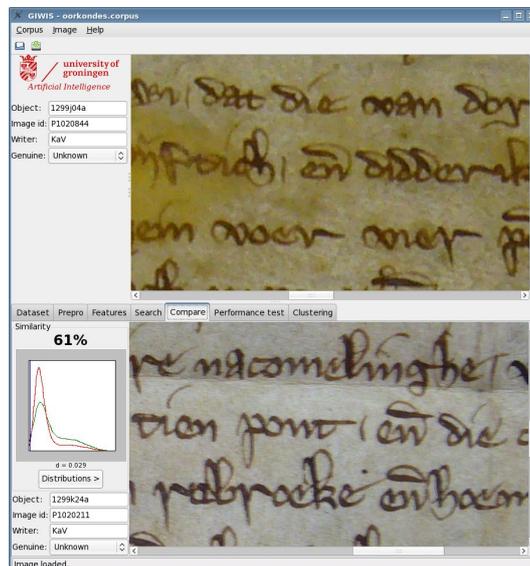
(a) Preprocessing



(b) Feature selection



(c) Search



(d) Visual comparison

Figure A.1: Screenshots of GIWIS

markers is shown on top of the image, which represents the region of interest. Only this region will be regarded in the processing. The user can modify the region by dragging corner markers with the mouse.

A.2.2 Tab 2: Preprocessing

The *Prepro* tab, shown in Figure A.1(a), displays the result of image preprocessing. Preprocessing comprises the following steps:

- *Perspective correction*: A check box is provided to indicate that the picture was taken at an angle instead of straight from above. If it is switched on, the perspective distortion is corrected by a reverse perspective projection. The parameters are derived from the positions of the four vertices of the ROI. The result is a rectangular image containing a version of the ROI, stretched using bilinear interpolation.
- *Automatic scaling*: This is necessary for images with varying camera distance. The image scale is estimated from the height of the text lines in the images, assuming that the true height of the text lines is equal in all documents. The text line height is determined by measuring the median width of the peaks in the smoothed horizontal projection profile of dark pixels. The image is then scaled to match a standard text line height of 50 pixels.
- *High-pass filtering*: Gradual intensity variations are canceled by applying a high-pass filter, after grayscale conversion. This is implemented by blurring the image and subtracting that from the original image.
- *Otsu thresholding*: The last preprocessing step is to binarize the image using Otsu thresholding [75].

If the result looks bad, the user can decide to exclude the image from the database by changing the *quality* setting to *Reject*, on the panel on left.

A.2.3 Tab 3: Feature selection

The next tab, *Features*, shows a list of available feature extraction methods on the left side. See Figure A.1(b). The following features are available: *Random* (a feature vector consisting of random numbers irrespective of the image), *Runlength* [2], *Brush* [87], *Fraglets* [86], *Hinge* [21, 19], *Hinge2* [21, 19] (another implementation of the *Hinge*

feature with a custom parameter value for the length of the hinge ‘legs’), *Directions* [16], *Inkwidth*, *Quill*, and *QuillHinge* (the latter three are introduced in chapter 5).

By default, one of the features is selected. The user can change the default selection by checking one or more boxes. If more than one box is selected, the combination of features is weighted by averaging distance values. After clicking the name of a feature, a visualization of the feature values shows on the right side.

A.2.4 Tab 4: Search

The *Search* tab provides one button: ‘Search’. See Figure A.1(c). After clicking this button, writer identification is performed with the image shown at the top as the query document. It is compared to all images in the dataset by means of comparing feature vectors. If these feature vectors have not been computed before, they are computed on the fly, after preprocessing. Thus, this step only takes significant time the first time. After this process, the top-10 hit list is shown as a list of snippets. If the button is clicked without any prior ROI adjustments or feature selection, default settings will be used.

A.2.5 Tab 5: Visual comparison

The snippets in the hit list can be closely compared to the original image by clicking on one of them. This will highlight the *Compare* tab, and show the selected snippet on the bottom in full size; see Figure A.1(d). An alternative way to get a comparison image in this tab, is by selecting one in the *Dataset* tab and hitting the ‘Compare’ button. The panel on the left side also shows probability distributions and a writer match probability estimation based on these distributions. This match probability must be interpreted with care, as the probability distributions are by default only based on the current dataset and smoothed using parameters that may not be optimal.

A.2.6 Tab 6: Performance results

The *Performance test* tab includes a button to test the writer identification performance of the selected features on the current dataset. The results are shown in the white panel on the right side.

A.2.7 Tab 7: Clustering

The last tab, *Cluster*, provides the possibility to cluster the images in the dataset into groups using k-means clustering.

