IJDAR (2001) 3: 218–231

International Journal on Document Analysis and Recognition


# A computer-based system to support forensic studies on handwritten documents

**Katrin Franke, Mario Köppen**

Fraunhofer Institute for Production Systems and Design Technology, Department of Pattern Recognition, Pascalstrasse 8–9, 10587 Berlin, Germany; E-mail: {katrin.franke,mario.koeppen}@ipk.fhg.de

Received July 12, 2000 / Revised October 13, 2000

**Abstract.** Computer-based forensic handwriting analysis requires sophisticated methods for the pre-processing of digitized paper documents, in order to provide high-quality digitized handwriting, which represents the original handwritten product as accurately as possible. Due to the requirement of processing a huge amount of different document types, neither a standardized queue of processing stages, fixed parameter sets nor fixed image operations are qualified for such pre-processing methods. Thus, we present an open layered framework that covers adaptation abilities at the parameter, operator, and algorithm levels. Moreover, an embedded module, which uses genetic programming, might generate specific filters for background removal on-the-fly. The framework is understood as an assistance system for forensic handwriting experts and has been in use by the Bundeskriminalamt, the federal police bureau in Germany, for two years. In the following, the layered framework will be presented, fundamental document-independent filters for textured, homogeneous background removal and for foreground removal will be described, as well as aspects of the implementation. Results of the framework-application will also be given.

**Key words:** Forensic handwriting analysis – Document pre-processing – Background filter design – Genetic programming – Mathematical morphology

## 1 Introduction

Handwriting examination or identification is frequently used in criminal investigation, by the prosecution, and in the sentencing of criminal offenders. Important fields of interest are, for example, financial fraud, extortion, threat, terrorism, drugs-related crime, pornography, and racism. The aims, methods, and techniques of forensic handwriting examination differ fundamentally from those of graphology. A forensic handwriting expert compares handwriting on the basis of well-defined sets of characteristics and does not make any relation between handwriting characteristics and personality traits.

*Correspondence to*: K. Franke

a) b)

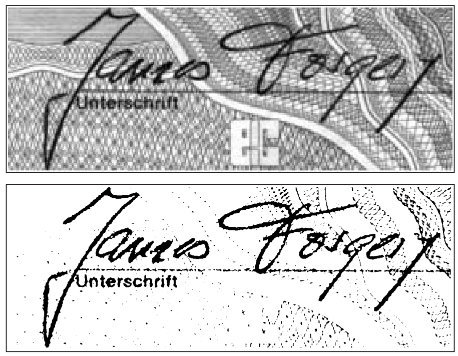


**Fig. 1a,b.** Poor separation of handwriting from textured background: **a** original signature image; **b** signature image after processing

Forensic handwriting examination has become an established part of forensic science. As in other related fields of forensic science, such as fingerprint identification and examination of toolmarks and bullets, handwriting examination is primarily based upon the knowledge and experience of the forensic expert. Although much effort has been put into the training of the examiners, the methodology of forensic handwriting analysis is being criticized, in particular, the lack of proof of validation. Due to non-objective measurements and non-reproducible decisions, traditional methods, such as visual inspection and expert rating, have been supported, to a certain extent, by computerized semi-automatic and interactive systems. Two systems operating in forensic labs for such a purpose are the Fish-system [21] and the Script-system [4]. Although there is some promising research in this area [7,8,24], it is rarely applied in forensic labs.

Computer-supported analysis of handwriting, using digital image processing and pattern recognition methods, requires sophisticated pre-processing methods to extract the handwriting specimens from documents, in par-

a)
b)


**Fig. 2a,b.** Poor threshold binarization: **a** original signature image; **b** signature image after processing

ticular, to remove the document background as well as other imprints which are not the subject of the handwriting examination. Unfortunately, the systems that were used in forensic labs [21,4] could not fulfill this demand properly (Fig. 1 and Fig. 2). By request of the handwriting examination group of the Bundeskriminalamt (the German police bureau) an open layered framework was developed to process handwritten documents.

The key idea of the model is to provide distinguished levels of user interaction during the configuration of the system for document processing, in particular, from a simple parameter adjustment up to an initialization for an automated filter generation. In contrast to known approaches that focus on parallel processing [1], the layers of the proposed model represent abstraction levels of potential user interaction. Of course, the overall processing of a certain document is carried out automatically.

In the following section the layered model of the proposed framework is motivated and explained as a whole. Then, Sect. 3 focuses on single methods of handwriting pre-processing that are applied within the framework. Such methods that are not published yet will be presented in detail. Section 4 provides some facts about the implementation. Results of the framework's application are stated in Sect. 5, followed by conclusions and further studies in Sect. 6.

## 2 System overview

Within this section the developed framework will be presented from various perspectives. At first a brief description of the major elements of the user interface will be given. In this way the reader becomes familiar with the available tools. The second point of view comprises the on- and offline phase of the framework application, whereby the third view of the framework focuses on its abstract architecture to explain user interaction abilities.

Before this, the framework approach will be presented.

### *2.1 State at the project beginning and requirements*

At the beginning of our project in 1997, there was no suitable solution available. The systems [21,4] employed

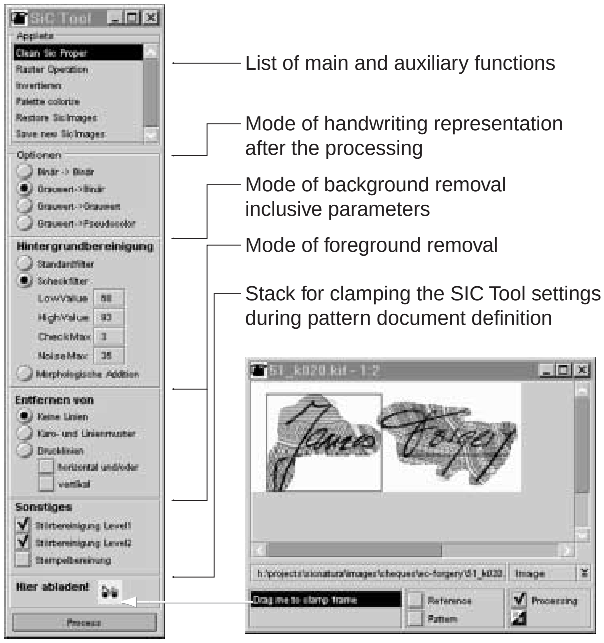


**Fig. 3.** A tool for specifying processing operators and their parameters

at this time were not able to provide high quality images of digitized handwriting. Major drawbacks were caused by only partially eliminated noise signals, cases where a correct separation of handwriting from textured background were impossible (Fig. 1), and cases where parts of the handwriting were lost (Fig. 2), in particular, those with low contrast to the background. These problems revealed the need for specific methods dealing with handwritten documents and for taking into account the typicality of handwriting. Such methods should be aimed at focusing on a true representation of the original handwriting, as this is required for forensic examinations. (Technical factors, however, limit the scope of true handwriting representation. Handwritten documents are scanned with 200–600 dpi, depending on the examination goals as well as performance goals of the computer-based processing). Probably, the main reason for the mentioned problems is that, at the time, when the employed systems [21,4] were developed, the technical premises were not yet available. (In the case of the Fish-system [21] the development started in 1975). Today, however, this is no longer the case. The time has come to consider the further development of traditional systems – reflecting on new technical possibilities.

