Comparison of Handwritings

Miroslava Božeková

2008

Comenius University
Faculty of Mathematics, Physics
and Informatics
Department of Applied Informatics

Miroslava Božeková

Comparison of Handwritings

Diploma thesis
Thesis advisor: Doc. RNDr. Milan Ftáčnik, CSc.

computer graphics

Bratislava 2008

Assignment of diploma thesis

Having one and more scanned images with handwritten text as input, the task is to create methods, which can determine whether the documents are written by the same person or not. The goal is implementation of methods and making experiments.

By this I declare that I wrote this diploma thesis by myself, only with the help of the referenced literature, under the careful supervision of my thesis advisor.

Bratislava, 7 May 2008 Miroslava Božeková

Acknowledgments

Sincere thanks to my thesis supervisor doc. RNDr. Milan Ftáčnik, CSc., my seminary supervisor doc. RNDr. Andrej Ferko, PhD., Mgr. Matej Novotný, my friend Tomáš Sako and the others for their regulations, suggestions and helpful comments. I am grateful to my family and friends for encouragement through my all studies.

Abstract

Title: Comparison of Handwritings
Author: Miroslava Božeková
Department: Department of Applied Informatics
Supervisor: Doc. RNDr. Milan Ftáčnik, CSc.

Abstract: The goal of this paper is to contribute to the solution of a following problem. Having one and more scanned images with handwritten text as input, our task is to create methods which can determine whether the documents are written by the same person or not. We concentrate on a fundamental problem - comparison of two images and decision whether two images are written by the same person or not - so-called writer verification problem. We present three approaches consisting of preprocessing, feature vector extraction and combination with graphemes clustering. The first approach is based on feature vector, the second is combination of the first and Kohonen's Self-Organizing Map and finally the third joins the first approach and modified hierarchical clustering. We made our experiments on 100 images from 40 different writers. These images were taken from the IAM Handwriting Database which contains 1539 pages of scanned text from 657 writers. In these experiments we have tested three approaches mentioned on two input images. We achieved 96,5 % accuracy by the third approach. A working implementation is presented together with experimental results.

Keywords: handwriting, writer verification, grapheme, clustering

Contents

List of Figures

List of Tables

1 Introduction

Handwriting is one of the most fundamental stone of civilization. It is a system of graphic signes which are agreed by certain human society. Handwriting serves for recording of ideas, feelings, emotions and postures which can be expressed by language. Birth of the handwriting essentially sped up evolution of civilization. Massive expansion of handwriting was accompanied by a new sign - individuality of handwriting of single writers. People differ from each other e.g. by their expressions of speech, i. e. physical attitudes, gestures, postures, mimicry, handwriting, colouring and modulation of voice etc. Writing is possible to understand as an automatic process. From school age, children study and pick up writing movements. They modify movements to write the most exactly according to given school model. Fluency of writing movement increases by drill and it becomes more involuntary. Writer doesn't have to think of writing movement after its total automation. W. Preyer (1841 - 1897) understands writing movement (which is performed by 500 muscles) as a process controlled by brain, relatively independent on hand movement. We can find two same neither people nor handwritings. Legal significance of signature is based on the individuality of handwriting and on its relative uniformity. On the other hand, handwriting can be temporary influenced by short-term psychical state [1] [2].
We would like to show that everybody writes differently. Individuality in handwriting can be illustrated as in figure 1. In this example, eight authors provide three handwriting samples each of the word "referred". As can be see, the variation within a person's handwriting (within-author variation) is less than the variation between the handwriting of two different people (between-author variation) [3].

Figure 1: Variability in handwriting. Eight authors provided three handwriting samples each, showing within and between author variations. Figure comes from [3].

1.1 The analysis of handwriting

The analysis of handwriting crops up in many diverse applications [3] as given in figure 2.

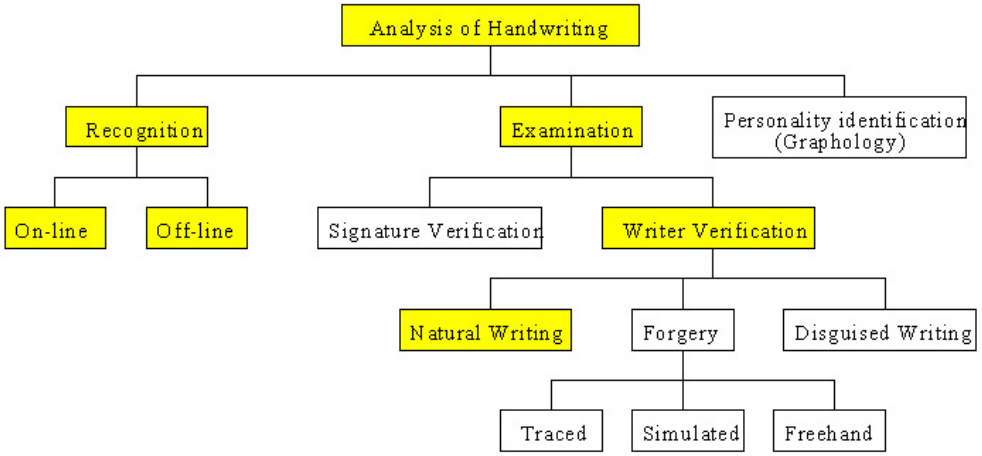

Figure 2: Taxonomy of topics in handwriting analysis by [3].

1.1.1 Recognition

Handwriting recognition is the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. For English orthography, as with many languages based on the Latin alphabet, this symbolic representation is typically the 8-bit ASCII representation of characters [4].

1.1.2 Handwriting interpretation

Handwriting interpretation is the task of determining the meaning of a body of handwriting, e.g., a handwritten address [4].

1.1.3 Indexing and searching collections of handwritten historical documents

Libraries contain an enormous amount of handwritten historical documents. Such collections are interesting to a great range of people, be it for historians, students or just curious readers. Efficient access to such collections (e.g. on digital media or on the Internet) requires an index, for example like in the back of a book. Handwriting recognizers do not perform well on such noisy documents as historical. The wordspotting idea has been proposed as an alternative to OCR solutions for building indexes of handwritten historical documents, which were produced by a single author: this ensures that identical words, which were written at different times, will have very similar visual appearances. This fact can be exploited by

clustering words into groups with image matching techniques. Ideally, each cluster of word images consists of all occurrences of a word in the analyzed document collection [5].

1.1.4 Signature verification

Signature is a socially accepted authentication method and is widely used as proof of identity in our daily life. Automatic signature verification by computers has received extensive research interests in the field of pattern recognition. In the context of signature verification, tested signatures whether genuine or skillfully forged, usually have similar shapes with the registration ones (if a tested signature has a shape very different from the registration, it can be easily identified as a forgery) [6].

1.1.5 Writer verification

Writer verification involves a one-to-one comparison with a decision as to whether or not the two samples are written by the same person [7] (see figure 3). Writer verification can based on natural writing, forgery or disguised writing (see figure 2).

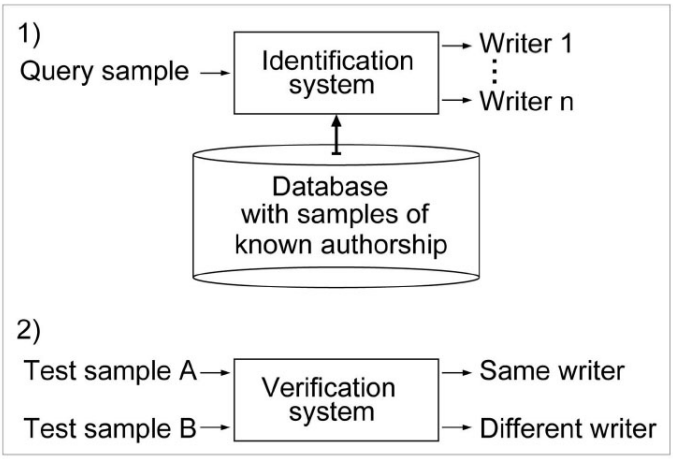

Figure 3: (1) A writer identification system retrieves, from a database containing handwritings of known authorship, those samples that are the most similar to the query. The hit list is then analyzed in detail by a human expert. (2) A writer verification system compares two handwriting samples and makes an automatic decision as to whether or not the input samples were written by the same person. Figure comes from [7].

1.1.6 Writer identification

A writer identification system performs a one-to-many search in a large database with handwriting samples of known authorship and returns a likely list of candi-

dates [7] (see figure 3).

1.1.7 Graphology

Graphology is the analysis of the psychological structure of the human subject through his or her handwriting. The central nervous system provides a direct and undistorted link to the deeper self. Every human mind comprises a unique and immensely complex blend of character and accumulated experiences of life. Handwriting reflects this by evolving constantly. No two samples are the same [8].
The way we learn to write in school is standard. Every teacher in school teaches their students to write cursive in a uniform way. However, we choose to 'deviate' or to do things differently then the teacher taught us. That is why every single person's individual handwriting reflects what is unique about them [9].
Graphology is based on approximately 250 signs which are grouped under the following headings: layout, dimension, pressure, form, speed, continuity and direction [10].

1.2 Categories of methods

1.2.1 Text-dependent versus text-independent methods

The text-dependent methods are very similar to signature verification techniques and use the comparison between individual characters or words of known semantic content. These methods therefore require prior localization and segmentation of the relevant information, which is usually performed interactively by a human user. The text-independent methods for writer identification and verification use statistical features extracted from the entire image of a text block. A minimal amount of handwriting (e.g. a paragraph containing a few text lines) is necessary in order to derive stable features insensitive to the text content of the samples [7].

1.2.2 Off-line and on-line methods

Handwriting data is converted to digital form either by scanning the writing on paper or by writing with a special pen on an electronic surface such as a digitizer combined with a liquid crystal display. The two approaches are distinguished as off-line and on-line handwriting, respectively. In the on-line case, the two-dimensional coordinates of successive points of the writing as a function of time are stored in order, i.e., the order of strokes made by the writer is readily available. In the off-line case, only the completed writing is available as an image. The on-line case deals with a spatio-temporal representation of the input, whereas the off-line case involves analysis of the spatio-luminance of an image [4].

2 Theoretical background

We would like to explain some terms from image processing and make clear cluster analysis.

2.1 Image processing

2.1.1 Thresholding

Converting a greyscale image into a binary image is an important step. Foreground (in our case handwriting) must be separated from the background of the image. After the thresholding, background has white color and foreground black. Obtained handwriting should be unchanged (according to comparation with the original image). If background doesn't contain black pixels and foreground doesn't contain white pixel, the resulting binarization is acceptable.

Classification
Most of the existing thresholding methods can be allocated into four main classes [12].

- **histogram-based techniques**
  The structure of the peaks, valleys and curvatures in the smoothed greyscale histogram is analysed to determine the final global threshold value (e.g. Otsu's method, Solihin and Leedham,...).
- **entropy-based techniques**
  Entropy is used to separate the global thresholding classes. For example, the optimal threshold value can be calculated by maximising the sum of the foreground and background entropies. i.e. maximally separating region intensities of the foreground and background (e.g. Pun, Kapur et al., Brink,...).
- **local adaptive techniques**
  (e.g. Niblack, Zhang and Tan, Bernsen,...)
- **other global techniques**
  (e.g. Gorman, Gu et al.,...)

In global thresholding, a single threshold value is used to separate the foreground and the background of an image. Global thresholding is attractive because it is simple and is sufficient in many cases. However, when the image is unevenly illuminated, local thresholding may be necessary. In local thresholding, a threshold value is assigned to each pixel to determine whether it is a foreground or background pixel using local information from the image [13].
Methods can be also classified into two categories [13]:

- **one stage thresholding and**
  Traditional thresholding finds a threshold value in one stage.

- **multi-stage thresholding**
  The multi-stage thresholding approach is a generalisation of traditional thresholding.

Global Multi-stage Thresholding
Global multi-stage thresholding can be viewed as a process of reducing the search space of threshold candidate values stage by stage where each stage uses different information from the image until the final stage chooses the final threshold value [13]. The difference between traditional thresholding, iterative thresholding, and multi-stage thresholding techniques is illustrated in figure 4.