A.3 Architecture

The program consists of two parts: a user interface and a model. The user interface was written in Python using wxWidgets. The model performs the actual work. It is mainly written in Python, but performance-critical parts were written in C++. The model can be used in conjunction with the graphical user interface or it can be controlled without user interaction by a separate Python script.

The input image files are found in a specified directory; each image is accompanied by an XML file in another directory. The XML files contain metadata, including the writer label, a list of required preprocessing steps, and computed feature values, if known, and this information is updated whenever new information is available. The most common task of the model is to compute feature values given the name of a feature and its parameters. If the feature values for this request are not available in the XML file, the image is preprocessed according to the steps mentioned in the XML file, and the feature values are extracted and added to the XML file. The changes are only saved when the user chooses 'Save'.

A.4 Discussion

GIWIS is robust and applicable in real-life conditions. This has shown in results of several research projects:

- History research on 14th-century Dutch documents [91, 92];
- History research on 15th-century French documents [4];
- History research on 16th-century German documents [10];
- Language history research on 17th-century and 18th-century Dutch documents [74];

In the latter two cases, GIWIS was controlled by a script. The program was also demonstrated in several convincing demonstrations for real-time writer identification with a digital camera, despite the conditions of bad illumination and low image quality. With GIWIS, a tool for handwriting biometrics has become available that is robust and applicable.

Bibliography

- [1] F. Aioli, M. Simi, D. Sona, A. Sperduti, A. Starita, and G. Zaccagnini. SPI: A system for paleographic inspections. In *Notiziario AI*IA*, volume 4, pages 34–38, 1999.
- [2] B. Arazi. Handwriting identification by means of run-length measurements. *SMC*, 7:878–881, 1977.
- [3] J. Arlandis, J. C. Perez-Cortes, and J. Cano. Rejection strategies and confidence measures for a k-nn classifier in an ocr task. In *ICPR*, 2002.
- [4] M. Aussems and A. Brink. Digital palaeography. In M. Rehbein, P. Sahle, and T. Schassan, editors, *Kodikologie und Paläographie im Digitalen Zeitalter / Codicology and Palaeography in the Digital Age*, volume 2 of *Schriftenreihe des Instituts für Dokumentologie und Editorik*, pages 293–308. Books on Demand, Norderstedt, 2009.
- [5] H. S. Baird. The state of the art of document image degradation modeling. In *Proc. of 4 th IAPR International Workshop on Document Analysis Systems*, pages 1–16, 2000.
- [6] A. Bensefia, T. Paquet, and L. Heutte. A writer identification and verification system. *Pattern Recognition Letters*, 26(13):2080–2092, 2005.
- [7] R. Bertolami, S. Uchida, M. Zimmermann, and H. Bunke. Non-uniform slant correction for handwritten text line recognition. In *Proc. of 9th International Conference on Document Analysis and Recognition*, pages 18–22, 2007.

- [8] V. Blankers, R. Niels, and L. Vuurpijl. Writer identification by means of explainable features: shapes of loop and lead-in strokes. In *Proceedings of the 19th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2007)*, pages 17–24, 2007.
- [9] R. M. Bozinovic and S. N. Srihari. Off-line cursive script word recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 11(1):68–83, 1989.
- [10] A. Breitenbach and M. Hömberg. Research project “Schriftlichkeit in süddeutschen Frauenklöstern” at Heinrich-Heine-Universität Düsseldorf. <http://www.uni-muenster.de/Geschichte/hist-sem/MA-G/L3/forschen/DFGProjekt.html>.
- [11] A. Brink, M. Bulacu, and L. Schomaker. How much handwritten text is needed for text-independent writer verification and identification. In *Proc. of the 19th International Conference on Pattern Recognition (ICPR)*. IEEE, 2008. doi 10.1109/ICPR.2008.4761908.
- [12] A. Brink, R. Niels, R. van Batenburg, C. van den Heuvel, and L. Schomaker. Towards robust writer verification by correcting unnatural slant. *Pattern Recognition Lett.*, 32:449–457, 2011.
- [13] A. Brink, L. Schomaker, and M. Bulacu. Towards explainable writer verification and identification using vantage writers. In *ICDAR*, pages 824–828, 2007.
- [14] A. Brink, J. Smit, M. Bulacu, and L. Schomaker. Writer identification using directional ink-trace width measurements. *Pattern Recognition*, (45):162–171, 2012. doi: 10.1016/j.patcog.2011.07.005.
- [15] A. Brink, H. van der Klauw, and L. Schomaker. Automatic removal of crossed-out handwritten text and the effect on writer verification and identification. In B. Yanikoglu and K. Berkner, editors, *Proceedings of Document Recognition and Retrieval XV, IS&T/SPIE International Symposium on Electronic Imaging*, 2008.
- [16] M. Bulacu and L. Schomaker. Writer style from oriented edge fragments. In *CAIP 2003*, LNCS, pages 460–469. Springer, 2003.
- [17] M. Bulacu and L. Schomaker. GRAWIS: Groningen automatic writer identification system. In *Proc. of 17th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC)*, pages 413–414, Brussels, Belgium, October 2005.