Hence, the Bundeskriminalamt (the German central police bureau) initiated a research and development project. The aim was to design and to realize a new framework for the elimination of noise signals from digitized handwritten documents that are the subjects of forensic examination. Moreover, the new framework should be able to be integrated into an existing system environment such as the Fisch-system [21] and it should be able to use the framework as a stand-alone application as well. The practical relevance of the project might be pointed out with some facts. The Fish-system [21] was conceptu-

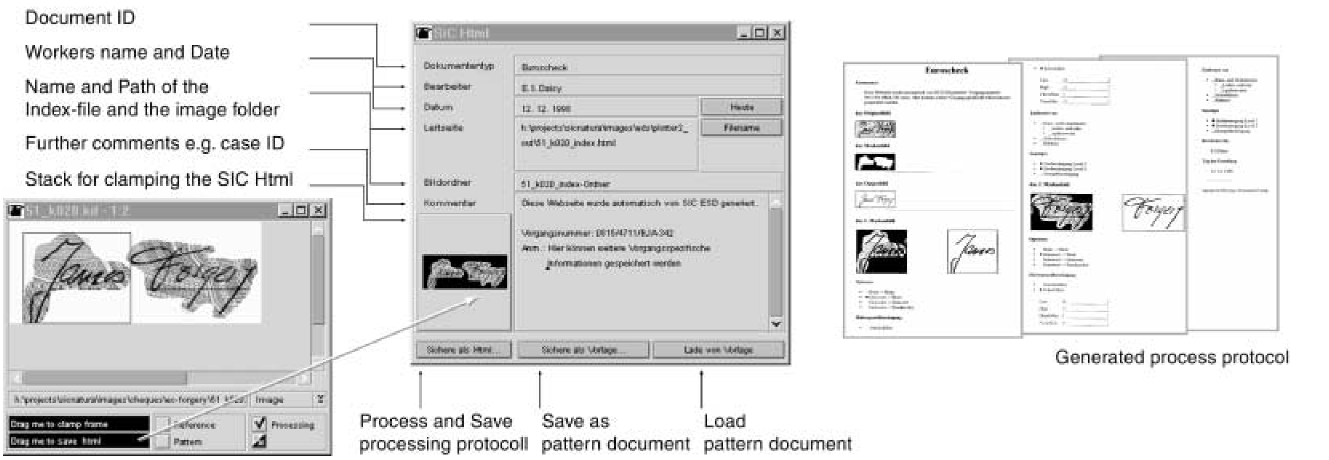


**Fig. 4.** Tool for using Html to store process protocols

alized to handle 10,000 investigation cases per year. This means in the worst case there would be 10,000 different types of document. There are no restrictions to the document types. The only thing that they have in common is handwriting on them. Seventy-seven thousand documents were stored until 1995 in the Fisch-system [21] and this number increases every day. The scientific challenge is caused by this huge number of different documents, which is assumed as infinite. Moreover, at the same time, high-quality images of handwriting must be provided even if there are low-contrast strokes and/or textured background.

### *2.2 System architecture*

The description of the user interface is limited to such elements that enable the adaptation of the operators and parameters, and that support the archiving of control sequences in so-called *pattern documents* as well as of processing protocols to match the required proof of processing and validation.

The *SIC Tool* is the central element of the user interface. It supports the operator and the numerical parameter specification for document processing. From the chosen operators the resulting processing algorithm is derived automatically. Within the offline phase of the framework application (compare Sect. 2.3.1) the *SIC Tool* is applied for the manual definition of pattern documents as well as for the interactive processing of questioned documents without a specific pattern document. In the case of the application of additional structural parameters, like a mask image that defines various regions of interest within the questioned document, the *SIC Tool*-settings were applied sequentially and by simple drag-and-drop clamped to each region. In such a case, the management of this pile, bunch of various settings is carried out by the designed document class; for the *SIC Tool* it is not relevant. As can be seen in Fig. 3, the *SIC Tool* enables the selection of various operators for back- and foreground removal. Among these are operators for document-independent as well as document-dependent processing (compare Sect. 3). The document-specific pro-

**Fig. 5.** Specific methods, taking into account the typicality of handwriting, realized as *Plug-In* for Adobe Photoshop

cessing (also Sect. 3) is supported by the aforementioned mask images as well as by the extension of the function-list at the top of the *SIC Tool* (see Sect. 3.3).

Either the created pattern document or the processing protocol of interactive processing can be stored by employing the *SIC Html*-tool (Fig. 4). The *SIC Html*-tool uses **h**yper**t**ext **m**arkup **l**anguage (html) to relate ASCII-data that specify the operators and the numerical parameter to images, like the questioned document image or the mask image. Moreover, the html-standard was chosen to display and to exchange protocol files and pattern documents very easily by a normal web-browser. A proprietary editor is really not state-of-the-art. In addition, the html-files might be provided via Intranet within a lab, e.g., for validation purposes or for expert ratings. The *SIC Html*-tool also supports the inclusion of administration data, such as the case number, the name of the expert, and the date of the file creation. This information is important to meet the requirements of quality management in forensic handwriting examination.

For fundamental digital image processing, such as the definition of the mask images, common tools should be used, too. Within the forensic labs Adobe Photoshop is widely in use. Thus, some routines of inter-application communication support the data exchange between Photoshop and the presented framework. Moreover, a *Plug-In* was developed to extend Photoshop by sophisticated methods in handwriting pre-processing (Fig. 5). Hence, the forensic experts might tickle around with the provided document-independent operators, e.g., for parameter adaptations, or the *Plug-In* might be employed in small-sized forensic labs that are not able to install the whole system.

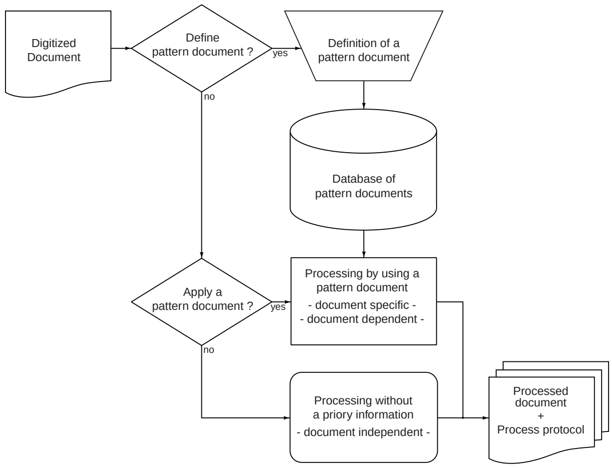


**Fig. 6.** Flow chart of document processing

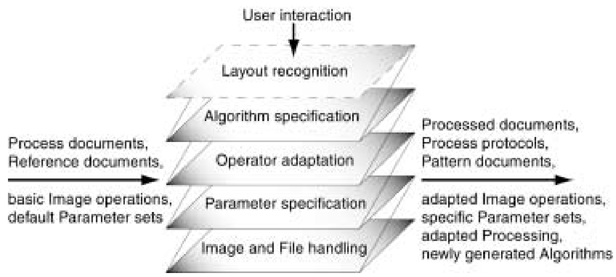


**Fig. 7.** Layer model of the framework for document processing

## *2.3 Framework for document processing*

From the authors' point of view, the realization of fully document-independent handwriting processing will stay a scientific challenge for a while. Although there is a universal processing flow, which comprises document-layout recognition and document segmentation, and handwriting segmentation and handwriting quality enhancement, a standardized queue of processing stages cannot be used as a framework concept. With respect to the expected amount of different document types, being the subject of forensic analysis, and all the inhomogeneous structures and characteristics of the documents, classic serial processing has to be replaced by an iterative and parallel one. Hence, the processing operations have to be selected by considering the prevailing document and by considering the specific processing-target. Moreover, an iterative and parallel processing concept allows taking into account intermediate data to improve final processing operations. It is understood that fixed parameter sets for the processing are not useful, either. In addition, even the number of different image operations should be extendable, because there are many approaches promising good processing results [6,17,18,3,28,27].