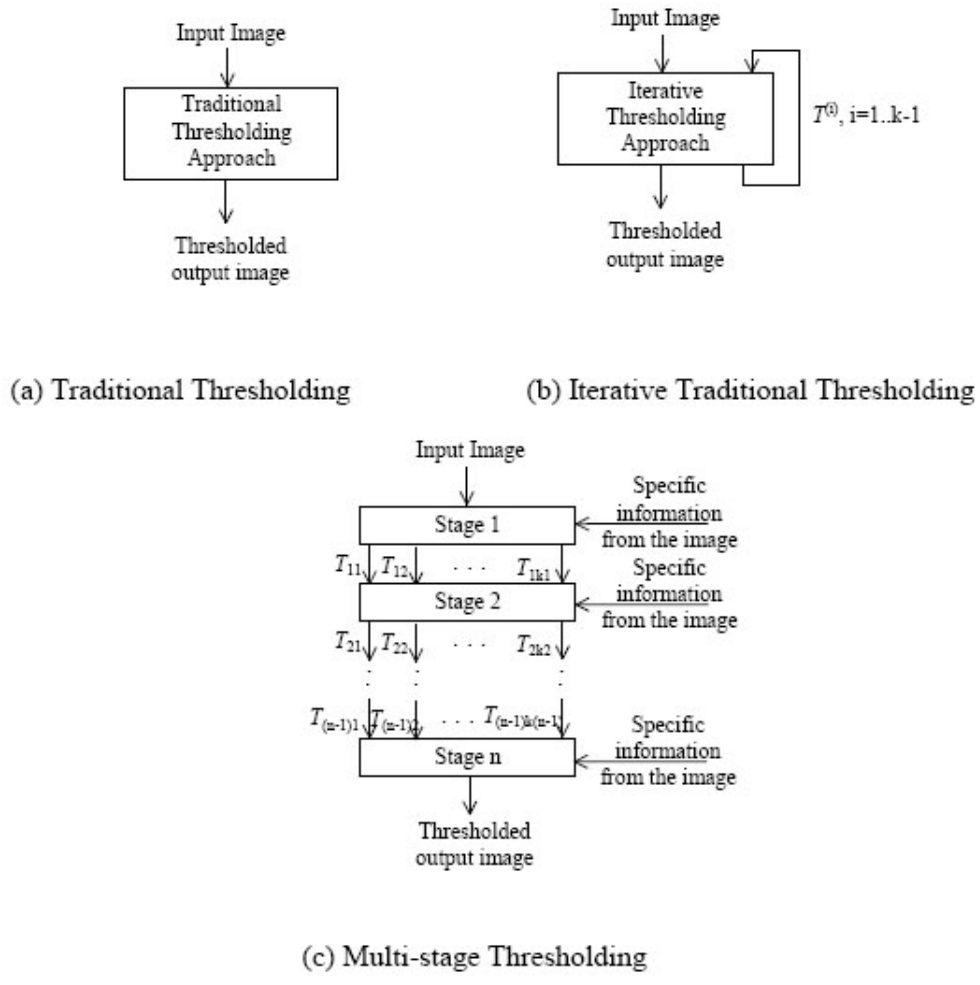

Figure 4: Three different approaches to global thresholding. The figure comes from [13].

2.1.2 Horizontal and vertical projection profile

The horizontal/vertical projection profile (HPP, VPP) is the histogram of the number of black pixels along horizontal/vertical scan lines [15] (see figure 5).

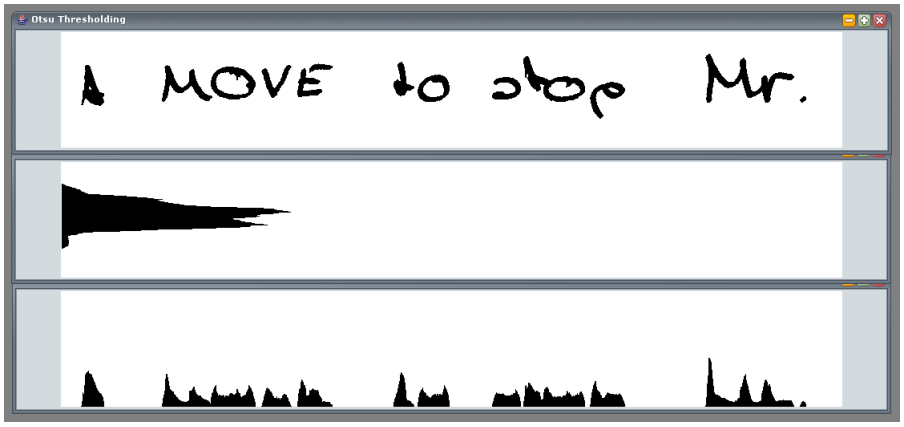

Figure 5: Horizontal (in the center) and vertical projection profile (below) of the image occuring upwards.

2.1.3 Averaging filter

The averaging filter is the most simple method for smoothing of input image. A new grey level value of pixel is obtained as arithmetic average of values from its neighbourhood. The most used is 3x3 filter window:

$$h_1 = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, h_2 = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}, h_3 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

5x5 (7x7) filter window is made analogically [14].

2.1.4 Contour Tracing

Contour Tracing (or boundary following) is a technique that is applied to digital images in order to extract their boundary [16].

Moore-Neighborhood Tracing [16]
The Moore neighborhood (also known as the 8-neighbors or indirect neighbors) of pixel P, is the set of 8 pixels which share a vertex or edge with that pixel (these pixels are namely pixels P1, P2, P3, P4, P5, P6, P7 and P8 shown in figure 6). Let's introduce the idea behind Moore-Neighborhood tracing. Given a group of black pixels on a background of white pixels, we locate a black pixel and declare it as our "start" pixel. We start at the top left corner of the grid, scan each column of pixels from the top going below - starting from the leftmost column and proceeding to the right - until we encounter a black pixel. We declare that pixel as our "start" pixel. Now, we stand on the start pixel. Without loss of generality, we extract the contour by going around the pattern in a clockwise direction. (It doesn't matter which direction you choose as long as you stick with your choice throughout the algorithm). The general idea is: every time you hit a black pixel P, backtrack i.e. go back to the white pixel you were previously

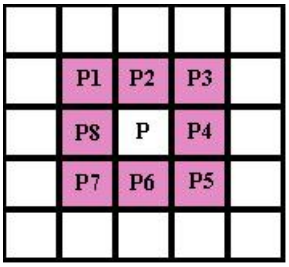

Figure 6: Moore neighborhood. The figure comes from [16].

standing on, then go around pixel P in a clockwise direction, visiting each pixel in its Moore neighborhood, until you hit a black pixel. The algorithm terminates when the start pixel is visited for a second time. The black pixels you walked over will be the contour of the pattern.
Table 1 describes a formal algorithm. An animated demonstration and more contour tracing algorithms can be found in [16].

2.2 Clustering

Clustering can be considered the most important *unsupervised learning* problem (i.e., the training data doesn't specify what we are trying to learn). So, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data. A loose definition of clustering could be "the process of organizing objects into groups whose members are similar in some way". A cluster is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters [17].
Clustering algorithms can be applied in many fields, for instance [17]:

- Marketing (finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records)
- Biology (classification of plants and animals given their features)
- Libraries (book ordering)
- Insurance (identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds)
- City-planning (identifying groups of houses according to their house type, value and geographical location)
- Earthquake studies (clustering observed earthquake epicenters to identify dangerous zones)
- WWW (document classification; clustering weblog data to discover groups of similar access patterns).

Input: A square tessellation T containing a connected component P of black cells.
Output: A sequence B $(b_1, b_2, ..., b_k)$ of boundary pixels i.e. the contour.
Define $M(a)$ to be the Moore neighborhood of pixel a.
Let p denote the current boundary pixel.
Let c denote the current pixel under consideration i.e. c is in $M(p)$.

Begin

- Set B to be empty.
- From top to bottom and left to right scan the cells of T until a black pixel, s, of P is found.
- Insert s in B.
- Set the current boundary point p to s i.e. $p = s$
- Backtrack i.e. move to the pixel from which s was entered.
- Set c to be the next clockwise pixel in $M(p)$.
- **while** c not equal to s **do**

  If c is black

  - insert c in B
  - set $p = c$
  - backtrack (move the current pixel c to the pixel from which p was entered)

  else

  - advance the current pixel c to the next clockwise pixel in $M(p)$

  end while

End

Table 1: A formal description of the Moore-Neighborhood tracing algorithm. The algorithm comes from [16].

Clustering algorithms may be classified as listed below:

- Hierarchical algorithms
- Non-hierarchical algorithms

2.2.1 Hierarchical algorithms

Hierarchical algorithms can be classified as follows [21]:

- Agglomerative (build-up) methods (produce a classification in a bottom-up manner; at the beginning, each object is an individual cluster and gradually in each step two the closest clusters are joined into one until one big cluster containing all objects doesn't create)
- Divisive methods (generate a classification in a top-down manner; at the beginning one big cluster containing all objects is given and objects are gradually divided into smaller clusters)

In agglomerative clustering, we have several methods for definition of similarity between groups (clusters). Methods differ each other by distances between clusters [20] (see figure 7):

- single linkage (the nearest neighbor method)
- complete linkage (the furthest neighbor method)
- average linkage (unweighted pair-group method using arithmetic averages)
- centroid method (unweighted pair-group method using centroids)

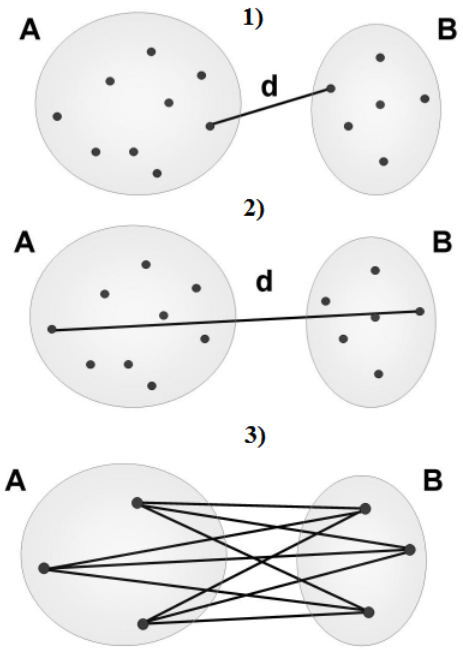

Figure 7: 1) single linkage, 2) complete linkage 3) average linkage. Figures come from [20].

Result of hierarchical algorithm is represented by a dendrogram, see figure 8.

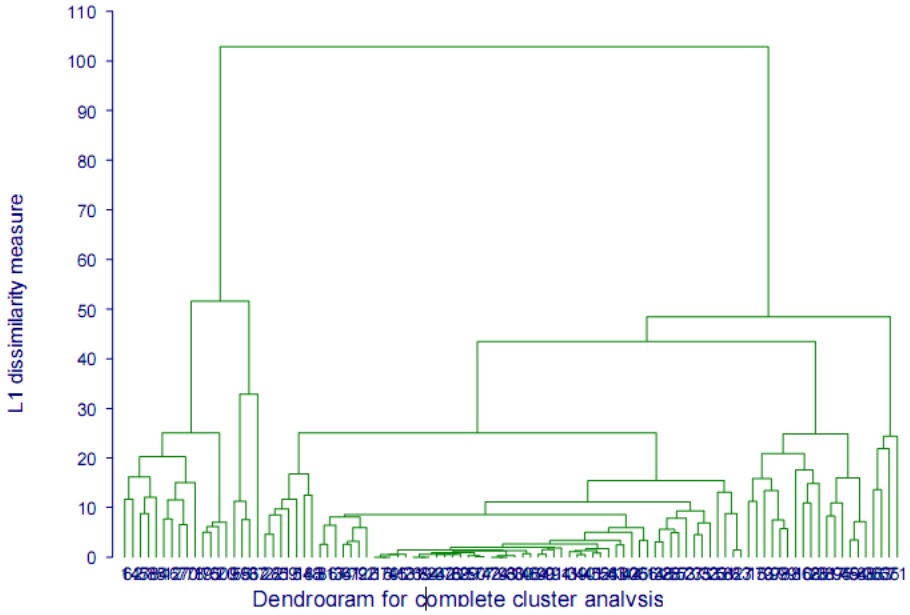

Figure 8: A dendrogram. Figure comes from [18].