- [18] M. Bulacu and L. Schomaker. Automatic handwriting identification on medieval documents. In *Proc. of 14th Int. Conf. on Image Analysis and Processing (ICIAP 2007)*, pages 279–284, Modena, Italy, 11-13 September 2007. IEEE Computer Society.
- [19] M. Bulacu and L. Schomaker. Text-independent writer identification and verification using textural and allographic features. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 29(4):701–717, 2007.
- [20] M. Bulacu, L. Schomaker, and L. Vuurpijl. Writer identification using edge-based directional features. In *ICDAR 2003*, pages 937–941, 2003.
- [21] M. L. Bulacu. *Statistical Pattern Recognition for Automatic Writer Identification and Verification*. PhD thesis, University of Groningen, 2007.
- [22] J. Burgers. *De paleografie van de documentaire bronnen in Holland en Zeeland in de dertiende eeuw*. Leuven: Peeters, 1995.
- [23] S. F. Bush, J. Hershey, and K. Vosburgh. Brittle system analysis, 1999. arXiv:cs/9904016v1.
- [24] S. Cha and C. C. Tappert. Automatic detection of handwriting forgery. In *Proc. of the 8th IWFHR*, pages 264–267, 2002.
- [25] S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *International journal of mathematical models and methods in applied sciences*, 1:300–307, 2007.
- [26] J.-P. Crettez. A set of handwriting families: style recognition. In *Proc. of the 3rd ICDAR*, pages 489–494, 1995.
- [27] J. Daugman. Identification of medieval scribal handwriting. Suggested student project, 2009. Website consulted in March 2009.
- [28] W. de Jong, L. K. van der Kooij, and D. Schmidt. Computer aided analysis of handwriting, the NIFO-TNO approach. In *Proc. 4th European Handwriting Conference for Police and Government Handwriting Experts*, 1994.

- [29] A. Derolez. Possibilités et limites d'une paléographie quantitative. In P. Defosse, editor, *Hommages à Carl Deroux*, volume 279, pages 98–102. Brussels: Editions Latomus, 2003.
- [30] D. S. Doermann and A. Rosenfeld. Recovery of temporal information from static images of handwriting. *International Journal of Computer Vision*, 15:143–164, 1995.
- [31] E. Dooijes. Decomposition of the pen displacement signal in the analysis of handwriting motorics. In I.T. Young et al., editor, *Signal processing III: Theories and Applications*, pages 1343–1345, 1986.
- [32] M. Drogin. *Medieval calligraphy: Its history and technique*. Dover Publications, 1989.
- [33] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2nd edition edition, 2000.
- [34] T. Fawcett. An introduction to roc analysis. *Pattern Recogn. Lett.*, 27(8):861–874, 2006.
- [35] B. Found and D. Rogers. A consideration of the theoretical basis of forensic handwriting examination: The application of 'complexity theory' to understanding the basis of handwriting identification. *International Journal of Forensic Document Examiners*, 4:109–118, 1998.
- [36] B. Found and D. Rogers. Problem types of questioned handwritten text for forensic document examiners. In *Advances in Graphonomics: Proceedings of the 12th Biennial Conference of the International Graphonomics Society*, pages 8–12, 2005.
- [37] B. Found and D. Rogers. The relative strength of forensic document examiners' identification and elimination opinions on individuals' natural writings. In *Proc. of the 12th Biennial Conference of the International Graphonomics Society*, pages 157–161, 2005.
- [38] K. Franke. *The influence of physical and biomechanical processes on the ink trace - Methodological foundations for the forensic analysis of signatures*. PhD thesis, University of Groningen, The Netherlands., 2005.