The best solution seems to be an open layered framework concept to create an assistance system for forensic handwriting analysis. It has to comprise a functional kernel providing basic processing operations for the most frequent document types, as well as opportunities to adapt/extend the functionality by user interactions. The open layered architecture of the framework that was developed by the authors can be seen in Fig. 7. Each layer stands for an abstraction level of procedures working on the documents. Furthermore, the layers support the understanding of how the framework operates and how user interaction might adapt/extend the functionality to specific demands.

*2.3.1 The application of the framework.* The framework can be applied in two separate cycles (see Fig. 6); once a pattern document is prepared, it can be used for automatic control later on.

The so-called online phase works as follows: from a collection of pre-defined pattern documents, a human user selects the suitable one, which includes all that information needed to process the questioned document(s) correctly. The information stored in the pattern document includes the structural and numerical parameters, the involved image operators, as well as the processing flow. Thus, the online phase operates autonomously. Documents with their corresponding pattern documents might be processed on a remote machine or as batch job overnight.

In the offline phase the aforementioned pattern documents might be generated and small numbers of documents might be processed interactively, too. In every case a protocol will be provided in the same style as a usual pattern document. Moreover, in the offline phase the user can also generate new image operators, which are adapted to special demands that were not predictable during the framework development. A detailed description of all the layers supporting user interaction in the offline phase follows.

*2.3.2 The layered model architecture.* The presented framework, whose block diagram is shown in Fig. 7, is made of the *IFH* (image and file handling), the *PS* (parameter specification), the *OA* (operator adaptation), and the *AS* (algorithm specification) layers. With respect to the huge amount of different documents, the *LR* (layout recognition) layer is not working autonomously. Here, the user's assistance is still required.

*The IFH layer for image and file handling:* Dealing with digitized documents requires managing them. Users want to scan, load, view, browse, save, and process document images. A huge database is not part of the presented framework. Depending on the application it is readily available, as in the case of the Fish-system, or it might be extended. The aforementioned pattern documents as well as the protocol files were stored as **h**yper**t**ext **m**arkup **l**anguage-(html)-files. Of course, these html-files are generated automatically. The advantage of using them is that a fixed relation of document images to all other parameters and the files can be distributed and displayed easily.

*The PS layer for parameter specification:* Assigning different image operators to various regions of interest within a document is a common method. Within the PS-layer these structural parameters (see Sect. 3.3.1), also called mask images, can be defined interactively. Then, the specific parameter sets, the operators, and the algorithms have to be assigned to each region. There are also some adjustable numerical parameters, e.g., the maximal noise ratio.

*The OA layer for operation adaptation:* To extend the functional kernel, a module for the automated generation of image processing operators is embedded within the framework. This Lucifer-module [15, 10] (see Fig. 18 and Sect. 3.3) uses genetic programming to design texture filters, which can be integrated in the document processing for background filtering. The user has only to provide a so-called goal image, as well as a small part of the original image, from which the handwriting cannot be extracted using the other background filters. The goal image is hand-made by tracing the foreground information, in particular, the handwriting. In this way, the goal image can verify the filters generated by the Lucifer-module. A detailed description is given in Sect. 3.3.

*The AS layer for algorithm specification:* The open architecture of the framework allows the use of various approaches to document processing. Unfortunately, there is no really intelligent layout recognition yet, which would be able to process this huge number of different documents. Therefore, a human user has to do this job once, in advance. The image operators, implemented in the framework, are chosen in such a way that users are not overwhelmed, hopefully. They can switch on/off single operations like:

- Homogeneous background + grayvalue output image
- Textured background + lines + binary output image
- Generated filter $XYZ$ + machine-print + noise + lines + binary output image
- Empty reference dropout (reference $ABC$) + noise + grayvalue output image.

Even if the framework is extended, this procedure will hold. The open layered architecture of the proposed framework is able to consider upcoming operators, too. Within each layer of the framework, new approaches might be included and the users of the framework might interact with them. Algorithms of document processing, which are included within the framework up to now, are presented in Sect. 3.

## 3 Methods for document pre-processing

The operators for document processing can be grouped into algorithms for: document-layout analysis, such as the automatic location of text blocks; algorithms for background texture and/or image cleaning, such as histogram and threshold based techniques; algorithms for foreground cleaning, such as guideline and preprinted data removal; and algorithms for handwriting reconstruction and enhancement. There are different strategies for cleaning of background and foreground. Some approaches use separated processing [5, 18, 6, 2] while others use combined cleaning operators for fore- and background [20, 17].

Although, this grouping is fundamental, the authors want to focus on a further aspect. As mentioned before, it is necessary to handle various distinguished document types, therefore a distinguished amount of a priori knowledge about the characteristics of a document type is expected. Hence, the operators for document processing must be able to handle this different a priori knowledge. Neglecting given information or not adapting parameters may lead to a moderate document quality. On the other hand, to expect too much information could mean that some documents might not be processed. The authors propose a grouping of the processing operators that is derived from the given a priori knowledge of a document type.

In general, the document types being examined in the daily forensic casework can be sorted into three groups. Types of documents being examined very often represent group one. For this group it should be easy to get an empty reference document, e.g., a driver's license, a pass document or a standardized bank check. The second document group includes documents, which occur more or less often, such as a registration form of a certain hotel or an insurance contract. In some cases it might be difficult to obtain an empty reference document. However, it would be nice to create a pattern document, because the document type could be examined repeatedly. The third document group covers all those document types appearing only few times and/or when there is no way to get an empty reference, e.g., an aged testament or a piece of paper that has been washed. From these three document groups the required operations were derived.

The processing operators and the derived generic types of document processing differ in their consideration of a priori knowledge (e.g., empty reference, layout information and/or structural and numeric parameters). In the following, the types of document processing are listed in an increasing order by the strength of their connectivity to a reference:

- Document-independent (without a priori knowledge)
- Document-specific (by using adapted parameters)
- Document-dependent (by using an empty reference)

The required processing quality of the handwritten documents and the flexibility of the whole framework are opposing targets (see Fig. 8). Therefore, the operators to be included in the framework have to be selected carefully. In addition, there could be diverse operators available, which may be chosen alternatively. The processing operators chosen by the authors cover local and global image filters, textural- and structural- as well as syntactical- and layout-driven approaches. More special-

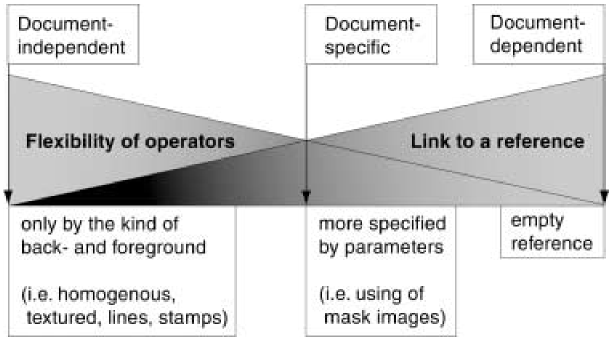


**Fig. 8.** Types of document processing

ized document-independent operators were provided for background removal of homogeneous and textured backgrounds, and for foreground removal of lines, machine-prints and noise. The document-specific processing is realized by using special parameter sets as an additional input for aforementioned document-independent operators. Moreover, an embedded module (see Fig. 18) allows the automated generation of document-specific background filters. The document-dependent processing is realized by using the extended approach of morphological subtraction [20, 10].