2.2.2 Non-hierarchical algorithms

A non-hierarchical method generates a classification by partitioning a dataset, giving a set of (generally) non-overlapping groups having no hierarchical relationships between them [19]. Four of the main categories of non-hierarchical method are [19]:

- relocation (such as k-means, assign objects to a user-defined number of seed clusters and then iteratively reassign objects to see if better clusters result. Such methods are prone to reaching local optima rather than a global optimum, and it is generally not possible to determine when or whether the global optimum solution has been reached)
- single-pass methods (produce clusters that are dependent upon the order in which the objects are processed)
- nearest neighbour (such as the Jarvis-Patrick method, assign objects to the same cluster as some number of their nearest neighbours. User-defined parameters determine how many nearest neighbours need to be considered, and the necessary level of similarity between nearest neighbour lists)
- other non-hierarchical methods (e.g. Self-Organizing Map)

Kohonen's Self-Organizing Map
The self-organizing map (SOM) was developed by professor Teuvo Kohonen in 1995. SOM network is a special type of neural network that can learn from complex, multi-dimensional data and transforn them into visually decipherable

clusters. The theory of the SOM network is motivated by the observation of the operation of the brain. Various human sensory impressions are neurologically mapped into the brain such that spatial or other relations among stimuli correspond to spatial relations among the neurons organized into a two-dimensional map [22].
The SOM network performs unsupervised training; that is, during the learning process the processing units in the network adjust their weights primarily based on the lateral feedback connections. Unsupervised learning does not require the knowledge of target values. The nodes in the network converge to form clusters to represent groups of entities with similar properties. SOM networks combine competitive learning with dimensionality reduction by smoothing the clusters with respect to an a priori grid and provide a powerful tool for data visualization. However, the output of an SOM network does not automatically provide groupings of the points on the map. The current practise is to design the Kohonen SOM map so that the number of nodes on the map matches the desired number of clusters. For example, a 2x2 network has four nodes hence forms four groups. However, often times it is difficult to design a two-dimensional map for a problem with small and/or odd number of clusters (e.g. 3 clusters). A common problem when using SOM network is that the number of nodes on the output map is more than the number of target groups [22].
The SOM network typically has two layers of nodes, the input layer and the Kohonen layer. The input layer is fully connected to a two-dimensional Kohonen layer. During the training process, input data are fed to the network through the processing elements (nodes) in the input layer. An input pattern $\vec{x_v}(v = 1, ..., V)$ is denoted by a vector of order m as: $\vec{x_v} = (x_{v1}, x_{v2}, ..., x_{vm})$, where x_{vi} is the ith input signal in the pattern and m is the number of input signals in each pattern. An input pattern is simultaneously incident on the nodes of a two-dimensional Kohonen layer. Associated with the N nodes in the n x n ($N = n$ x n) Kohonen layer, is a weight vector, also of order m, denoted by: $\vec{w_i} = (w_{i1}, w_{i2}, ..., w_{im})$, where w_{ij} is the weight value associated with node i corresponding to the jth signal of an input vector [22].
As the training process proceeds, the nodes adjust their weight values according to the topological relations in the input data. The node with the minimum distance is the winner and adjusts its weights to be closer to the value of the input pattern. The most common way of measuring distance between vectors is Euclidean distance [22].

Weight adaptation function
After the finding the winner, the next step is adaptation of weights - learning:

$$\vec{w_i}(t+1) = \vec{w_i}(t) + \alpha(t).h(i^*, i).[\vec{x_v} - \vec{w_i}(t)] \quad (1)$$

where t is time, $\alpha(t) \in (0, 1)$ is adaptation coefficient represents various rate of learning which descents with time to zero what ensures the end of process of learning, i^* is an index of the winner neuron, $h(i^*, i)$ is neighborhood kernel centered on the winner unit and defines region of cooperation between neurons

(how much weight vectors corresponding to neurons in neighborhood of the winner will be adapted) (see figure 9) [23].

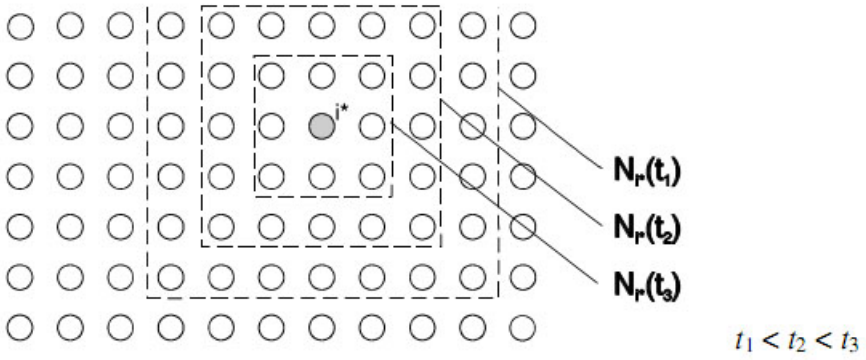

Figure 9: An example of neighborhood of the winner neuron i^*, defined for different time moments, $t_1 < t_2 < t_3$. Image comes from [23].

The most simple used function is orthogonal neighborhood:

$$\mathrm{h(i^*}, i) = \begin{cases} 1 & \text{if } \mathrm{d}_M(i^*, i) \leq \lambda(t) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $d_M(i^*, i)$ represents distance of type "Manhattan" between neurons i^* and i in grid of map. The second often used choice is Gaussian type neighborhood:

$$\mathrm{h(i^*}, i) = \exp\left(-\frac{d_E^2(i^*,i)}{\lambda^2(t)}\right) \tag{3}$$

where $d_E(i^*, i)$ represents euclidean distance of neurons i^* and i in grid. Parameter $\lambda(t)$ decreases with time to zero what ensures decreasing of neighborhood during learning [23].

3 Recent research

In this section, we will survey the related work of preprocessing of handwritten images (thresholding, line segmentation, slant detection and correction, word segmentation and grapheme segmentation and normalization) and writer verification problem.

3.1 Thresholding

3.1.1 Characteristics of handwriting images [13]

1. The foreground (handwriting) has far fewer pixels compared to the background.

2. The location of the peak of the foreground is usually near the darkest intensity, while the location of the peak of the background often varies considerably.

3. Though the foreground pixels are fewer, they tend to be more scattered than the background pixels. The distribution of the foreground depends heavily on the type of writing implement. For example, a fibre-tipped pen has a concentrated distribution, but a ballpoint pen or a pencil has a scattered distribution. It is often observed that the foreground pixels lie very near to the background peak if the handwriting is written using a ballpoint pen or a pencil.

4. Experiments show that the best threshold values for handwriting images lie nearer to the background peak than the foreground peak.

3.1.2 A brief summary of thresholding techniques

If you are interesting in thresholding methods, in [24] are compared 40 selected thresholding methods from various categories. We chose the following algorithms which are used for handwritings or for historical document images (with handwritten text). A voluminous summary of thresholding methods for historical document images can be found in [12].
Otsu's thresholding algorithm 1979 [25] chooses the optimal threshold value by maximizing the between-class variance. If the gray level of pixels on an image ranges in L [1, 2,..., L] and the number of pixels at level i is denoted n_i, the total number of pixel can be calculated by equation: N=n_1+n_2+n_3+...+n_i+...+n_L. For bi-level thresholding, the pixels sets [1, 2,..., L] are divided into two classes; C_1 = [1, 2,..., t] and C_2= [t+1,..., L]. Otsu's algorithm determines the optimal

threshold value (t) that maximizes the between-class variance, σ_B^2

$$\begin{aligned} & \sigma_B^2(t) = \omega_1(t)\omega_2(t)(\mu_2(t) - \mu_1(t))^2 \\ \text{where} \\ & \omega_1(t) = \textstyle\sum_{i=1}^{t} p_i,\ \omega_2(t) = \sum_{i=t+1}^{L} p_i \\ & \mu_1(t) = \textstyle\sum_{i=1}^{t} ip_i/\omega_1(t),\ \mu_2(t) = \sum_{i=t+1}^{L} ip_i/\omega_2(t), \\ \text{and} \quad & \mathrm{p}_i = n_i/N \text{ [26]}. \end{aligned} \tag{4}$$

Pun 1981 [27] presented a maximum-entropy-based method. It used Shannon's concept to define the entropy of an image. Pun used this concept to derive an expression for an upper bound of the aposteriori entropy. The expression was finally used to threshold an image [12].
Kapur et al 1985 [28] reported an improvement of Pun's method. It is a histogram analysis and maximum-entropy-based technique, which uses the maximum of the sum of the entropy of the grey-level distribution of the foreground and background [12].
Bernsen 1986 [29] proposed a local thresholding technique based on neighbours of each pixel. It has proven to be a fast algorithm. The disadvantage is that it does not work well when the background regions have varying greylevel intensities and contain ghost images [12].
Niblack's algorithm 1986 [31] is a local thresholding method based on the calculation of the local mean and of local standard deviation. The threshold is decided by the formula:

$$T(x, y) = m(x, y) + k \bullet s(x, y), \tag{5}$$

where $m(x, y)$ and $s(x, y)$ are the average of a local area and standard deviation values, respectively. The size of the neighborhood should be small enough to preserve local details, but at the same time large enough to suppress noise. The value of k is used to adjust how much of the total print object boundary is taken as a part of the given object [30].
Yanowitz and Bruckstein 1989 [32] suggested using the grey-level values at high gradient regions as known data to interpolate the threshold surface of image document texture features. The key steps of this method are [30]:

1. Smooth the image by average filtering.
2. Derive the gradient magnitude.
3. Apply a thinning algorithm to find the object boundary points.
4. Sample the grey-level in the smoothed image at the boundary points. These are the support points for interpolation in step 5.
5. Find the threshold surface $T(x, y)$ that is equal to the image values at the support points and satisfies the Laplace equation using Southwell's successive over relaxation method.

6. Using the obtained $T(x, y)$, segment the image.

7. Apply a post-processing method to validate the segmented image.

Wang and Pavlidis 1993 [33] assumed that the grey-scale image is a surface with top logical features corresponding to the shape features of the original image. Each pixel of the image was classified as: peak, pit, ridge, ravine, saddle, flat or hillside. Rules can be built based on the estimated first and second directional derivatives of the underlying image intensity surface. The characters can be extracted according to the rules and the features [12].
Brink 1995 [34] proposed a minimum-spatial-entropy based global thresholding algorithm [12].
Solihin and C.Leedham 1999 [35] investigated two global techniques: native integral ratio (NIR) and quadratic integral ratio (QIR). The QIR method is a global two-stage thresholding approach. In the first stage, the image is divided into three classes of pixels: foreground, background and a fuzzy class where it is hard to determine whether a pixel actually belongs to the foreground or the background. During the second stage, a final threshold value is chosen in the fuzzy region [30].
Zhang and Tan 2001 [36] proposed an improved version of Niblack's method:

$$T(x, y) = m(x, y) \bullet [l + k \bullet (1 - \tfrac{s(x,y)}{R})], \tag{6}$$

where k and R are empirical constants. The improved Niblack method uses parameters k and R to reduce its sensitivity to noise [30].

3.2 Line Segmentation

Many approaches such as **Yanikoglu and Sandon** 1998 [38], **Kavallieratou et al** 2003 [39] and **Weliwitage et al** 2005 [40], use global/piece-wise projection profile but fail to recognize when the projection profile information is useful and when it is not.
The method explained in **Feldbach and Tönnies** 2001 [41], requires parameters to be set, according to the type of handwriting.
The Cut Text Minimization method (**Weliwitage et al** [40]), fails to detect short lines and those which do not begin in the beginning of the document. Skew detection methods [42], [43] and baseline estimation methods [41], [42], are not flexible to capture the variation in handwriting.
The method in **Nicolas, Paquet and Heutte** 2004 [44] assumes that each connected component belongs to one line, which is not the case in documents with lines running into each other. Clustering algorithm based on heuristics have been used for line segmentation in the CEDAR-FOX system [45], [46]. The systems algorithm was originally designed for handwritten postal envelopes but the heuristics do not generalize well to the varations encoutered in other handwritten documents [37].
Arivazhagan, Srinivasan and Srihari 2007 [37] proposed an approach involving the use of bivariate Gaussian densities to model lines. In this way, the skew of

the document is captured by the covariance of the Gaussian density. Piece-wise projection profiles are used to obtain an initial set of candidate line starting positions. The line drawn traverses around any obstructing handwritten component by associating it to the above or the below line. The algorithm also automatically recognizes lines running into each other and cuts through at the most appropriate position.

3.3 Slant correction

Slant is deviation of average near-vertical strokes from the vertical direction [48] (see figure 10).