- [39] K. Franke and M. Köppen. A framework for document pre-processing in forensic handwriting analysis. In L. Schomaker and L. Vuurpijl, editors, *Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition*, pages 73–82, September 2000.
- [40] K. Franke, L. Schomaker, L. Vuurpijl, M. van Erp, and I. Guyon. Wanda: A common ground for forensic handwriting examination and writer identification. In *ENFHEX news - Bulletin of the European Network of Forensic Handwriting Experts*, number 1/04, pages 23–47, 2004.
- [41] K. Franke and S. N. Srihari. Computational forensics: Towards hybrid-intelligent crime investigation. In *Proc. Int. Symposium on Information Assurance and Security / Int. Workshop on Computational Forensics (IWCF 2007)*, pages 383–386, Manchester, England, Aug. 2007. IEEE-CS Press.
- [42] H. Fujisawa. Robustness design of industrial strength recognition systems. In B. B. Chaudhuri, editor, *Digital document processing: major directions and recent advances*, chapter 9, pages 185–212. Springer-Verlag, 2006.
- [43] H. Fujisawa. Forty years of research in character and document recognition – an industrial perspective. *Pattern Recognition*, 41:2435–2446, 2008.
- [44] U. Garain and T. Paquet. Off-line multi-script writer identification using AR coefficients. In *10th International Conference on Document Analysis and Recognition (ICDAR)*, pages 991–995, 2009.
- [45] L. Gilissen. *L’expertise des écritures médiévales: recherche d’une méthode avec application à un manuscrit du XIème siècle: le lectionnaire de Lobbes (Codex Bruxellensis 18018)*. Ghent: Story-Scientia, 1973.
- [46] GRAPHEM: Projet ANR. Ulysse 0.3g09 pour windows. Downloadable software on research project website, 2009. <http://liris.cnrs.fr/graphem/?p=73>.
- [47] J. K. Guo. *Forgery Detection by Local Correspondence*. PhD thesis, Center for Automation Research, University of Maryland, 2000.
- [48] H. Hardy and W. Fagel. Methodological aspects of handwriting identification. *Journal of Forensic Document Examination*, 8:33–69, 1995.

- [49] J. J. Harris. Disguised handwriting. *Journal Of Criminal Law, Criminology And Police Science*, 43:685–689, 1953.
- [50] D. Hearn and M. P. Baker. *Computer graphics, C version*. Prentice Hall, 1997.
- [51] S. Howard. The steel pen and the modern line of beauty. *Technology and Culture*, 26(4):785–798, 1985.
- [52] L. Huang, G. Wan, and C. Liu. An improved parallel thinning algorithm. In *ICDAR '03: Proceedings of the Seventh International Conference on Document Analysis and Recognition*, page 780, Washington, DC, USA, 2003. IEEE Computer Society.
- [53] R. A. Huber and A. M. Headrick. *Handwriting identification: facts and fundamentals*. CRC, 1999.
- [54] T. Joachims. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.
- [55] S. Kabra, H. Srinivasan, C. Huang, and S. Srihari. On computing the strength of evidence for writer verification. In *ICDAR*, pages 844–848, 2007.
- [56] M. Kam, G. Fielding, and R. Conn. Writer identification by professional document examiners. *Journal of Forensic Sciences*, 42:778–785, 1997.
- [57] E. Kavallieratou, N. Fakotakis, and G. Kokkinakis. Slant estimation algorithm for OCR systems. *Pattern Recognition*, 34:2515–2522, 2001.
- [58] K. M. Koppenhaver. *Forensic Document Examination: Principles and Practice*. Humana Press, 2007.
- [59] E. Lecolinet, L. Likforman-Sulem, L. Robert, F. Role, and J. Lebrave. An integrated reading and editing environment for scholarly research on literary works and their handwritten sources. *Proceedings of the third ACM conference on Digital libraries*, pages 144–151, 1998.
- [60] W. Lidwell, K. Holden, and J. Butler. *Universal Principles of Design*. Rockport Publishers, 2010.

- [61] H. Love. *Attributing Authorship: An Introduction*. Cambridge University Press, 2002.
- [62] H. Lv, W. Wang, C. Wang, and Q. Zhuo. Off-line chinese signature verification based on support vector machines. *Pattern Recogn. Lett.*, 26(15):2390–2399, 2005.
- [63] F. Maarse. *The study of handwriting movement: peripheral models and signal processing techniques*. PhD thesis, Katholieke Universiteit Nijmegen, 1987.
- [64] F. J. Maarse and A. J. Thomassen. Produced and perceived writing slant: difference between up and down strokes. *Acta psychologica*, 54:131–147, 1983.
- [65] U. Marti and H. Bunke. A full english sentence database for off-line handwriting recognition. In *Proc. of the 5th ICDAR*, pages 705–708, 1999.
- [66] R. N. Morris. *Forensic Handwriting Identification*. Academic Press, 2000.
- [67] J. Nickell. *Pen, Ink, and Evidence: A Study of Writing and Writing Materials for the Penman, Collector, and Document Detective*. University Press of Kentucky, 1990.
- [68] J. Nickell. *Detecting forgery: forensic investigation of documents*. University Press of Kentucky, 2007.
- [69] S. Nicolas, T. Paquet, and L. Heutte. Markov random field models to extract the layout of complex handwritten documents. In *Proc. of the 10th IWFHR*, 2006.
- [70] R. Niels. *Allograph based writer identification, handwriting analysis and character recognition*. PhD thesis, Donders Centre for Brain, Behaviour and Cognition, Radboud University Nijmegen, the Netherlands, 2010.
- [71] R. Niels, F. Grootjen, and L. Vuurpijl. Writer identification through information retrieval: the allograph weight vector. In *11th International Conference on the Frontiers of Handwriting Recognition (ICFHR)*, pages 481–486, 2008.
- [72] R. Niels and L. Vuurpijl. Generating copybooks from consistent handwriting styles. In *Proc. of the 9th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1009–1013, 2007.