### *3.1 Document-dependent processing*

Usually, for the removal of nearly homogeneous backgrounds, different kinds of histograms and threshold techniques are used. Against this, Okada and Shridhar [20] pointed out that enhanced inter-image subtraction, the so-called morphological subtraction, provides acceptable cleaning results for textured and/or image check backgrounds. For document-dependent processing, the authors provide a method for background and foreground cleaning [10] that was derived from the morphological subtraction. In contrast to the fundamental approach, a blank reference-document is filtered only once in the offline phase and is used for grayvalue comparison during the background cleaning process [10]. For the comparison, a segmented 2D-Histogram is used for looking-up. Hence, it is possible to provide qualitatively comparable processing results by using a smaller amount of computational time. Concomitant to the introduction of 2D-histogram segmentation, which will be referred to as 2D-hookup (for further details see the appendix), the basic question is about the origins of the two images which are used for the look-up. So far, they have been the treated, empty reference-document and the, usually untreated, filled-in document usually untreated (Fig. 9).

### *3.2 Document-independent processing*

*3.2.1 Textured background removal.* The processing of textured documents was studied intensively in the past [11, 10, 9]. Textured background removal on grayvalue images is a challenge. Handwriting might not be separated by global grayvalue intensities or empty reference subtractions [10]. Only the consideration of local structures and local intensities by applying adapted morphological operators provided acceptable results (compare Fig. 10). The first implementation of this filter was done empirically by the authors and was later proven by the Lucifer-module [10] (Sect. 3.3). It was also shown [10] that the 2D-lookup can be derived as an abstraction of the former inter-image subtraction, which gives a more flexible approach to inter-image filtering. This is caused by the fact that for the graylevel lookup a blank check image is not mandatory. It is also possible to use two different filtered images of one filled-in check image.

In this manner, the difficult procedure for positional adjustment between filled-in and blank document image can be avoided. The fundamental problem for a practical and efficient usage of the 2D-lookup is the selection of filter operations, which provide significant differences between background and user-entered information. In an intermediate step, the 2D-histogram has to be segmented. The result of this histogram analysis depends on both the free-to-choose image operators and the quality of the labeling in the histogram. The application of the 2D-lookup, as it is employed for textured background removal of various types of textures, such as Eurochecks and American Express Traveler Checks, is shown in Fig. 10. It has to be considered that in this application-case of the 2D-lookup only such image elements can be eliminated which are included in the labeled segment of the 2D-histogram. Further image elements such as the guidelines that show comparable characteristics to the handwriting (e.g., the same line thickness), and that is therefore not influenced by the filtering, has to be eliminated by an additional foreground cleaning.

For any further requests, such as background removal of a new passport, applying the Lucifer-module (compare Sect. 3.3) to determine a suitable filter operation seems to be more convenient.

*3.2.2 Homogeneous background removal.* In the following, methods for the elimination of homogeneous background will be listed in detail. Independent of the writing material (e.g., kind of ink, kind of paper, and paper color) and writing pad, these methods offer a binary image of high quality, without neglecting relevant parts of the writing product. It is expected that the background has no textures and that there is a small contrast difference between the handwriting and background. There are two major steps included in the procedure, which were optimized for the special case of handwriting.

*Local contrast enhancement*: Local image operators, along with other operators of mathematical morphology, are used for the purpose of contrast enhancement. In this context "local" means that the parameter assessment of the image operators is performed in the neighborhood of every pixel. This is done by a mask-like structuring element, which is positioned on every image position (slides over the image). The parameter choice of every position depends on a computation from the other image points underneath the mask. This procedure allows a better

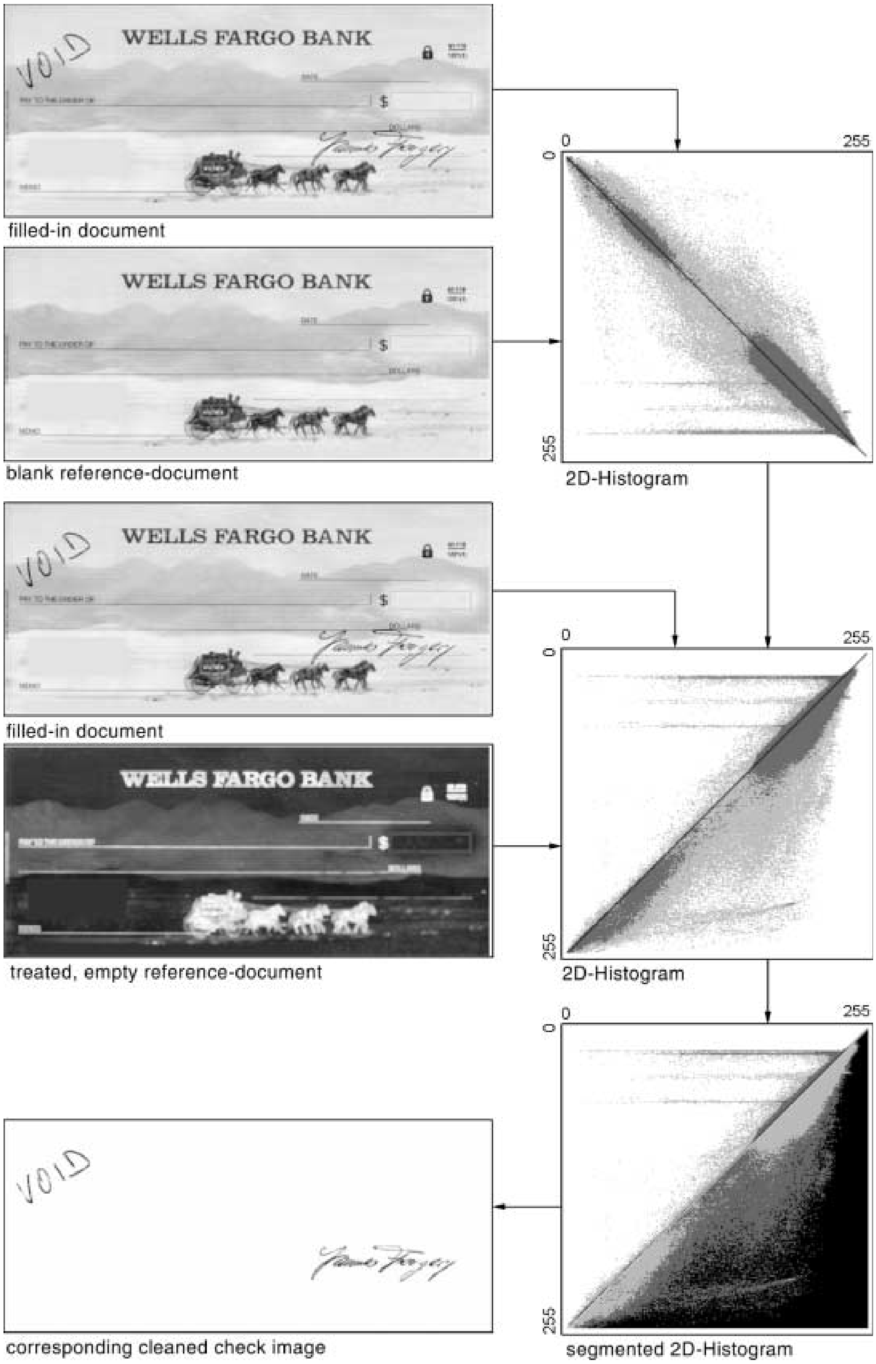


**Fig. 9.** Application of the 2D-lookup for document-dependent handwriting segmentation

representation of local disturbances of the intensity distribution. Moreover, this method reduces the possibility of error resulting from global image properties such as a dark background causing the loss of low-contrast strokes.

The size of the structuring element should not exceed the average stroke width of the writing trace. Previous research [11] established that the stroke width of handwriting in a document for a broad range of varying writing devices such as ball-point pen, fountain pen, or roller pen, digitized with a resolution of 300 dpi, was about 7–10 pixels. Due to this fact, for the contrast enhancement 5 pixels are chosen for the structuring element.