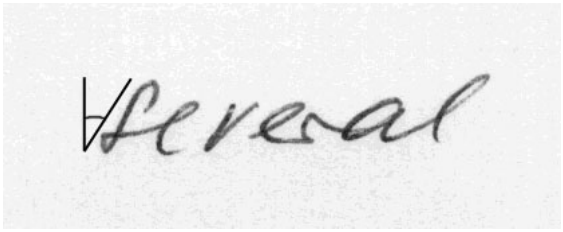

Figure 10: Slant angle

Several techniques for estimating word slant have been proposed. Uniform slant correction techniques work with the assumption that each word is written with a constant slant. However, it is a more widely acceptable assumption that the slant angle fluctuates in a word due to various factors such as writer's habit, the inherent shape of each character, and writing position. Therefore nonuniform techniques (work with this assumption) estimate local slant angles [47].
Bozinovic and Srihari 1989 [49] and **Kim and Govindaraju 1997** [50] have proposed slant correction techniques where the average slant angle is estimated from the angles of extracted vertical strokes [47].
Guillevic and Suen 1994 [51], **Kavallieratou et al.** 2000 [52], and **Nicchiotti and Scagliola** 1997 [53] analyzed a set of projection histograms for the estimation of the average slant angle [47].
Kavallieratou et al. 2000 [52] proposed a slant estimation algorithm based on the use of vertical projection profile of word images and the Wigner-Ville distribution [48].
Kimura et al. 1993 [54], **Simoncini and Kovács** 1995 [55], **Ding et al.** 2000 [56] and **Britto et al.** 2000 [57] utilized statistics of chain-coded stroke contours [47].
Vinciarelli et al. 2001 [58] proposed a technique based on a cost function which measures slant absence across the word image. The cost function is evaluated on multiple shear transformed word images. The angle with the maximal cost is taken as a slant estimate [48].
Uchida et al. 2001 [47] presented a nonuniform slant correction technique where the slant correction problem is formulated as an optimal estimation problem of

local slant angles at all horizontal positions. The optimal estimation is governed by a criterion function and several constraints designed to evaluate global and local goodness of the estimated local angles. The optimal local slant angles which maximize the criterion function satisfying the constraints are searched for efficiently by a dynamic programming (DP)-based algorithm [47].
Hase et al. 2001 [60] have proposed a realignment technique for inclined and curved texts, where the inclination and the curve of a text is approximated by some quadratic functions and then its component characters are realigned horizontally and slant-corrected using those functions. This technique, however, will not be appropriate for the slant correction of handwritten words whose component characters have slants regardless of the shape of their text line [59].
Dong et al. 2005 [48] presents fast and robust algorithm for slant corrections based on Radon transform. Radon transform is used to estimate the long strokes and a word slant is measured by the average angle of these long strokes. Compared with the previous methods, these two algorithms do not require the setting of parameters heuristically. Moreover, the algorithms perform well on words of short length, where the traditional methods usually fail [48].

3.4 Word segmentation

Line separation is usually followed by a procedure that separates the text line into words. Few approaches in the literature have dealt with word segmentation issues [4].
Among the ones that have dealt with segmentation issues, most focus on identifying physical gaps using only the components, like **Mahadevan and Nagabushnam** 1995 [65] and **Seni and Cohen** 1994 [66]. These methods assume that gaps between words are larger than the gaps between the characters. However, in handwriting, exceptions are commonplace because of flourishes in writing styles with leading and trailing ligatures [4]. **Seni and Cohen** [66] evaluate eight different distance measures between pairs of connected component for word segmentation in handwritten text. In **Mahadevan and Nagabushnam** [65] the distance between the convex hulls is used [69].
Srihari et al 1997 [67] present techniques for line separation and then word segmentation using a neural network [69]. Another method **Kim and Govindaraju** 1998 [68] incorporates cues that humans use and does not rely solely on the one-dimensional distance between components. The author's writing style, in terms of spacing, is captured by characterizing the variation of spacing between adjacent characters as a function of the corresponding characters themselves. The notion of expecting greater space between characters with leading and trailing ligatures is enclosed into the segmentation scheme [4].
Feldbach and Tonnies 2003 [62] present a system using constraints on the semantics to segment the date from church registers using a neural network. **Marti and Bunke** 2001 [63] propose a full-page word segmentation algorithm and the evaluation is done by using the IAM database. **Manmatha and Rothfeder** 2005 [64] described a scale space approach for segmenting words from historical

handwritten documents [61].

Huang and Srihari 2008 [61] proposed a gap metrics based approach. Their approach has two main differences from previous methods. First of all, the gap metrics is computed by combining three different distance measures (see figure 11), which avoids the weakness of each of the individual one and thus provides a more reliable distance measure. Secondly, besides the local features, such as the current gap, a new set of global features are also extracted to help the classifier to make a better decision. The classification is done by using a three-layer neural network [61].

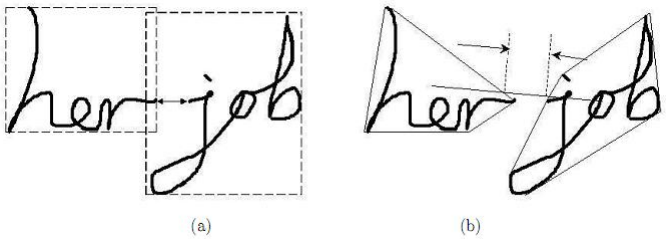

Figure 11: Examples of types of distance measures between a pair of connected components. The bounding box method and minimum run-length method are shown in (a), and the convex hull distance is shown in (b) (image was taken from [61]).

3.5 Writer verification

Several widely acknowledged efforts have been presented in recent years in the writer verification problem.

Cha and Srihari [70] model the problem as a two class classification problem: authorship or non-authorship. Feature distance is computed and the dichotomizer takes this feature distance vector as an input and outputs the authorship. They use 12 feature distances and achieved 97% accuracy on 1000 writers with 3 sample documents per writer.

Bensefia et al. [71] use sample set of 88 writers and extract the set of graphemes in two input handwritten documents. A grapheme feature set G is extracted thanks to a particular sequential clustering technique: G = { g_1 , g_2 , g_3 ,..., g_N }. Some of these features may occur on the two documents while the others may occur specifically on one single document. Writer verification is based on the mutual information between the grapheme distributions in the two handwritings that are compared.

Schlapbach and Bunke [72] use HMM based recognizers. For each writer, they build an individual recognizer and train it on text lines of that writer. This gives their recognizers that are experts on the handwriting of exactly one writer. In the identification or verification phase, a text line of unknown origin is presented to each of these recognizers and each one returns a transcription that includes the log-likelihood score for the considered input. Using over 8,600 text lines from 120

writers an Equal Error Rate(EER) of about 2.5% is achieved (EER quantifies in a single number the writer verification performance).
A statistical model of the task and a large number of features divided into two categories: macro-features which capture the global characteristics of the writers individual writing habit and style and micro-features which capture finer details at the character/word level are proposed by **Srihari et al.** [73]. Same writer accuracy was 94.6% and different writer accuracy was 97.6%.
Bulacu and Schomaker 2007 [7] developed new and very effective techniques for automatic writer identification and verification that use probability distribution functions (PDFs) extracted from the handwriting images to characterize writer individuality. Their methods operate at two levels of analysis: the texture level and the character-shape (allograph) level. The probability distribution of grapheme usage is computed using a common codebook of shapes obtained by clustering. Combining multiple features yields increased writer identification and verification performance. They use data sets from 3 databases Firemaker, IAM and ImUnipen.

3.6 CedarFox - Forensic Document Examination System

CedarFox [74] is a system for analyzing complex documents, particularly those that are handwritten. The system is primarily designed for interactive use by Forensic/ Questioned Document Examiners. It is also document image processing system for use with scanned documents. The system also can be used to archive documents and search them. Software functionalities:

- Handwriting Identification
- Writer and Signature Verification
- Image Processing - CedarFox has several image processing tools such as underline and rule-line removal, background removal, etc.
- Handwriting Segmentation - CedarFox is able to separate words of text in a handwritten document.
- Searching handwritten documents
- Handwriting Recognition
- Legibility and Readability Analysis - Word gap comparison and comparison with Palmer metrics is supported.
- System Utilities - CedarFox has user interfaces for scanning documents directly as well as for entering the results directly into spread-sheets and for printing intermediate results.

4 Analysis and design of application

Our approach is to create methods which can determine whether one and more input documents (with handwritten text) are written by the same person or not. We concentrate on a fundamental problem - comparison of two images and decision whether two images are written by the same person or not. We decided for off-line (described in section 1.2.2) and text-independent methods (described in section 1.2.1). Our primary approach is method based on natural writing, we don't detect forgery. We try to minimize human intervention to system. A working implementation is presented together with experimental results. Our application works on English handwritten text, but should be applicable to any Latin-based language.

4.1 Name and Logo of application

- Name: Adel
- Logo: see figure 12

Figure 12: Logo

4.2 Input data

Input data are scanned images. We decide to use the IAM Handwriting Database because it's a huge database suitable for our experiments. The IAM Handwriting Database [75] contains forms of handwritten English text which can be used on training and testing handwritten text recognizers and performing writer identification and verification experiments. The database contains forms of unconstrained handwritten text, which were scanned at the resolution of 300 dpi and saved as PNG images with 256 gray levels (see figure 13).

4.3 Integrated development environment

We use programming language *Java* and an open development platform *Eclipse*. Another option is *NetBeans IDE*. Both Eclipse and Netbeans are popular and it

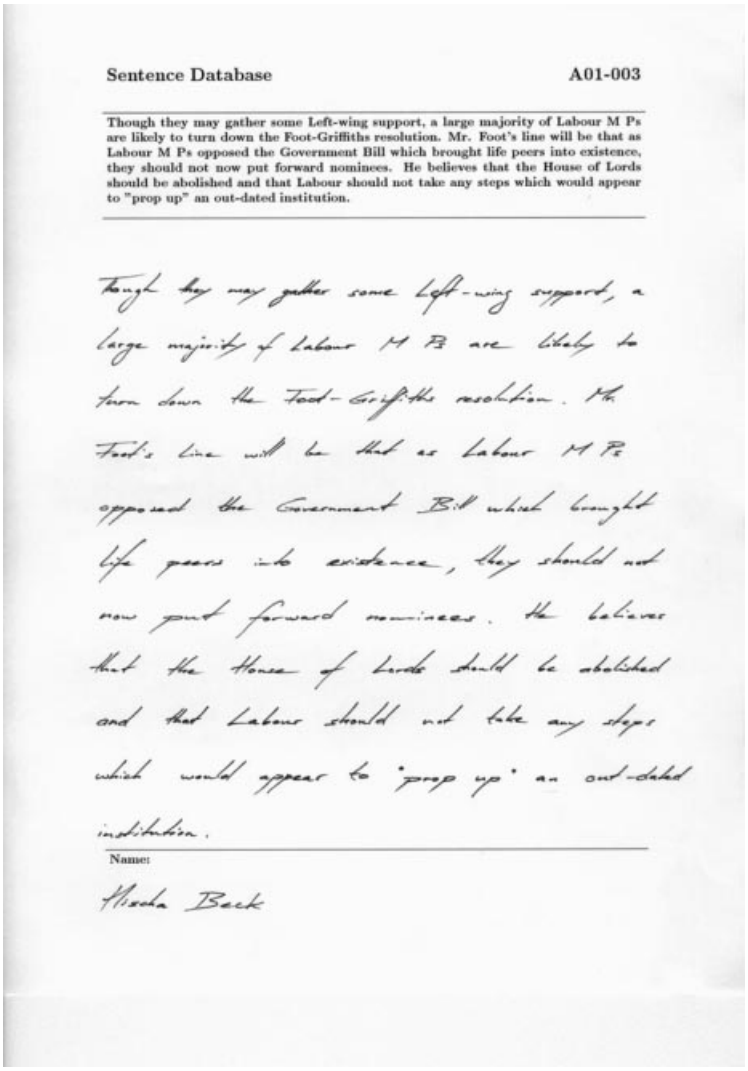

Figure 13: The IAM Handwriting Sample

depends on programmer what prefers.

4.4 Libraries

- GUI (Graphical User Interface) libraries - Swing and Awt (Abstract Windowing Toolkit)
- Jimi (Java Image Management Interface) [77] - is a class library for managing images. Its primary function is image I/O. Jimi's range of supported formats includes GIF, JPEG, TIFF, PNG, PICT, Photoshop, BMP, Targa, ICO, CUR, Sunraster, XBM, XPM, and PCX, although some of these formats do not have complete support for all features.
- ImageJ (Image Processing and Analysis in Java) and its plugins which are

available in [76].

4.5 Application

Firstly, each input image is adapted by preprocessing to form which is suitable for next algorithms. Preprocessing consists of:

- Thresholding
- Line segmentation
- Slant detection and correction
- Word segmentation
- Grapheme segmentation and normalization

The next step depends on amount of input images. Let's describe the following three cases:

- *Two input images* - this is the fundamental problem.
  The handwriting features are extracted from the images. For each image we create feature vector consisting of the set of numbers which represent features. Then a new vector is created by the subtraction of two feature vectors and its coordinates are changed into absolute values. Result is given by the comparation of this new vector and vector of our thresholds (obtained from experiments). The first approach, solving our task, is based on this feature vector.
  Next, graphemes which were obtained from preprocessing, are clustered using two different clustering methods, namely Kohonen's self-organizing map (SOM) and our modification of hierarchical clustering. Result from each clustering determines how much the input images are similar. These results (from each clustering) are compared with unique thresholds obtained from our experiments. We join results from the first approach and results from SOM to the second approach. The third approach is based on results from first approach and results from modified hierarchical clustering. These three approaches are compared experimentally.

- *Three and more input images*
  Given n input images, take first and second image and apply on them the methods from the second case. If the result is that two images aren't written by the same writer, we finish with the same result. Otherwise (two images are written by the same writer), we continue with comparison of the second and third image. We repeat mentioned steps until we get two images which aren't written by the same writer or until we reach the last two images - $n-1$ and n, which are written by the same writer (see table 2).