- [73] R. Niels, L. Vuurpijl, and L. Schomaker. Automatic allograph matching in forensic writer identification. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 21(1):61–81, 2007.
- [74] J. Nobels and M. van der Wal. Tackling the writer-sender problem: the newly developed Leiden identification procedure (LIP). *Historical Sociolinguistics and Sociohistorical Linguistics*, 9, 2009.
- [75] N. Otsu. A threshold selection method from gray-level histograms. *IEEE trans. on Systems, Man and Cybernetics*, 9(1):62–66, 1979.
- [76] E. Pekalska and R. Duin. Dissimilarity representations allow for building good classifiers. *Pattern Recognition Letters*, 23(8):943–956, 2002.
- [77] M. Philipp. Fakten zu FISH, das forensische informations-system handschriften des bundeskriminalamtes - eine analyse nach über 5 jahren wirkbetrieb. Technical report, Kriminaltechnisches Institut 53, Bundeskriminalamt, Thaerstraße 11, 65173 Wiesbaden, 1996.
- [78] R. Plamondon and G. Lorette. Automatic signature verification and writer identification - the state of the art. *Pattern Recog.*, 22(2):107–131, 1989.
- [79] F. Provost and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42:203–231, 2001.
- [80] H. Said, T. Tan, and K. Baker. Personal identification based on handwriting. *Pattern Recognition*, 33:149–160, 2000.
- [81] T. Scheidat, C. Vielhauer, and J. Dittman. Handwriting verification – comparison of a multi-algorithmic and a multi-semantic approach. *Image and Vision Computing*, 27:269–278, 2009.
- [82] A. Schlapbach and H. Bunke. A writer identification and verification system using hmm based recognizers. *Pattern Analysis & Applications*, 10(1):33–43, February 2007.
- [83] A. Schlapbach and H. Bunke. Off-line writer identification and verification using gaussian mixture models. In S. Marinai and H. Fujisawa, editors, *Machine Learning in Document Analysis and Recognition*, volume 90 of *Studies in Computational Intelligence (SCI)*, pages 409–428. Springer-Verlag Berlin / Heidelberg, 2008.

- [84] L. Schomaker. Writer identification and verification. In N. Ratha and V. Govindaraju, editors, *Advances in Biometrics: Sensors, Systems and Algorithms*, pages 247–264. Springer-Verlag, 2007.
- [85] L. Schomaker and M. Bulacu. Automatic writer identification using connected-component contours and edge-based features of uppercase western script. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6):787–798, 2004.
- [86] L. Schomaker, M. Bulacu, and K. Franke. Automatic writer identification using fragmented connected-component contours. In F. Kimura and H. Fujisawa, editors, *9th IWFHR*, pages 185–190, Tokyo, Japan, October 2004.
- [87] L. Schomaker, M. Bulacu, and M. van Erp. Sparse-parametric writer identification using heterogeneous feature groups. In *ICIP*, volume 1, pages 545–548, September 2003.
- [88] L. Schomaker and L. Vuurpijl. Forensic writer identification: A benchmark data set and a comparison of two systems. Technical report, NICI, Nijmegen, 2000.
- [89] L. R. B. Schomaker and R. Plamondon. The relation between pen force and pen-point kinematics in handwriting. *Biological Cybernetics*, 63:277–289, 1990.
- [90] I. Siddiqi and N. Vincent. A set of chain code based features for writer recognition. In *10th International Conference on Document Analysis and Recognition (ICDAR)*, pages 981–985, 2009.
- [91] J. Smit. The death of the palaeographer? experiences with the groningen intelligent writer identification system (giwis). *Archiv für Diplomatik*.
- [92] J. Smit. Meten is weten? De toepassing van het Groningen Intelligent Writer Identification System (GIWIS) op Hollandse kanselarijoorkonden, 1299-1345. *Handelingen van de Koninklijke Commissie voor Geschiedenis/Bulletin de la Commission royale d’Histoire*, (176/2):343–359, 2010.
- [93] A. M. Smith. *Printing and writing materials: their evolution*. Philadelphia: Smith, 1904.
- [94] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee. Individuality of handwriting. *Journal of Forensic Sciences*, 47:856–872, 2002.