Figure 11 shows the result of the application of contrast enhancement and grayvalue normalization for handwriting. This time, even fine, small details are preserved in the resulting image. Hence, by applying local-adaptive contrast enhancement, the inclusion of fine details is ensured. A detailed description of the operator is stated elsewhere [9].

*Local adaptive binarization*: The following binarization step should not result in broad lines or unexpected alterations to the fine lines in the original image. For this reason, a direction-oriented binarization, which considers image contents as well as grayvalue contrast, is advised. For the special case of handwriting, the slant and

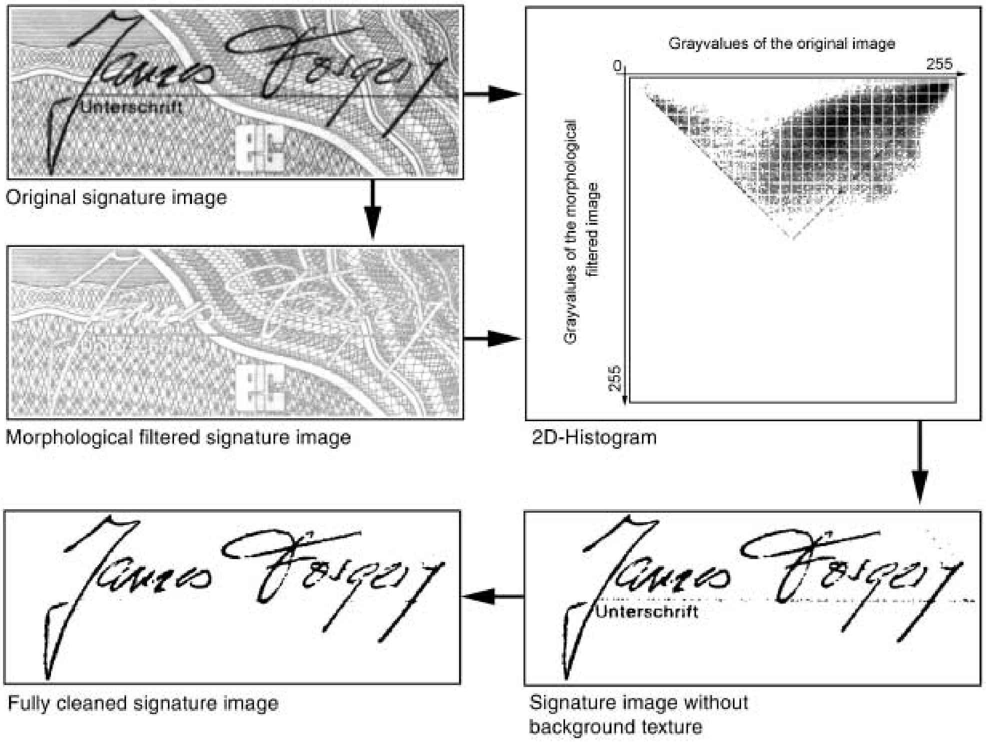


**Fig. 10.** Application of the 2D-lookup for document-independent textured background cleaning


**Fig. 11a,b.** Local contrast enhancement: **a** original signature image; **b** signature image after processing

the width of every stroke are also considered. This is performed by local image operators with the size of the structural element of 7 pixels.

$$F_b(x,y) = \begin{cases} 1 & \text{if } \bigvee_{\mathrm{i}=0}^{3} [\,\mathrm{L(P_i)} \wedge \mathrm{L(P'_i)} \wedge \mathrm{L(P_{i+1})} \wedge \mathrm{L(P'_{i+1})}\,] \\ & \text{is true} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

with

| | |
|---|---|
| $w$ | pre-defined maximal stroke width |
| $P'_i$ | $P'_{(i+4)\mathrm{mod}_8}$, for $i = 0, \ldots, 7$ |
| $L(P)$ | $= \mathrm{ave}(P) - g(x,y) > T$ |
| $T$ | pre-defined threshold |
| $P_x, P_y$ | coordinates of point $P$ |
| $g(x,y)$ | grayvalue at point $P(P_x, P_y)$ |

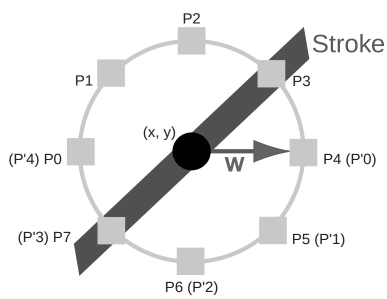


**Fig. 12.** Functional principle of the locally adaptive binarization [14]

This fundamental binarization process (compare Fig. 12 and 1) was introduced by Kamel and Zhao in 1993 and further extended by replacing an average filtering through a Gaussian filtering [19].

*3.2.3 Reconstruction and noise removal.* A binarized image might contain distortions resulting from the poor quality of the document or the scanning process. The evaluation of a so-called seed image followed by a reconstruction of the binary image solves this problem. Here, a morphological reconstruction operation is employed [23]. The seed image is obtained by clearing all obscure image contents, such as strokes with a width below a given threshold. Figure 13 shows the reference image of a handwritten signature. In the reconstruction step, all image segments, which do not contain at least 1 pixel of the reference image, are cleared. The result is a binary image

**Fig. 13a,b.** Reconstruction and noise removal: **a** seed image derived from the original image in Fig. 11; **b** final, high-quality binary image of the signature

**Fig. 14a,b.** Removal of stamps and other imprints: **a** original signature image with overlapping stamp; **b** signature image after stamp removal

of high quality (Fig. 13.) After these image operations, other algorithms are used to remove imprints like lines and machine printing. A further, smoother noise filtering is realized by the well-known analysis of connected components. The corresponding parameter controls the size of image elements to be removed.

*3.2.4 Stamp and imprint removal.* The removal of stamps and other machine-imprints employs an extended connected component analysis. Besides the segment-sizes, the frequency of occurrence in a row of uniform segments is also considered. This approach works quite well (Fig. 14). However, problems arise if the stamp-segments are connected to the handwriting. Here, further research and development is required.

*3.2.5 Line removal and stroke reconstruction.* There are two kinds of operators for horizontal and/or vertical line removal. The first one is applied on grayvalue images of handwritten documents. This approach is inspired by the legendary Otsu-method. Here, the operator acts like a background filter, by simple subtraction of the average pixelvalue per image row/column from the pixelvalue from the actual image position. The filter for homogeneous background removal is applied to the rest of the work. This approach works quite well for lined and

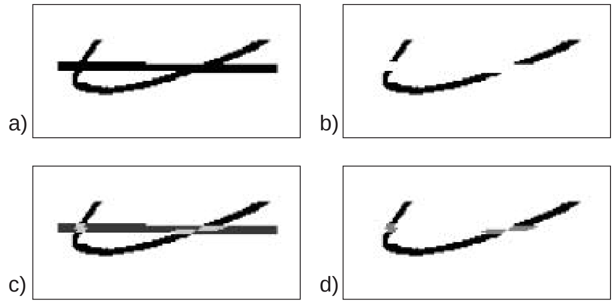

**Fig. 15a–d.** Line reconstruction by using morphological filtering (length of the structuring element = 5 pixels; angle = $\pi/4$): **a** binarized image; **b** stroke image after line removal; **c** reconstructed stroke; **d** result image

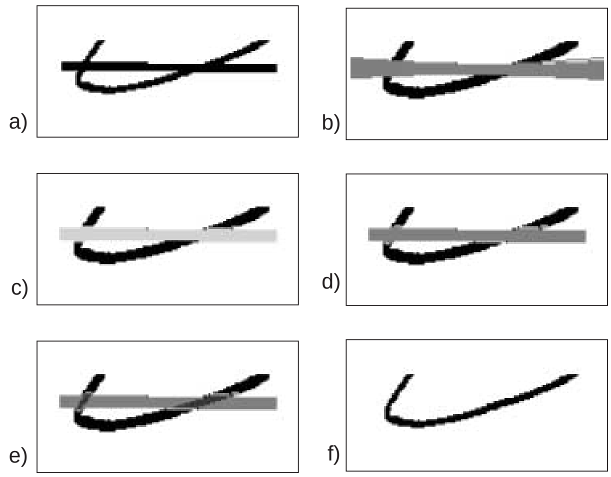

**Fig. 16a–f.** Proposed line reconstruction: **a** binarized image; **b** searching for line candidates; **c** detected line; **d** line removal with concomitant stroke marking; **e** reconstructed line; **f** result image

squared paper, but the results for guidelines are unsatisfactory.