Input: number of input images (n)
Output: true (images are written by the same writer) or false (images are written by different writers)

```
boolean verifyThreeAndMoreImages (int n) {
    for (int k = 0; k < n; k++) {
        if (verifyTwoImages(k, k+1) ! = true ) return false;
    }
    return true;
}
```

Table 2: Algorithm for the second case (three and more input images). Method *verifyThreeAndMoreImages* returns true if all input images are written by the same writer and return false if aren't. Method *verifyTwoImages* compares two input images and return true or false.

- *One input image*
  The handwriting features are extracted from the image. Each feature is represented by a vector of numbers $F = (n_1, n_2, ..., n_k)$ where k is number of lines of image and $\forall i \in \{1, ..., k\} : n_i \in \mathbf{R}$ and n_i represents feature obtained from ith line. If the whole image is written by the same writer, all coordinates of vector F should be similar, i.e. differences between coordinates $|n_i - n_j|$ for $\forall i, j \in \{1, ..., k\} : i \neq j$ are smaller than a threshold (unique for each feature) obtained from our experiments. Features are described in section 4.7.

4.6 Preprocessing

4.6.1 Thresholding

We implement Otsu's algorithm described in section 3.1.2. This method has been often used for images with handwritten text in recent research (see figure 14).

4.6.2 Line segmentation

The document is divided across its width into chunks, each represents 5%. Then the horizontal projection profile (described in section 2.1.2) and averaging filter are applied for every chunk. The averaging filter smooths HPP. We use 7x7 filter window of type h_1 described in section 2.1.3. Afterwards the image is converted to binary form. At the moment, we have prepared chunks for next processing.
Algorithm finds a global maximum in each chunk. Next, global minimums between neighbouring chunks in vertical direction are defined. Algorithm sequentially traverses and joins obtained global minimums in horizontal direction and the result are color dashes which separate single lines (see figures 15 and 16).

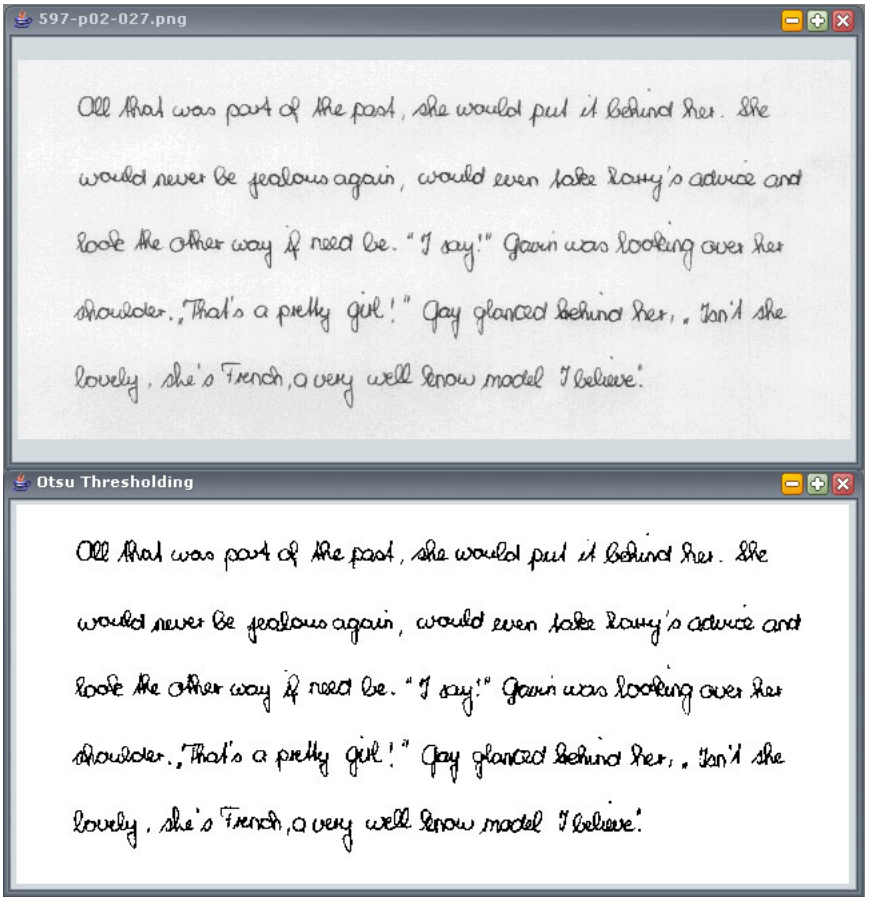

Figure 14: Otsu's thresholding: on upper image is an input greyscale image and on bottom image is the result of Otsu's thresholding - a binary image. Background has white color and foreground (handwriting) has black color. Result is good if background doesn't contain black pixels and foreground doesn't contain white pixels.

We are inspired by the paper [37] but they involve the use of bivariate Gaussian densities to model lines. Their method is robust to handle skewed documents with lines running into each other. In respect of our uncomplicated input images, we implement simpler version of line segmentation.

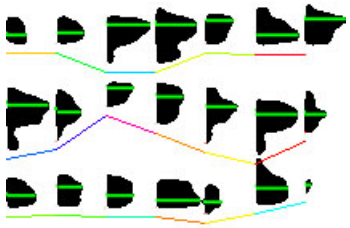

Figure 15: Detail of chunks. Green horizontal lines through chunks are places where maximums were found.

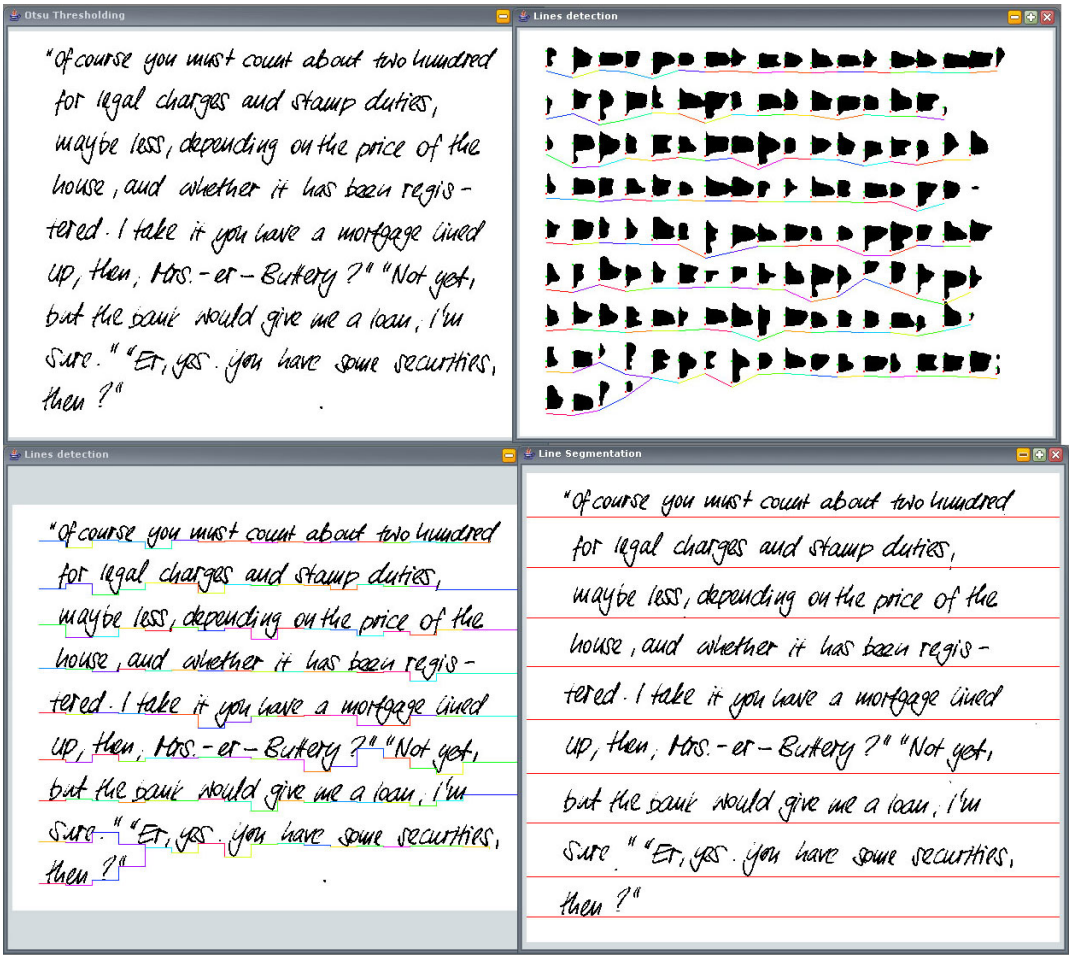

Figure 16: Line Segmentation

4.6.3 Slant detection and correction

Handwriting has tendency to slant to the right or left side or it can be vertical, too. We analyze a set of vertical projection profiles (described in section 2.1.2) for the estimation of the average slant angle like in [51], [52] and [53]. Our algorithm works on lines of handwriting. We determine 20 different slants (angles) between -1.0 rad (approximately -57,295°) and 1.0 rad (approximately 57,295°). For every slant, line is corrected. Then the vertical projection profile is applied. From these 20 tries, one is chosen as the best slant for particular line. The choosing criteria involves the fact that the projection profile has to contain the largest amount of the highest peaks and also maximum gaps (see figure 17). An example of corrected document is shown in figure 18.

4.6.4 Word segmentation

Word segmentation is based on the idea that gaps between words (inter-word gaps) are larger than gaps between characters (inter-character gaps). We takeover

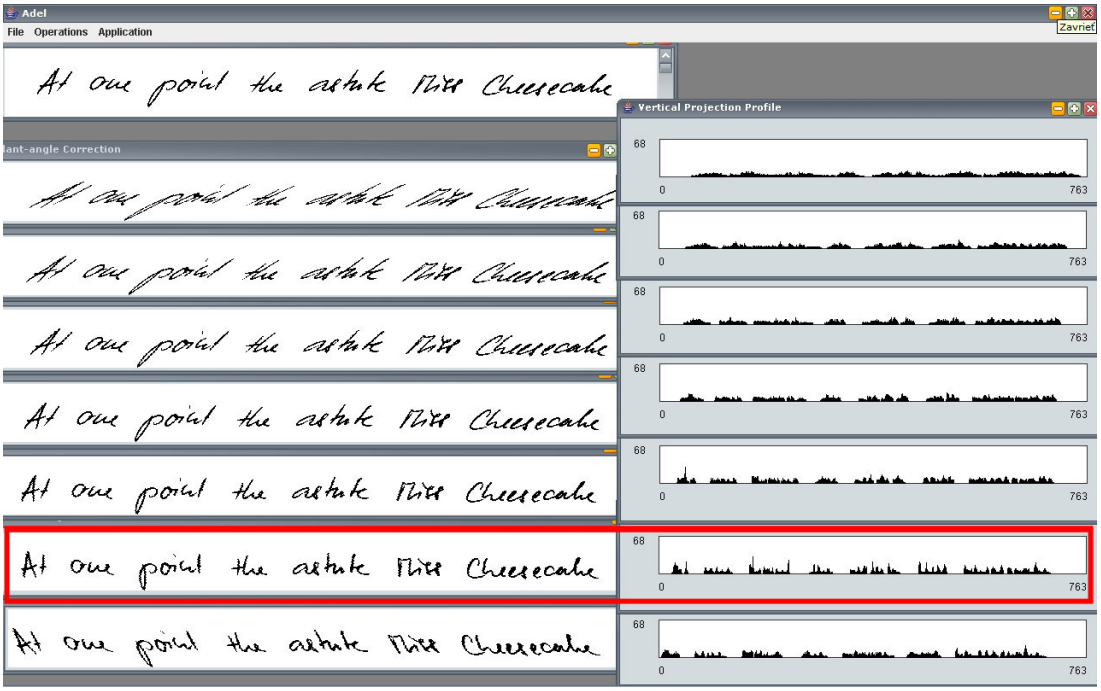

Figure 17: Slant correction: on the top of the image is original line. Under it, corrected lines (only 7 for illustration) are depicted with corresponding VPPs. The chosen segmented line is shown in the red rectangle.