- [95] S. N. Srihari, C. Huang, and H. Srinivasan. A search engine for handwritten documents. In *Proc. of Document Recognition and Retrieval XII*. SPIE, 2005.
- [96] S. N. Srihari and G. Leedham. A survey of computer methods in forensic document examination. In *Proc. Int. Graphonomics Soc. Conf.*, pages 2–5, 2003.
- [97] E. Stamatatos, N. Fakotakis, and G. Kokkinakis. Computer-based authorship attribution without lexical measures. *Computers and the Humanities*, 35(2):193–214, May 2001.
- [98] P. Stokes. Hand analyser and image viewer software. Downloadable software, 2009. <http://sourceforge.net/projects/handanalyser/>.
- [99] P. A. Stokes. Palaeography and image-processing: Some solutions and problems. *Digital Medievalist*, 3, 2007–2008. <http://www.digitalmedievalist.org/journal/3/stokes/>.
- [100] SWGIT. Recommendations and guidelines for the use of digital image processing in the criminal justice system. *Forensic Science Communications*, 5(1), January 2003.
- [101] G. X. Tan, C. Viard-Gaudin, and A. C. Kot. Automatic writer identification framework for online handwritten documents using character prototypes. *Pattern Recognition*, 42:3313–3323, 2009.
- [102] A. H. Toselli, A. Juan, and E. Vidal. Spontaneous handwriting recognition and classification. In *ICPR*, pages 433–436, 2004.
- [103] A. Vinciarelli and J. Luettin. A new normalization technique for cursive handwritten words. *Pattern Recognition Letters*, 22:1043–1050, 2001.
- [104] J. Vleugels and R. C. Veltkamp. Efficient image retrieval through vantage objects. In *Proc. of the 3rd Int. Conf. on Visual Information and Information Systems VISUAL'99*, LNCS 1614, pages 575–584, 1999.
- [105] V. Vuori. Clustering writing styles with a self-organizing map. In *Proc. of the 8th IWFHR*, pages 345–350, 2002.
- [106] D. H. Wilson. *The Encyclopedia of Calligraphy Techniques*. London: New Burlington Books, 1991.

- [107] S. Yoon, S. Choi, S. Cha, and C. Tappert. Writer profiling using handwriting copybook styles. In *Proceedings of the 8th ICDAR*, pages 600–604. IEEE, 2005.

Samenvatting (summary in Dutch)

Handschrift bevat kenmerken die kunnen verraden wie de schrijver is. Dat komt van pas wanneer de politie wil achterhalen wie de schrijver is van bijvoorbeeld een handgeschreven dreigbrief of een mogelijk valse zelfmoordbrief. Het is ook nuttig voor geschiedenisonderzoek, want veel historische bronnen zijn handgeschreven en soms is het mogelijk om iets te leren over onze geschiedenis door te achterhalen welke documenten door dezelfde persoon geschreven moeten zijn.

Het vergelijken van handschrift gebeurt traditioneel nog met de hand, door experts, op basis van hun kennis en ervaring. Hoewel zij doorgaans zeer kundig zijn zitten hier wel twee nadelen aan: deze aanpak is niet objectief en het is tijdrovend werk. Deze tekortkomingen kunnen goed worden gecompenseerd door de computer, want die heeft tegengestelde kwaliteiten: algoritmes kunnen onbevooroordeeld en razendsnel een enorme hoeveelheid metingen verrichten. Het ligt dus voor de hand om de traditionele handschriftvergelijking te verrijken met een slim stuk gereedschap in de vorm van een computerprogramma dat handschrift kan vergelijken en daar uitspraken over kan doen.

Zulke programmatuur noemen we systemen voor handschriftbiometrie. Er zijn twee typen: schrijververificatie en schrijveridentificatie. Een systeem voor schrijver*verificatie* bepaalt of twee documenten van dezelfde hand zijn; dit kan worden gebruikt om de uitspraak van handschriftdeskundigen te onderbouwen of juist te ontkrachten. Een systeem voor schrijver*identificatie* zoekt op basis van één document in een bestaande collectie documenten naar documenten met vergelijkbaar handschrift en levert daarbij de identiteit van de schrijvers daarvan. Dit kan worden gebruikt door de politie om mogelijke daders te vinden, of door historici om documenten te groeperen die mogelijk van dezelfde hand zijn. In hoofdstuk 1 wordt de basis van zulke systemen voor handschriftbiometrie toegelicht.