The second, more sophisticated algorithm is based on a structural analysis of the elements of the binary image. Most of the effort was spent in the reconstruction of the handwritten strokes during line-removal.

In contrast to the often-stated assumption that the handwritten strokes may be reconstructed by applying morphological filtering, Fig. 15 shows that in the case of a small cutting angle and thick guide-lines the reconstruction fails. No matter whether binary or grayvalue morphology is used, if the cutting angle is smaller than the angle of structuring element (also called dynamic kernels [27]), the upper and lower stroke cannot be connected. Furthermore, this procedure leads to distortion in the case of connected strokes.

The proposed method for stroke reconstruction combines handwritten stroke tracing and line removal in one step (Fig. 16). After detecting a line within the image, we try to remove this carefully by checking the local neighborhood of the line for crossing strokes. All upper and lower crossing points were marked (Fig. 17). Then, the marks were grouped and directly connected. The follow-

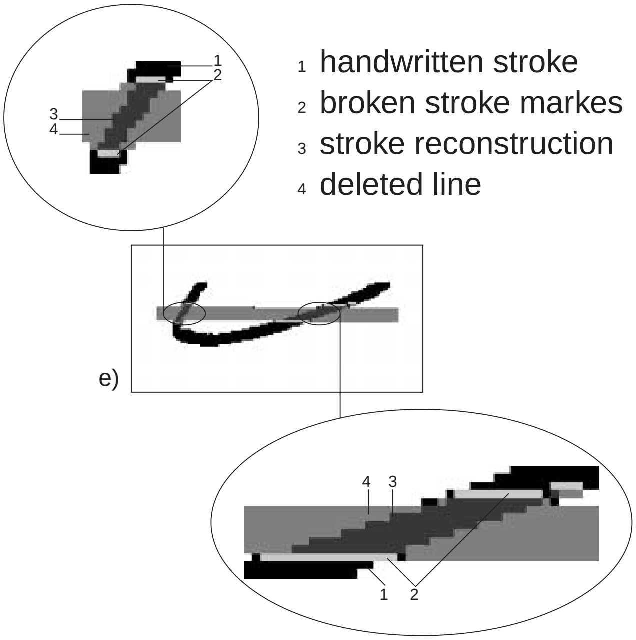


**Fig. 17.** Stroke marking: detail of Fig. 16

ing look-up with the original image excludes stroke distortions. The results turn out well; however, in the case of handwritten strokes that touch the line, the grouping works only moderately.

### *3.3 Document-specific processing*

The document-specific pre-processing is realized into two ways. The first one uses the available operators. Then, depending on the user's selection, specific operators with their corresponding parameters were clamped to the predefined regions of interest. The second approach supports the generation of new, adapted background filters that takes the characteristics of a certain document type into account.

*3.3.1 Application of mask images.* To realize efficient document processing, the application of so-called mask images is quite common. Such a mask image is employed to define regions of interest, such as a number-field or alpha-numeric-field, e.g., to support optical character recognition tools. In addition, this approach is useful for the separation of handwriting. A document may comprise various regions that have quite different background textures, as in the case of a passport. Then, a specific mask image helps to apply the correct filter operation to each region and to provide sophisticated cleaning results at the end.

*3.3.2 The Lucifer II framework.* For solving the problem of determining a suitable filter operation that distinguishes between the document background and user-entered information, the LUCIFER II framework for filter design is used. LUCIFER II was developed for textural filter design in the context of surface inspection, and it uses evolutionary algorithms for its adaptation [15]. The feasibility of using the LUCIFER II framework for the design of specific background filters will be presented in the following.

*General overview*: The framework is composed of (user-supplied) original image, filter generator, filter output images 1 & 2, result image, (user-supplied) goal image, 2D-lookup matrix, comparing unit, and filter design signal.

An evolutionary algorithm maintains a population of individuals, each of which specifies a setting of the framework. By applying the resulting 2D-lookup and measuring the quality of coincidence of goal and result image, a fitness value can be assigned to each individual. These fitness measures are used for the standard genetic operations of an evolutionary algorithm. The 2D-lookup algorithm, the fitness measure, the node and terminal functions of the individual's expression trees, and the setting of the 2D-lookup matrix will be shortly described in the next sections. A more comprehensive introduction to the framework is given elsewhere [15].

*Fitness function*: A fitness measure is given by the degree of coincidence of goal image and result image. Both are binary images. The fitness measure, which is computed by the comparing unit, is a weighted sum of three single measures: the quota of white pixels in the result image, which are also white in the goal image (**whitegoalok**), the quota of black pixels in the result image, which are also black in the goal image (**blackgoalok**), and the quota of black pixels of the goal image, which are also black in the result image (**blackresok**). Note, that **blackgoalok** and **blackresok** are different. The multiple objective here is to increase these measures simultaneously. After performing some experiments with the framework, it was decided to use the following weighted sum for these three objectives:

$$
\begin{aligned}
\mathrm{f} = {} & 0.1\,\mathbf{blackresok} + 0.5\,\mathbf{whitegoalok} \\
& + 0.4\,\mathbf{blackgoalok}.
\end{aligned}
$$

This fitness function was designed in order to direct the genetic search according to the schemata theorem, by which higher weighted objectives are fulfilled first.

*Genetic design of filter operations*: An essential part of the framework are the two image processing operations. In order to generate them, genetic programming is used [16]. Genetic programming maintains a population of expression trees (in short, trees). Every tree equals a program by its structure. For the design of the trees, the following operators were used as terminal functions: move, convolution, ordered weighted averaging (OWA) [26], fuzzy integral [12,25], and texture numbers. The node functions were out of the set of the operations minus, minimum, maximum, square, evaluate. In all tests, a maximum number of 50 generations was used. For details, and also for the relaxation-based technique for the setting of the 2D-lookup matrix, consider [15].