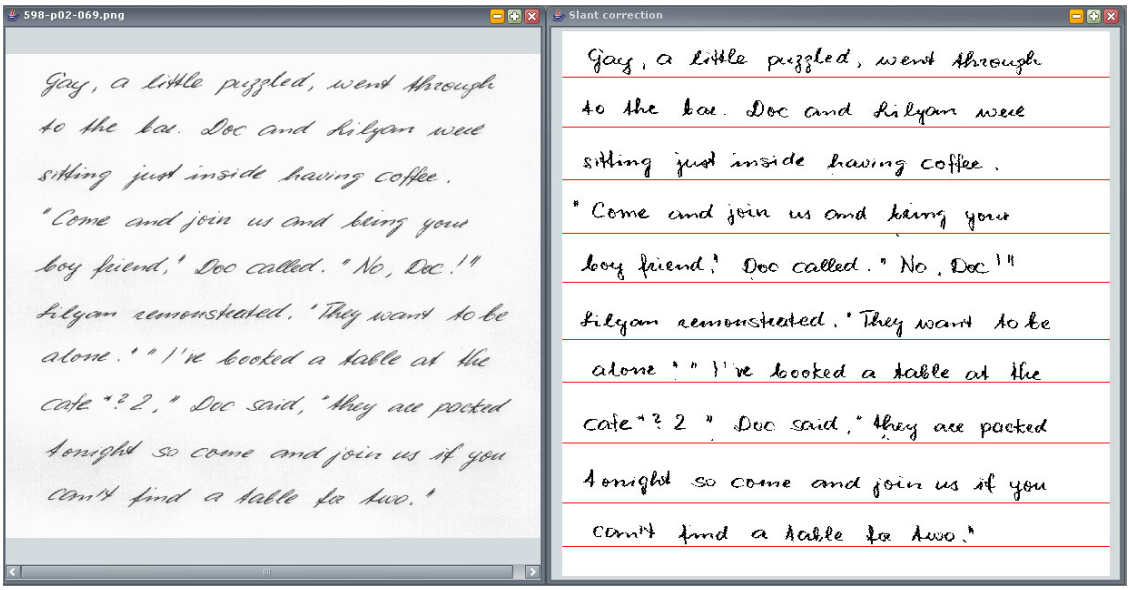

Figure 18: Slant correction: on the left side is input image and on the right side is the result of slant correction.

this idea from [66]. In this paper, they explored eight algorithms that compute the distances between pairs of connected components. One of them is the bounding box method which computes the minimum horizontal distance between the

bounding boxes of the connected components (see figure 19). The distance between bounding boxes that overlap horizontally, is considered to be zero. The bounding box is the smallest rectangle which surrounds the handwritten text.
We extract contours from image (using Moore's algorithm which is described in section 2.1.4) and then we count distances between horizontally neighbouring bounding boxes of contours for every line. If the distance between bounding boxes is larger than threshold (which is obtained by analyzing all distances), we obtain a inter-word gap. An example of segmented words is shown in figure 20.

Figure 19: The bounding boxes.

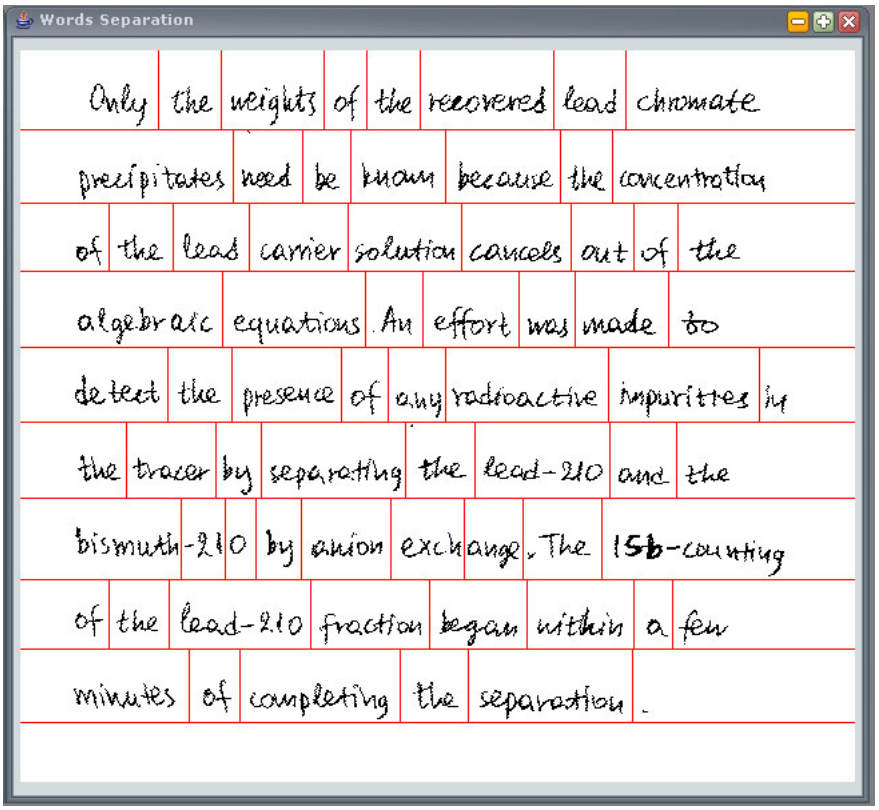

Figure 20: Word segmentation.

4.6.5 Grapheme segmentation and normalization

We takeover this method from [7]. Lower and upper contours are extracted from the image (see figure 21). Then we find local minimums which occur in places

where the upper contour is close to the lower contour. It means that the distance between lower and upper contour is the width of line of handwriting. Words are segmented into graphemes (grapheme may or may not overlap a complete character) at these local minimums by vertical lines (see figure 22). After segmentation, graphemes are normalized to 30x30 pixels.
The essential idea is that the ensemble of these simple graphemes still manages to capture the shape details of the allographs emitted by the writer [7].

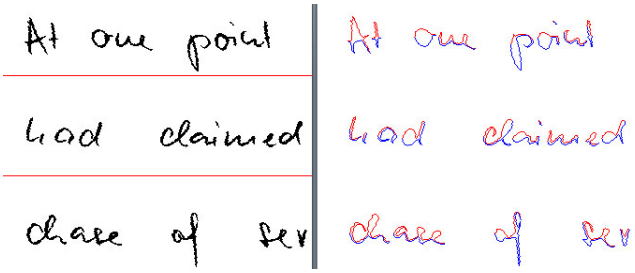

Figure 21: Lower and upper contours: on the left side is the original image and on the right side are extracted contours. Upper contours are illustrated by red curves and lower contours by blue curves.

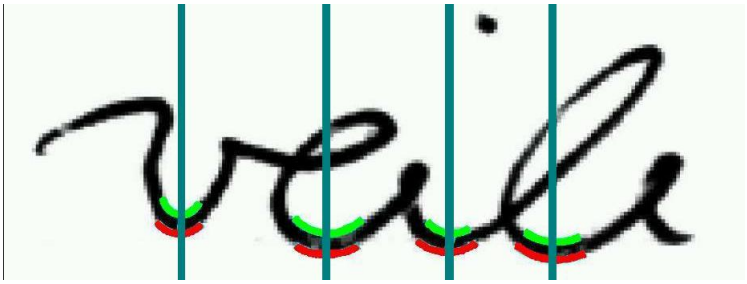

Figure 22: Grapheme segmentation: upper contours are illustrated with green curves and lower contours with red curves. Word 'veile' is divided into graphemes by vertical blue lines. Original image comes from [7].

4.7 Extraction of features

Individuality of handwriting is based on the idea that the variation within a person's handwriting is less than the variation between the handwriting of two different people. We use the following features:

- Proportion of handwriting
- Height of handwriting and distance between lines
- Slant
- Density

- Block letters

Each feature is described in more detail.

4.7.1 Proportion of handwriting

Four baselines are computed for every line (see figure 23):

- Ascender line passes through the topmost point of the line (Asc)
- Descender line passes through the bottom point of the line (Des)
- Upper baseline goes through the top of the lower case characters (Upp)
- Lower baseline goes through the bottom of the lower case characters (Low)

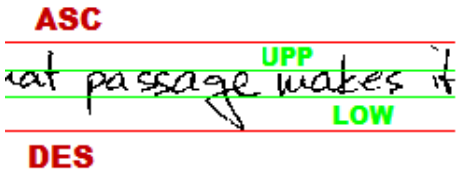

Figure 23: Four baselines

Proportion is defined as following ratio:

$$(Upp - Asc) : (Low - Upp) : (Des - Low) \tag{7}$$

Therefore we obtain three numbers for each line:

$$\begin{array}{l}(Upp - Asc) : (Low - Upp)\\(Low - Upp) : (Des - Low)\\(Upp - Asc) : (Des - Low)\end{array} \tag{8}$$

Children in the elementary schools are taught to write in a way that yields a 1:1:1 proportion. But later on, when each of us forms his/hers writing, the proportion is changed and that is characteristic for our handwriting. Baselines are a basic feature in handwriting and more of them can be found in [78].

4.7.2 Height of handwriting and distance between lines (DBL)

Height of handwriting is computed for every line as (Des - Asc). Distances between lines are obtained by line segmentation. Because we don't want to obtain result in pixels, we compute ratio of distance between lines to height of handwriting for each couple (see figure 24).

4.7.3 Slant

Slant angle is obtained for each line by slant detection which is part of preprocessing.

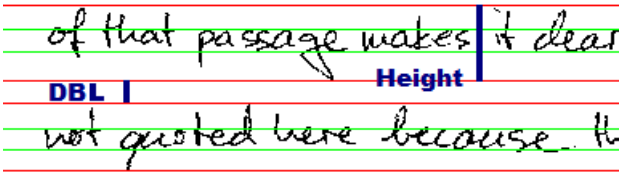

Figure 24: Image figures distance between lines (DBL) and height of handwriting.

4.7.4 Density

We compute number of black pixels for each line.

4.7.5 Block letters

Nowadays humans often use block letters because of its readibility. They sometimes mix both types, block letters and court hand. We define so-called 'measure of using block letters'. This measure is determined by the next criteria: the more gaps are in the words the more block letters writer use (see figure 25). Measure is obtained from each line of handwriting.

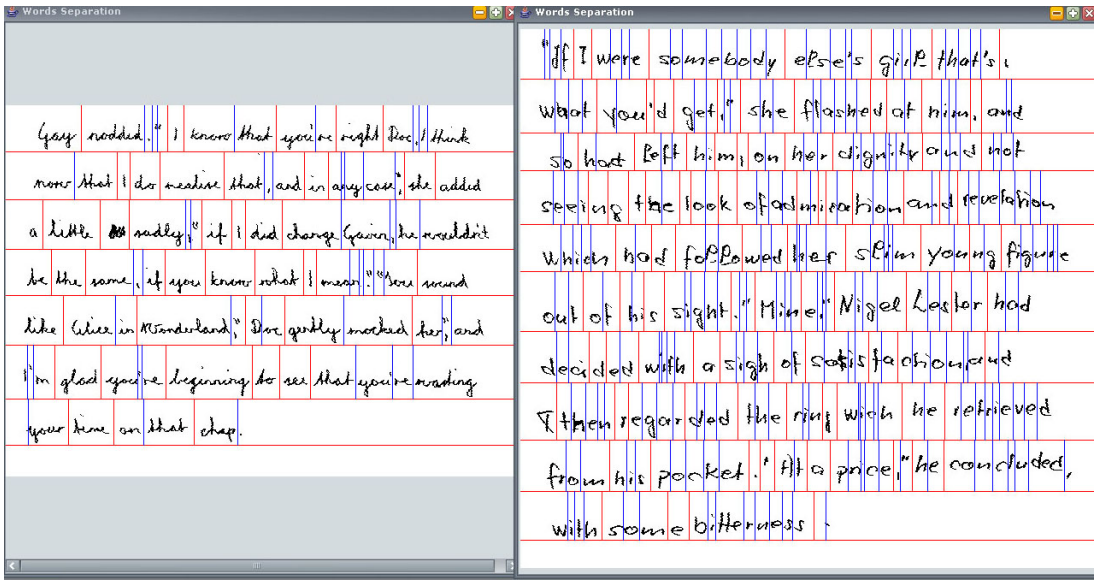

Figure 25: Block letters. On the left side is an example of handwriting close to court hand and on the right side close to block letters. Court hand has characters in words mostly connected contrary to block letters where characters mostly stand alone. Red vertical lines divide each line into words and blue vertical lines are depicted in places where occur gaps in word.

4.7.6 Additional features

Design of our system enables to add another features or to change actual features (e.g. distances between words).