Bestaande systemen presteren prima op handschrift dat is geschreven in laboratoriumcondities, maar er is weinig bekend over de prestaties in realistische omstandigheden. In praktijk kan handschrift moeilijkheden bevatten zoals een tekort aan tekst, doorgestreepte woorden en verdraaid handschrift. Mede hierdoor kan schrijver*verificatie* nog

niet zomaar worden toegepast. Bovendien is de werking van huidige systemen voor mensen in de toepassingsgebieden moeilijk te bevatten. Die moeilijkheden zijn minder problematisch bij schrijver*identificatie*, maar ze zorgen wel voor suboptimale prestaties. In dit proefschrift zijn een aantal stappen gezet om handschriftbiometrie meer robuust en toepasbaar te maken, zoals we hieronder zullen samenvatten.

In deel I, dat bestaat uit hoofdstuk 2–4, onderzochten we de robustheid van goed presterende methoden, gebaseerd op statistische patroonherkenning, voor een drietal problemen uit de praktijk: een tekort aan tekst, doorgestreepte woorden en handschriftverdraaiing. In hoofdstuk 2 werd onderzocht hoeveel tekst minimaal nodig is voor betrouwbare resultaten; dit is ongeveer 100 tekens. In hoofdstuk 3 toonden we aan dat handschriftbiometrie nauwelijks lijdt onder de aanwezigheid van doorgestreepte woorden. Tot op zekere hoogte zijn die automatisch te verwijderen, maar ook zonder verwijdering kunnen de documenten worden toegelaten. De lastigste kwestie in dit deel van het proefschrift gaat over verdraaid handschrift. Dit probleem is nog grotendeels onopgelost. In hoofdstuk 4 belichten we een belangrijke verschijningsvorm hiervan: verdraaiing door het veranderen van de hellingshoek. We toonden aan dat de hellingshoek op zich weinig informatief is en dat een veranderde hellingshoek makkelijk te corrigeren is, maar het gecorrigeerde handschrift bleek toch minder herkenbaar dan natuurlijk geschreven tekst. Daaruit blijkt dat een verandering van de helling ook invloed heeft op andere aspecten van het handschrift. Wat betreft handschriftverdraaiing kunnen we handschriftbiometrie dus nog niet robuust noemen.

In deel II, dat bestaat uit hoofdstuk 5–6, worden methoden voorgesteld om handschriftbiometrie meer robuust en toepasbaar te maken. In hoofdstuk 5 worden twee krachtige technieken geïntroduceerd waarmee schrijverspecifieke kenmerken uit het handschrift kunnen worden geëxtraheerd: *Quill* en *QuillHinge*. Deze technieken zijn geïnspireerd door arbeidsintensieve handmatige methoden in de moderne paleografie, namelijk metingen van de breedte en richting van het inktspoor. De gemeten waarden worden onder andere beïnvloed door veranderingen in pendruk en door de manier waarop de pen bij het schrijven is vastgehouden. De prestaties van deze nieuwe technieken op zowel modern als historisch handschrift zijn zeer goed, waarmee wordt aangetoond dat de combinatie van deze twee kenmerken zeer informatief is. Bovendien is handschriftbiometrie hiermee verrijkt met twee krachtige en bruikbare technieken. Vervolgens wordt in hoofdstuk 6 een mogelijke oplossing aangedragen om handschriftbiometrie begrijpelijker te maken, door elk handschrift uit te drukken als een relatieve positie ten opzichte van

een aantal typische handschriften. Dit maakt het makkelijker om de gevonden verschillen tussen handschriften te visualiseren en uit te leggen.

Tenslotte werd in appendix A een volledig functionele applicatie voorgesteld als een demonstratie van robuuste en toepasbare handschriftbiometrie: GIWIS. Het programma vereist slechts weinig gebruikerinteractie en bevat de krachtige technieken *Quill* en *QuillHinge*. Het kan worden gebruikt door de politie om snel verdachten te vinden, en door historici om documenten te groeperen die mogelijk van dezelfde hand zijn.

Author publications

- Axel Brink, Jinna Smit, Marius Bulacu, and Lambert Schomaker. Writer identification using directional ink-trace width measurements. *Pattern Recognition*, (45):162–171, 2012. doi: 10.1016/j.patcog.2011.07.005.
- A.A. Brink, R.M.J. Niels, R.A. van Batenburg, C.E. van den Heuvel, and L.R.B. Schomaker. Towards robust writer verification by correcting unnatural slant. *Pattern Recognition Lett.*, 32:449–457, 2011.
- Marius Bulacu, Axel Brink, Tijn van der Zant, and Lambert Schomaker. Recognition of handwritten numerical fields in a large single-writer historical collection. In *Proc. of ICDAR*, 2009.
- Mark Aussems and Axel Brink. Digital palaeography. In Malte Rehbein, Patrick Sahle, and Torsten Schassan, editors, *Kodikologie und Paläographie im Digitalen Zeitalter / Codicology and Palaeography in the Digital Age*, volume 2 of *Schriftenreihe des Instituts für Dokumentologie und Editorik*, pages 293–308. Books on Demand, Norderstedt, 2009.
- Henk Bekker, Axel A. Brink, and Jos B. T. M. Roerdink. Reducing the time complexity and identifying ill-posed problem instances of minkowski sum based similarity calculations. *International Journal of Computational Geometry and Applications (IJCGA)*, 19:441–456, October 2009.
- Axel Brink, Marius Bulacu, and Lambert Schomaker. How much handwritten text is needed for text-independent writer verification and identification. In *Proc. of the 19th International Conference on Pattern Recognition (ICPR)*, 2008.
- Axel Brink, Harro van der Klauw, and Lambert Schomaker. Automatic removal of crossed-out handwritten text and the effect on writer verification and identification. In B.A. Yanikoglu and K. Berkner, editors, *Proceedings of Document Recognition and Retrieval XV, IS&T/SPIE International Symposium on Electronic Imaging*, 2008.