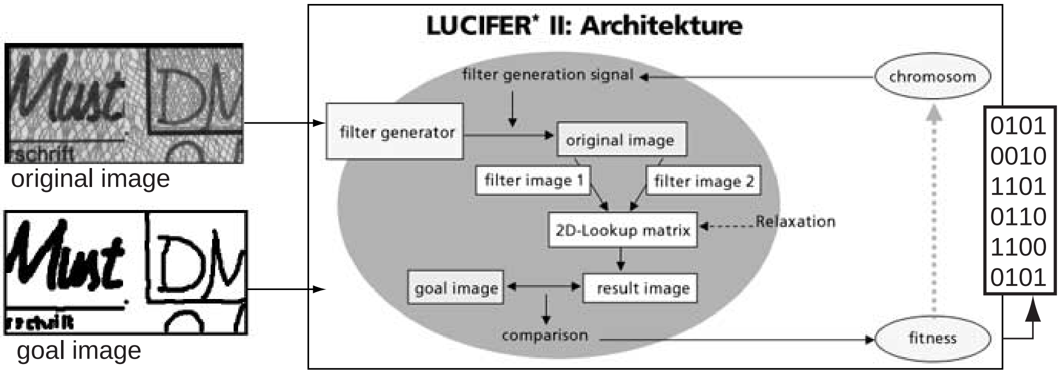


**Fig. 18.** Embedded LUCIFER-module for background filter generation

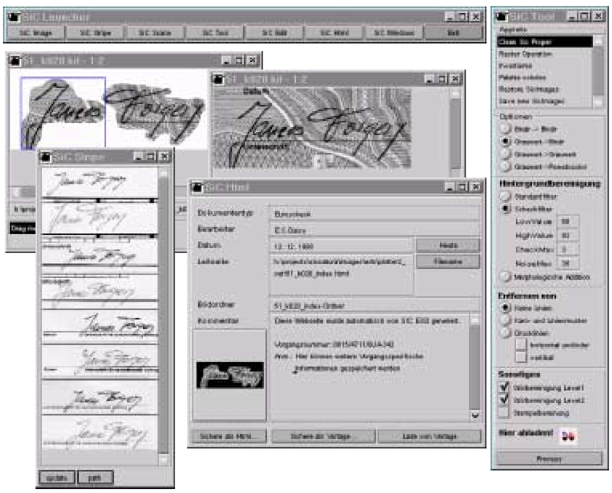

**Fig. 19.** Sneak preview of the main user interface

## 4 Implementation

The functional framework-kernel, which includes all modules for digital image processing was implemented as a library written in ANSI C. Up to now, it has run under UNIX (Solaris 2.5.1 and SUN OS 4.3), WIN NT 4.0, Windows 2000, and an OS/2 Warp 4.0 operating system. Due to a platform and operating system independent software design, it also can be used on a PowerMacintosh under MacOS 8.0. The main part of the graphical user interface (see Sect. 2.3 and Fig. 19) was implemented in Smalltalk using ParcPlace Visual Works with C-Connect running on various platforms, too. Besides the kernel function calls and the inter-application communication, the management of process and mask images and the handling of the html-protocol-files are covered by the main interface. Due to the software design with its strict separation of functional kernel and user interface, further applications were implemented, adapted to special demands in daily forensic casework. The first one allows us to process huge numbers of documents and is realized as a simple console application running on a remote machine. For small forensic labs with limited resources and small numbrers of documents to be processed an Adobe Photoshop *Plug-In* was implemented (compare Sect. 2.2). With respect to the programmer interface of Photoshop, the *Plug-In* covers only the document-independent processing functions for homogeneous and textures backgrounds as well as for line, imprint, and noise removal. Besides complete applications, the functional kernel was distributed as software library with a programmer interface and is now operating in automated check-processing systems.

**Table 1.** Processing times for background removal of various documents with 256 graylevels

| Document | Size | Time |
|---|---|---|
| homogeneous background | | |
| A4 page | 2478 × 3469 Pixel | 40.959 sec |
| bank cheque | 1771 × 1006 Pixel | 8.592 sec |
| snippet | 672 × 227 Pixel | 0.640 sec |
| Texture background | | |
| A4 page | 2478 × 3469 Pixel | 17.085 sec |
| bank cheque | 1771 × 1006 Pixel | 3.435 sec |
| snippet | 672 × 227 Pixel | 0.220 sec |

## 5 Discussion and results

When launching the research project, 212 different document types were provided by the Bundeskriminalamt. The document types were selected from various backgrounds and cover memos, diverse bank forms, passports, contracts, delivery notes, invoices, applications for a work permit, hotel registration, etc. The majority of documents were sized between DIN A6 and DIN A4.

A flatbed scanner supporting 256 grayvalues and 300 dpi resolution wa s used to digitize the handwritten documents. Grayvalue images were chosen with respect to the expected high quality images and the limited hardware resources. Therefore, up to now, the proposed framework includes only grayvalue image processing operators.

The total processing time for a document differs due to varying conditions. Average processing time for some samples processed on a Pentium II with 266 MHz and 196 MB RAM are listed in Table 1. Note that, currently, the homogeneous background removal is more sophisticated and able to keep low-contrast strokes, whereas texture background removal allows the extraction of handwriting from textured background with comparably lower quality. This is why the computational effort for homogeneous background removal is higher.

## 6 Conclusion and further work

A framework has been presented for the automated preprocessing of documents for forensic handwriting analysis. Fulfilling the stated requirements, this framework can be used in the daily casework of forensic experts and has been in use for more than 2 years now. The framework covers functions for digitalization of documents, their pre-processing, in particular the extraction and qualitative improvement of handwriting, and the archiving of processing protocols and processing parameters. The open architecture of the functional kernel supports the adaptation to further demands.

Document pre-processing itself follows a new concept that considers different kinds of bindings of a priori knowledge to processing documents. Within the concept, we distinguish between document-independent processing without considering a priori knowledge, document-specific processing with a binding of parameter sets to a document, and document-dependent processing with a strict binding of a reference to a document. To provide a wide variability of processing filters, the basic functional kernel might be extended by user-generated filters using the LUCIFER-framework [15,10] working as an embedded module.

The current restriction to graylevel image processing caused by limited hardware resources seems to be irrelevant for the future. To improve the processing quality, color image processing [3] has to be considered in framework updates and extensions.

*Acknowledgements.* The authors wish to thank Yu-Nong Zhang, Xiufen Liu, and Wensheng Ye for their contributions in coding and testing. Thanks also to Gerhard Grube and all other forensic handwriting examiners, who were involved in verifying the framework. This work was sponsored in part by the Bundeskriminalamt, section handwriting examination (GZ.:IX/B-2310297-Ge).

The authors also wish to thank the unknown reviewers for their comments that helped to improve the paper's quality.

## Appendix: The 2D-Lookup algorithm

The 2D-lookup algorithm stems from mathematical morphology [22,13]. It was primarily intended for the segmentation of color images. However, the algorithm can be generalized to be used for grayvalue images as well.

For starting off, the 2D-lookup algorithm, the two operation images 1 and 2, which are of equal size, need to be provided. The 2D-lookup algorithm goes over all common positions of the two operation images. For each position, the two pixel values at this position in operation images 1 and 2 are used as indices for looking-up the 2D-lookup matrix. The matrix element, which is found there, is used as the pixel value for this position of the result image. If the matrix is bi-valued, the resultant image is a binary image.

Let $I_1$ and $I_2$ be two grayvalue images, defined by their image functions $g_1$ and $g_2$ over their common domain $P \subseteq \mathcal{N} \times \mathcal{N}$:

$$\begin{aligned} g_1 :& \quad P \to \{0, \ldots, g_{max}\} \\ g_2 :& \quad P \to \{0, \ldots, g_{max}\}\ . \end{aligned} \tag{2}$$

The 2D-lookup matrix is also given as an image function $l$, but its domain is not the set of all image positions but the set of tuples of possible grayvalue pairs $\{0, \ldots, g_{max}\} \times \{0, \ldots, g_{max}\}$,

$$\begin{aligned} l \ :& \ \{0, \ldots, g_{max}\} \times \{0, \ldots, g_{max}\} \\ & \to S \subseteq \{0, \ldots, g_{max}\}. \end{aligned} \tag{3}$$

Then, the resultant image function is given by:

$$\begin{aligned} r \ :& \ P \to S \\ r(x, y) =& \ l(g_1(x, y), g_2(x, y)). \end{aligned} \tag{4}$$

In standard applications, every grayvalue is coded by 8 bits, resulting in a maximum grayvalue of 255. In addition, the domain of the image function is a rectangle. In this case, the 2D-lookup is performed by the following (object-oriented) pseudo-code:

```
for x=0 to img width -1 do
begin
   for y=0 to img height-1 do
   begin
      g1 = g1(x,y)
      g2 = g2(x,y)
      out(x,y) = l(g1,g2)
   end y
end x
```