4.8 Feature vector

Each feature (which characterises the document with handwritten text) is represented by a number (apart from proportion of handwriting which has 3 numbers) and is computed as an average of features obtained from segmented lines. So, for n features we get a vector of numbers.
We have two images as an input. For every image a feature vector is computed, therefore we have two feature vectors:

$$F = (f_1, ..., f_n), G = (g_1, ..., g_n) \tag{9}$$

Then a new vector is created by the subtraction of two mentioned vectors and its coordinates are changed into absolute values:

$$H = (\mid f_1 - g_1 \mid, ..., \mid f_n - g_n \mid) \tag{10}$$

A threshold vector is obtained from our experiments. We have unique threshold

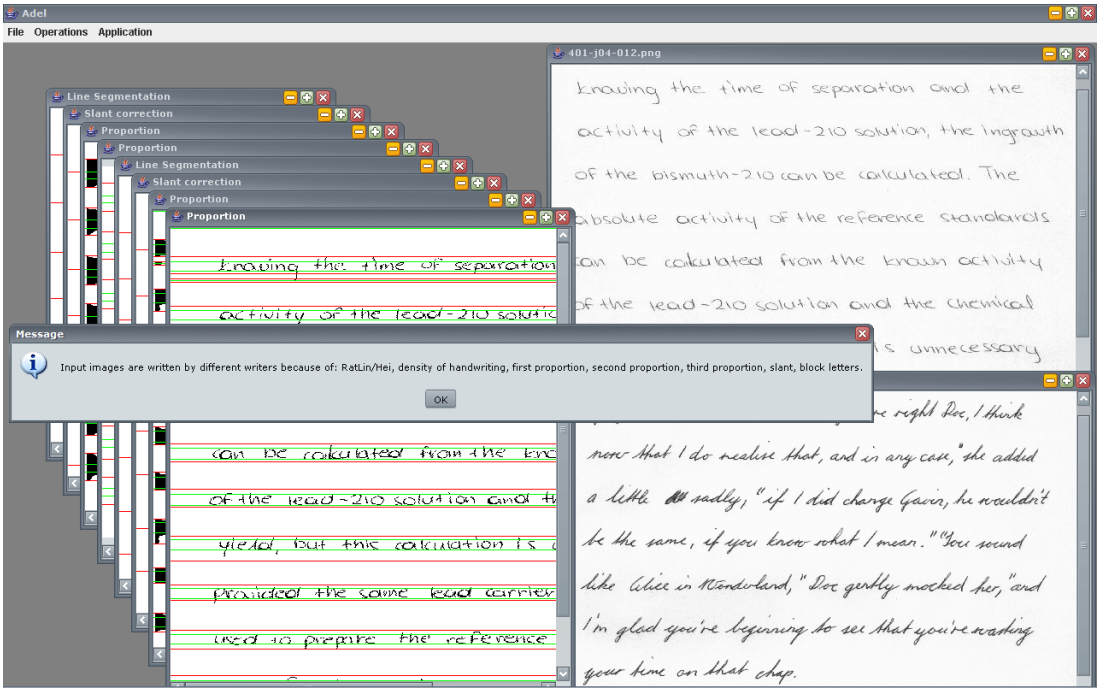

Figure 26: Result of the first approach. On the right side, two input images from different writers are depicted. After running the first approach, application gives message which notifies that input images are written by different writers. The message also writes features which aren't similar for input images.

for each feature (proportion 10, 10, 10, ratio of distance between lines to height 0.5, slant 0.2, density 300 and finally block letters 0.6):

$$T = (t_1, ..., t_n) \tag{11}$$

Then each coordinate of vector H (h_i) is compared with corresponding coordinate of threshold vector (t_i). If h_i is smaller than t_i, it's a sign of similarity, if it is bigger, it's a sign of dissimilarity. If we have at least one coordinate of vector H bigger than corresponding coordinate of threshold vector, two input images are marked as written by the different writers. Otherwise the images are marked as written by the same writer. It means that the only one dissimilar feature is enough to determine different writers:

$$\begin{aligned} &\text{if } \exists i \in \{1, ..., n\} : h_i > t_i \text{ then different writers} \\ &\text{otherwise same writer} \end{aligned} \tag{12}$$

4.8.1 First approach

Our first approach is based on comparing two feature vectors as described above. An example of application of first approach is shown in figure 26.

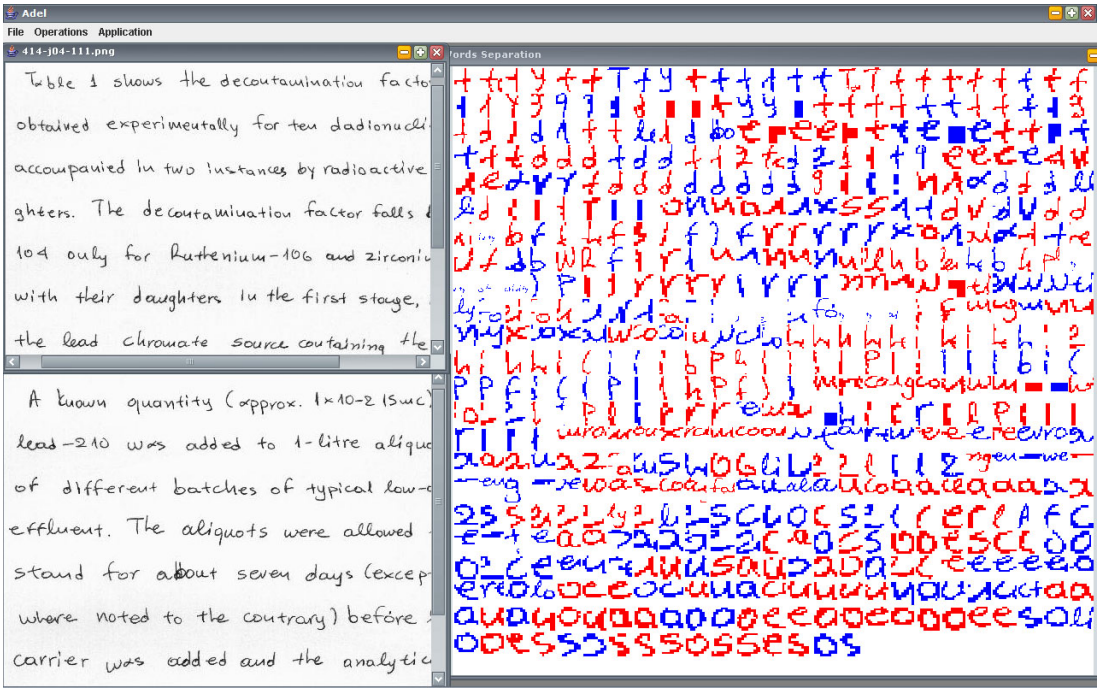

Figure 27: The result of SOM on two input images written by the same writer. Input images are shown on the left side and result from SOM on the right side. The red graphemes come from grapheme segmentation from the first input image and the blue graphemes come from the second input image. SOM groups the similar graphemes to the same cluster. All clusters are placed abreast on the image. You can see that SOM places similar graphemes abreast.

4.9 Graphemes clustering

The features mentioned in section 4.7 represent only a half of the whole solution. An important role is played by the individual graphemes and their clustering. Graphemes are segmented from each image and clustered afterwards.

The second approach

Inspired with [7], we use Kohonen's self-organizing map (SOM). SOM groups graphemes from both images according to the similarity of graphemes into the clusters. Similarity of two graphemes is determined by their Euclidean distance (see figure 27). We look at each cluster and find out whether it contains graphemes from both images or only from one image. The more clusters with 'mixed' graphemes exist, the more similar images are. The second approach is based on results from first approach and results from SOM.

The third approach

The second clustering method which we use is our modification of hierarchical clustering. As the input we use graphemes from both images. First step is finding for each grapheme the 'closest' grapheme (using Euclidean distance). Hence we have grapheme pairs (clusters). Second step is finding for each cluster the closest cluster (analogue with first step) using average linkage (see figure 7). The result is realized by percentage of clusters which contain graphemes coming from the same image. The third approach is based on results from first approach and results from modified hierarchical clustering.

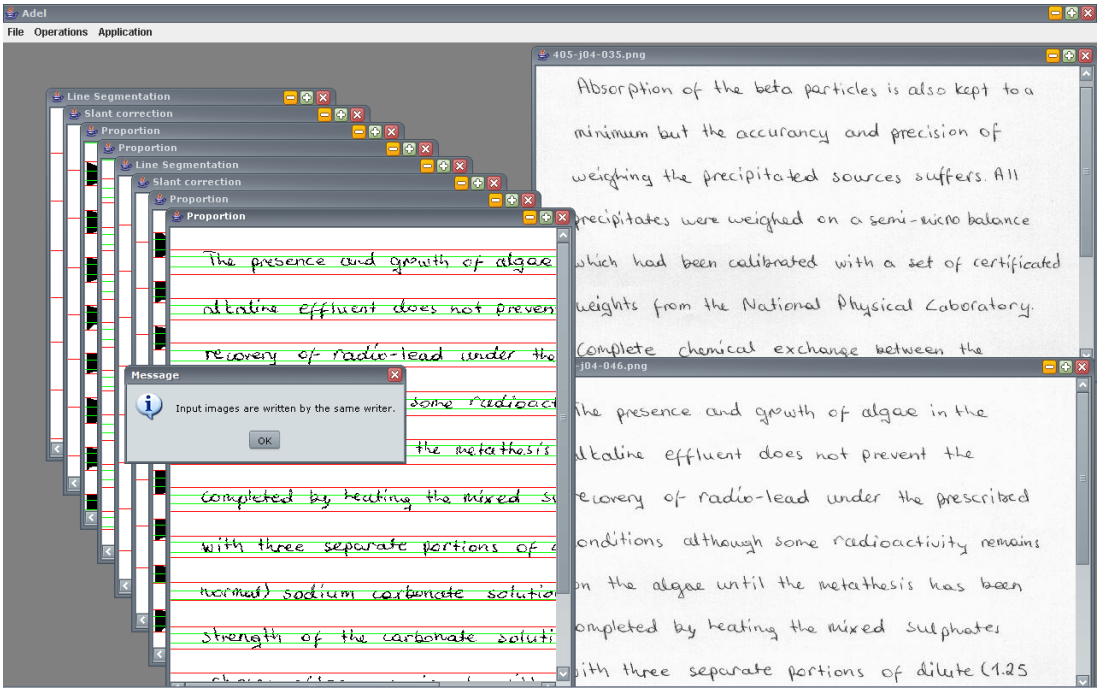

Figure 28: Result of the third approach. Two input images from the same writer are depicted on the right side. After running the third approach, application gives message which notifies that input images are written by the same writer.

Input: two input images i_1, i_2
Output: true (images are written by the same writer) or false (image are written by different writers)

```
boolean secondApproach (i1, i2) {
    double som = SelfOrganisingMap(i1, i2);
    if (som < 55) return false;
    else if (som > 70) return true;
    else return firstAproach(i1, i2);
}

boolean thirdApproach (i1, i2) {
    double[] cluster = ModifiedHierarCluster(i1, i2);
    if (cluster[0] < 55) return true
    else if (cluster[0] > 65) return false
    else {
       if (cluster[1] < 20) return true
       else if (cluster[1] > 30) return false
       else return firstAproach(i1, i2);
    }
}
```

Table 3: A formal description of the second and third approch.

Result from each approach (second and third) is a measure which determines how input images are similar. This measure is compared with two thresholds which are unique for each approach (we got them experimentally, for SOM the thresholds are 55 and 70 and for modified hierarchical clustering 55 and 65 after the first step and 20 and 30 after the second step).
If the result from SOM (modified hierarchical clustering) is smaller than the first threshold, it's a sign of dissimilarity (similarity), if it is bigger than the second threshold, it's a sign of similarity (dissimilarity) and if it fits between first and second threshold, it's uncertain and we rely on another method. Therefore SOM resp. modified hierarchical clustering is combined with the first approach (which is based on features). If measure fits between first and second threshold, we take the result from the first approach. Otherwise, result from SOM resp. modified hierarchical clustering is taken. Algorithm is described formally in table 3.
An example of application of third approach is shown in figure 28.

5 Implementation and results

5.1 Results

5.1.1 Experiments

We performed our experiments on 100 images from 40 different writers. We made 100 experiments on 2 different images from the same writer and 100 experiments on 2 images from 2 different writers. We tested 3 mentioned approaches (see table 4).

Approach	writer	NOE	NOCR	NOIR	accuracy	average
1.	same	100	85	15	85%	91%
	different	100	97	3	97%	
2.	same	100	86	14	86%	92%
	different	100	98	2	98%	
3.	same	100	93	7	93%	96,5%
	different	100	100	0	100%	

Table 4: Results of our experiments: signification of shortcuts - NOE (number of experiments), NOCR (number of correct results), NOIR (number of incorrect results). As seen in the table, the best results gives the third approach. The first approach gives good results in respect of its complexity. However the second approach doesn't bring expected enhancement.

5.1.2 Sources of errors

Results strongly depend on preprocessing. In some cases preprocessing can give incorrect output. For example line segmentation fails if lines are too close and touch each other. Also word segmentation has problem with image where gaps between words and gaps between characters are similar. If the slant has more directions in the line, for example at the beginning of the line handwriting has tendency to slant to left and at the end to right, it can be problem for our algorithm. Now we explain source of errors for each approach:

1. the first approach
   errors in experiments on images from 2 different writers: images were marked as written by the same writer because it didn't find any dissimilarity. It is not available to detect differences. We can solve it by adding another features.
   errors in experiments on images from 2 same writers: this errors are more serious because of their quantity. It can be caused by choosing an unsuitable threshold. Another problem is failure of some algorithms (slant detection, proportion of handwriting) which give bad result for particular input.