- Marius Bulacu, Lambert Schomaker, and Axel Brink. Text-independent writer identification and verification on offline arabic handwriting. In *Proc of ICDAR 2007*, pages 769–773, 2007.
- Axel Brink, Lambert Schomaker, and Marius Bulacu. Towards explainable writer verification and identification using vantage writers. In *ICDAR*, pages 824–828, 2007.
- T. van der Zant, L.R.B. Schomaker, M. Wiering, and A.A. Brink. Cognitive developmental pattern recognition: Learning to learn. In *Proceedings of the 2006 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1208–1213, 2006.

Curriculum vitae



Axel Brink was born in 1979 in Dalfsen, the Netherlands. He received the M.Sc degree in Computing Science on the topic of Scientific computing and Visualization from the University of Groningen in 2004. He completed one year of Communication and Information Sciences at the same university in 2005. Between 2005 and 2011, he has been a Ph.D. student at the Institute of Artificial Intelligence and Cognitive Engineering (ALICE), University of Groningen. He taught a course on handwriting recognition and supervised several student projects. Since 2010, he has been employed at Be Informed, a software company in Apeldoorn.

- χ^2 distance, 18
- angle measurements, 85
- automatic, 10
- between-writer variability, 6
- binarization, 9
- biometrics
 - behavioral, 2
 - handwriting, 1
- brittle, 27
- Brush feature, 14
- chain code, 12
- City-block distance, 16
- consistency, 7
- contents, 7
- Contour tracing, 85
- crossed-out text, 37
- curve style, 7, 13
- density estimation, 18
- Directions feature, 13
- Dutch charter dataset, 76
- Edge-direction feature, 13
- equal-error rate (EER), 19
- Euclidean distance, 16
- false accept, 19
- false reject, 19
- feature extraction, 10
- feature vector, 10
- Firemaker dataset, 21
- forensic document examiners, 1
- Fraglets feature, 11
- garbage in, garbage out, 27
- GIWIS, 117
- grayscale conversion, 9
- handwriting biometrics, 1
- handwriting recognition, 114
- Hinge feature, 13
- hit list, 2, 21
- IAM dataset, 22
- individuality, 6
- letter form, 7
- line wiping method, 31
- Manhattan distance, 16
- minimum amount of text, 34

- model
 - between-writer distances, 18
 - within-writer distances, 18
- motor program, 6
- nearest-neighbor, 21
- NFI dataset, 102
- off-line handwriting, 5
- Oslo lineup, 114
- Otsu's thresholding method, 9
- paleographers, 1
- performance influencing factors, 27
- preprocessing, 9
- pressure, 7
- probability distribution, 11
- proportions, 7
- quantitative paleography, 75
- query document, 21
- questioned document, 1
- Quill feature, 84
- Quill probability distribution (QPD), 88
- region of interest (ROI), 10
- robust system, 27
- robustness
 - intrinsic, 27
 - practical, 28
- segmentation, 10
 - character and word-level, 10
 - line and text-block level, 10
- slant, 7
- spacing, 7
- Srihari's dataset, 22
- statistical features, 11
- statistical pattern recognition, 5
- stylometry, 15
- text-dependent comparison, 10
- text-independent comparison, 10
- threshold, 18
- trace width, 75
 - analysis of production, 79
 - in modern handwriting, 84
 - measurements, 86
 - models, 82
- TriGraph, 4
- TriGraph Slant Dataset, 53
- Type-I error, 19
- Type-II error, 19
- unit of interest, 10
- vantage profile, 101
- vantage writers, 101
- White runs feature, 15
- wiping text, 31
- within-writer variability, 6
- writer identification, 2, 21
 - classifier, 21
- writer verification, 2, 18
 - classifier, 18
 - training, 18
- zone of tolerance, 28