To give a simple example for the 2D-lookup procedure, $g_{max} = 3$ is assumed in the following. Let

$$g_1 : \ \begin{array}{|c|c|c|} \hline 0 & 1 & 2 \\ \hline 0 & 3 & 3 \\ \hline \end{array} \quad \text{and} \quad g_2 : \ \begin{array}{|c|c|c|} \hline 2 & 3 & 1 \\ \hline 2 & 3 & 2 \\ \hline \end{array}$$

be the two input images and the 2D-lookup matrix be given by

$l$ :

| $g_2$ \ $g_1$ | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 2 | 2 |
| 2 | 1 | 2 | 3 | 3 |
| 3 | 2 | 3 | 3 | 2 |

Then, the resultant image is

$$r : \ \begin{array}{|c|c|c|} \hline l(0,2) & l(1,3) & l(2,1) \\ \hline l(0,2) & l(3,3) & l(3,2) \\ \hline \end{array} \quad = \quad \begin{array}{|c|c|c|} \hline 1 & 3 & 2 \\ \hline 1 & 2 & 3 \\ \hline \end{array}$$

Since the goal image is supplied as a binary one and in order to keep user instruction as simple as possible, the 2D-lookup matrix used in the LUCIFER-framework contains only binary entries Black (0) and White (1).

## References


1. R.A. Brooks: A robust layered control system for a mobile robot. IEEE J. Robotics Autom., 2: 14–23, 1986
2. M. Cheriet, J. Said, C. Suen: A formal model for document processing of business forms. Montreal, Canada, 1995
3. C. Cracknell, A. Downtown: A color classification to form dropout. In: Proc. 6th Int. Workshop on Frontiers in Handwriting Recognition (IWFHR), Tajon, Korea, 1998
4. W.C. de Jong, L.N.K. van der Kooij, D.P. Schmidt: Computer-aided analysis of handwriting, the NIFO-TNO approach. In: 4th European Handwriting Conference for Police and Government Handwriting Experts, 1994
5. G. Dimauro, S. Impedovo, G. Pirlo, A. Salzo: Automatic bankcheck processing: a new engineered system. Int. J. Pattern Recognition Artif. Intell., 11: 467–504, 1997
6. S. Djeziri, F. Nouboud, R. Plamondon: Extraction of items from checks. In: Proc. 4th Int. Conf. on Document Analysis and Recognition (ICDAR), Ulm, Germany, 1997
7. D. Doermann, A. Rosenfeld: Recovery of temporal information from static images of handwriting. Int. J. Comput. Vision, 15: 143–164, 1995
8. B. Found: The Forensic Analysis of Behavioural Artefacts: Investigations of Theoretical and Analytical Approaches to Handwriting Identification. PhD thesis, LaTrobe University Bundoora, 1997
9. K. Franke, G. Grube: The automatic extraction of pseudodynamic information from static images of handwriting based on marked grayvalue segmentation. J. Forensic Doc. Exam., 11: 17–38, 1998
10. K. Franke, M. Köppen: Towards a universal approach to background removal in images of bankchecks. In: Proc. 6th Int. Workshop on Frontiers in Handwriting Recognition (IWFHR), Tajon, Korea, 1998
11. K. Franke, M. Köppen, B. Nickolay, S. Unger: Machbarkeits- und Konzeptstudie Automaisches System zur Unterschriftenverifikation. Tech. Rep., Fraunhofer, Berlin, Germany, 1996
12. S. Grossert, M. Köppen, B. Nickolay: A new approach to fuzzy morphology based on fuzzy integral and its application in image processing. In: Proc. ICPR'96, Vienna, Austria, 1996, pp. 625–630
13. E.J. Serra: Image Analysis and Mathematical Morphology. Volume 2: Theoretical Advances. Academic, London, 1988
14. M. Kamel, A. Zhao: Extraction of binary character/graphics images from grayscale document images. CVGIP: Graphical Models Image Process., 55: 203–217, 1993
15. M. Köppen, M. Teunis, B. Nickolay: A framework for the evolutionary generation of 2D-lookup based texture filters. In: T. Yamakawa, G. Matsumoto (eds.), Methodologies for the Conception: Design and Application of Soft Computing, Proc. 5th Int. Conf. on Soft Computing and Information / Intelligent Systems (IIZUKA'98), pp. 965–970, 1998
16. J. Koza: Genetic programming — On the programming of computers by means of natural selection. MIT, Cambridge, Mass., USA, 1992
17. L. Lee, M. Lizarraga, N. Gomes, A. Koerich: A prototype for Brazilian bankcheck recognition. Int. J. Pattern Recognition Artif. Intell., 11: 549–570, 1997
18. K. Liu, C. Suen, M. Cheriet, J. Said, C. Nadal, Y. Tang: Automatic extraction of baselines and data from check images. Int. J. Pattern Recognition Artif. Intell., 11: 675–697, 1997
19. L. Lohmann: Bildanalysesystem zur robusten Erkennung von Kennzeichen an Fahrzeugen. PhD thesis, Technical University, Berlin, Germany, 1999
20. M. Okada, M. Shirdhar: Extraction of user-entered components from a personal bankcheck using morphological subtraction. Int. J. Pattern Recognition Artif. Intell., 11: 549–570, 1997
21. M. Philipp: Fakten zu Fish. Das Forensische Informations-System Handschriften des Bundeskriminalamtes – Eine Analyse nach über 5 Jahren Wirkbetrieb: Tech. Rep., Kriminaltechnisches Institut 53, Bundeskriminalamt, Thaerstraße 11, 65173 Wiesbaden, Germany, 1996
22. J. Serra: Image Analysis and Mathematical Morphology. Academic, London, 1982
23. P. Soille: Morphological Image Analysis: Principles and Applications. Springer, Berlin Heidelberg New York, 1999
24. Y. Solihin, C. Leedham: Preparing handwriting images for forensic examination. In: Proc. 8th Biennial Conference of the International Graphonomic Society, 1997, pp. 125–126
25. M. Sugeno: Theory of Fuzzy Integral and Its Applications. PhD thesis, Tokyo Institute of Technology, Tokyo, 1974
26. R. Yager: On ordered weighted averaging aggregation operators in multi-criteria decision making. IEEE Trans. SMC, 18: 183–190, 1988
27. X. Ye, M. Cheriet, C. Suen, K. Liu: Extraction of bankcheck items by mathematical morphology. Int. J. Doc. Anal. Recognition, 2: 53–66, 1999
28. K. Yoshikawa, Y. Adachi, M. Yoshimura: Extracting the signatures from travelers' checks. In: Proc. 6th Int. Workshop on Frontiers in Handwriting Recognition (IWFHR), Tajon, Korea, 1998

**Katrin Franke** studied electrical engeneering and received her Diploma at the Technical University in Dresden. After finishing her studies in 1994, Katrin Franke went to the Fraunhofer-Institute for Production Systems and Design Technology Berlin and focused her research on digital image processing. Currently, she is a research assistant at the Department of Pattern Recognition where she is involved in various research as well as industrial projects in the field of document processing and handwriting analysis, in particular, signature verification for finance and forensic applications.

**Mario Köppen** studied physics at the Humboldt-University of Berlin and received his Diploma in solid state physics in 1991. Afterwards, he worked as research assistant at the Central Institute for Cybernetics and Information Processing in Berlin and changed his main research interests to image processing and neural networks. Since 1992, he has been with the Fraunhofer-Institute for Production Systems and Design Technology. He continues his work on image processing, in particular, on the application of softcomputing in image processing.