2. the second and third approach
   SOM and hierarchical clustering tend to cluster some particular graphemes

that are not significantly similar with respect to the goals of our effort. Errors can be also caused by choosing two unsuitable thresholds. When result of SOM resp. modified hierarchical clustering fits between these thresholds, approaches take the results from the first approach. So in this case the second resp. third approach takes over errors from the first approach. We have no errors in experiments on images from 2 different writers in the third approach, but we should do more experiments to detect defection of the approach.

5.1.3 Performance

We made experiments using Intel Core 2 CPU 1,66 GHz with 1 GB RAM. Average elapsed times: preprocessing 5 seconds, extraction of features less than 1 second, modified hierarchical clustering 14 seconds and SOM 52 seconds. Average elapsed times of three approaches is shown in table 5 and compared in respect of average accuracy.

Approach	**average elapsed time**	**average accuracy**
1.	13 sec	91%
2.	67 sec	92%
3.	27 sec	96,5%

Table 5: Elapsed time vs. accuracy: Experiments show that the second approach (with SOM) computes 5 times slower than the first and it brings only 1 % enhancement. Therefore it seems to be the least appropriate. The third approach computes 2 times slower than the first and it brings 5,5 % enhancement. So this is our winning approach.

6 Conclusion

We have presented a combination of particular steps that creates a workflow for identification and decision whether one and more documents are written by the same author or not. Our effort was concentrated on a comparison of two images as on a fundamental problem. We proposed three approaches consisting of preprocessing, feature vector extraction and combination with graphemes clustering. We brought a voluminous review of preprocessing and writer verication and presented implementation of mentioned task. Our experiments were made on 100 samples from the IAM handwriting Database. We tested 3 approaches described above and the best result was achieved by the third approach (96,5 % accuracy). The presented results show the efficiency of our method. Our application is user-friendly with simple control and minimization of interaction. No special knowledge is expected for control our application.

7 Future work

Our future work is to bring a better preprocessing (deskew document, noise reduction, deskew lines, rule line removal), include more handwriting features, accelerate our approaches, implement another clustering techniques, compare the results for different thresholds in each approach and make more experiments, also with another handwriting databases.

8 List of Symbols and Abbreviations

SOM	Self-Organizing Map
HPP	Horizontal projection profile
VPP	Vertical projection profile
OCR	Optical Character Recognition
DBL	Distance between lines

9 Glossary

Allograph	a variant shape of a letter; i.e. one particular letter from an alphabet can be realized using a number of shapes.
Grapheme	(sub or supra-allographic fragments) may or may not overlap a complete character, it's the smallest meaningful unit of a written language.
Connected component	is a set of black pixels, P, such that for every pair of pixels p_i and p_j in P, there exists a sequence of pixels $p_i, ..., p_j$ such that: 1. all pixels in the sequence are in the set P i.e. are black, and 2. every 2 pixels that are adjacent in the sequence are "neighbors" language.

10 Bibliography

References

[1] Jan Jeřábek. 2003. *Grafologie*. Vydavateľstvo Argo. 2003. ISBN 807203524X.

[2] František Stritz. 1996. *Čo nám hovorí písmo?; Grafológia pre každého*. Nitra : Enigma, 1996. ISBN 80-85471-25-6 ISBN 978-80-85471-25-0.

[3] Sung-Hyuk Cha. 2001. *Use of distance measures in handwriting analysis*. Buffalo, N.Y. : Dept. of Computer Science, State University of New York at Buffalo. 2001.

[4] Réjean Plamondon and Sargur N. Srihari. 2000. *On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1. 2000.

[5] R.Manmatha and Toni M. Rath. 2003. *Indexing of Handwritten Historical Documents - Recent Progress*. In: Proc. of the 2003 Symposium on Document Image Understanding Technology (SDIUT), Greenbelt, MD. 2003.

[6] Yu Qiao and Jianzhuang Liu and Xiaoou Tang. 2007. *Offline Signature Verification Using Online Handwriting Registration*. Computer Vision and Pattern Recognition, 2007.

[7] Marius Bulacu and Lambert Schomaker. 2007 *Text-Independent Writer Identification and Verification Using Textural and Allographic Features*. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2007.

[8] The British Academy of Graphology. *www.graphology.co.uk/index02.html*

[9] LearnGraphology.com. *www.learngraphology.com/graphology.html*

[10] The London College of Graphology. *www.collegeofgraphology.co.uk/graphology1.htm*

[11] Lukáš Čechovič. *Neurónové siete pri rozpoznávaní písma*. http://frtk.fri.utc.sk/ cechovic/dajzdrojak.pdf.

[12] Y. Chen and G. Leedham. 2005. *Decompose algorithm for thresholding degraded historical document images*. School of Computer Engineering, Nanyang Technological University, Singapore. 2005

[13] Yan Solihin and C.G. Leedham. 2000. *The Multi-stage Approach to Grey-Scale Image Thresholding for Specific Applications*. Nanyang Technological University, School of Computer Engineering, Nanyang Avenue, Republic of Singapore. 2000.

[14] Interactive handbook of Image Processing
http://dip.sccg.sk

[15] H. Said, G. Peake, T. Tan, and K. Baker. 1998. *Writer identification from non-uniformly skewed handwriting images.* In Proc. of the 9th British Machine Vision Conference, pages 478–487. 1998.

[16] Contour tracing
www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/index.html

[17] A Tutorial on Clustering Algorithms.
http://home.dei.polimi.it/matteucc/Clustering/tutorial_html

[18] Matthias Schonlau. *Visualizing non-hierarchical and hierarchical cluster analyses with clustergrams.* RAND 1700 Main Street Santa Monica, CA 90407 USA.

[19] Hierarchical and non-Hierarchical Clustering.
www.daylight.com/meetings/mug96/barnard/E-MUG95.html

[20] Danka Némethová. *Zhluková analýza* Inštitút bioštatistiky a analýz, Masarykova univerzita.
http://is.muni.cz/el/1431/podzim2007/Bi8600/Zhlukovky_teoria.pdf?fakulta=1431;obdobi=3843;kod=Bi8600

[21] Šárka Došlá, Michal Juráška, Matúš Maciak, Jaroslav Ševcík. *Shluková analýza* Univerzita Karlova v Praze Mnohorozmerná statistická analýza. 2006.

[22] Melody Y. Kiang. 2001. *Extending the Kohonen self-organizing map networks for clustering analysis* Computational Statistics Data Analysis 38 161-180. 2001.

[23] Kvasnička V., Beňušková., Pospíchal J., Farkaš I., Tiňo P. a Kráľ A.. 1997. *Úvod do teórie neurónových sietí* Iris: Bratislava. 1997.

[24] Mehmet Sezgin, Bulent Sankur. 2004. *Survey over image thresholding techniques and quantitative performance evaluation.* Journal of Electronic Imaging 13(1), 146–165 (January 2004). 2004.

[25] N. Otsu. 1979. *A threshold selection method from gray level histograms.* IEEE Trans. Systems, Man and Cybernetics. 1979.

[26] V.V. Nabíyev, C. Köse, S. Bayrak. 2006. *An artificial neural network approach for sign language vowels recognition.* Department of Computer Engineering, Faculty of Engineering, Karadeniz Technical University. 2006.

[27] T. Pun. 1981. *Entropic Thresholding: A New Approach*. CVGIP: Graphical Models and Image Processing 16, pp. 210–239, July 1981.

[28] Kapur, Sahoo and Wong. 1985. *A new method for Gray-level picture thresholding using the entropy of the histogram*. Computer Vision, Graphics, and Image Processing, vol 29, pp 273-285, 1985.

[29] J. Bernsen. 1986. *Dynamic thresholding of grey-level images*. in Proc. Eighth Int'l Conf. on Pattern Recognition, Paris, France, pp. 1251-1255, Oct. 1986.

[30] Graham Leedham, Chen Yan, Kalyan Takru, Joie Hadi Nata Tan and Li Mian. 2003. *Comparison of Some Thresholding Algorithms for Text/Background Segmentation in Difficult Document Images*. School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore. Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR 2003). 2003.

[31] W. Niblack. 1986. *An Introduction to Digital Image Processing*. pp. 115-116, Prentice Hall, 1986.

[32] S.D. Yanowitz and A.M. Bruckstein. 1989. *A new method for image segmentation*. Computer Vision, Graphics and Image Processing, vol. 46, no. 1, pp. 82-95, Apr. 1989.

[33] Wang L. and Pavlidis T.. 1993. *Direct Gray-Scale Extraction of Features for Character Recognition*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 15, No. 10, Oct. 1993, pps. 1053-1067.

[34] Brink A.D.. 1995. *Minimum spatial entropy threshold selection*. IEE Proceedings-Vision Image and Signal Processing. v142. 128-132. 1995.

[35] Y.Solihin, C.G.Leedham. 1999. *Integral Ratio: A New class of Global Thresholding Techniques for Handwriting Images*. IEEE Trans. PAMI, vol.21, no. 8, pp. 761-768, August 1999.

[36] Z. Zhang and C. L. Tan. 2001. *Restoration of images scanned from thick bound documents*. Proc. Int. conf. Image Processing., vol. 1, 2001, pp.1074-1077.

[37] M. Arivazhagan, H. Srinivasan and S. Srihari. *A Statistical approach to line segmentation in handwritten documents*. Center of Excellence for Document Analysis and Recognition (CEDAR) University at Bualo, State University of New York. 2007.

[38] B. Yanikoglu, P. A. Sandon. 1998. *Segmentation of off-line cursive handwriting using linear programming*. Pattern Recognition 31, pp. 1825 1833, 1998.

[39] E. Kavallieratou, N. Dromazou, N. Fakotakis, G. K. Kokkinakis. 2003. *An integrated system for handwritten document image processing*. IJPRAI, International Journal of Pattern Recognition and Articial Intelligence 17(4), pp. 101120, 2003.

[40] Weliwitage C., Harvey A.L. and Jennings, B.. 2005. *Handwritten document offline text line segmentation, Digital Image Computing: Techniques and Applications*. Digital Image Computing: Techniques and Applications, DICTA , pp. 184 187, December 2005.

[41] Markus Feldbach and Klaus D. Tönnies. 2001. *Line Detection and Segmentation in Historical Church Registers*. Sixth International Conference on Document Analysis and Recognition, , Seattle, USA, IEEE Computer Society , pp. 743747, September 2001.

[42] Jisheng Liang, Ihsin T. Phillips, Robert M. Haralick. 1999. *A statistically based, highly accurate text line segmentation method*. Proceedings 5th ICDAR , pp. 551-554, 1999.

[43] Douglas J. Kennard, William A. Barrett. 2006. *Separating Lines of Text in Free-Form Handwritten Historical Documents*. Second International Conference on Document Image Analysis for Libraries(DIAL) , pp. 1223, April 2006.

[44] Stephane Nicolas, Thierry Paquet, Laurent Heutte. 2004. *Text Line Segmentation in Handwritten Document Using a Production System*. Ninth International Workshop on Frontiers in Handwriting Recognition, IWFHR-9 26-29, pp. 245 250, October 2004.

[45] S.-H. Cha and S. N. Srihari. 2000. *System that identies writers*. Proceedings National Conference of American Association of Articial Intelligence, Austin, TX , p. pp. 1068, July 2000.

[46] S. N. Srihari, B. Zhang, C. Tomai, S. Lee, and Y.-C. Shin. 2000. *A system for handwriting matching and recognition*. Proc. Symp. Document Image Understanding Technology Greenbelt, MD , pp. pp. 6775, April 2003.

[47] Seiichi Uchida, Eiji Taira, and Hiroaki Sakoe. 2001. *Nonuniform Slant Correction Using Dynamic Programming*. icdar, p. 0434, Sixth International Conference on Document Analysis and Recognition (ICDAR'01). 2001.

[48] Jian-xiong Dong, Ponson Dominique, Adam Krzyyzak, Ching Y. Suen. 2005. *Cursive word skew/slant corrections based on Radon transform*. icdar, pp. 478-483, Eighth International Conference on Document Analysis and Recognition (ICDAR'05). 2005.

[49] R.M. Bozinovic and S.N. Srihari. 1989. *Off-Line Cursive Script Word Recognition*. IEEE Trans. PAMI, Vol. 11, No. 1, pp. 68–83, Jan. 1989.

[50] G. Kim and V. Govindaraju. 1997. *A lexicon driven approach to handwritten word recognition for real-time applications*. IEEE Trans. PAMI, Vol. 19, No. 4, pp.366-379, April 1997.

[51] D. Guillevic and C.Y. Suen. 1994. *Cursive Script Recognition: A Sentence Level Recognition Scheme*. Proc. 4th IWFHR, pp. 216–223, Dec.1994.

[52] E. Kavallieratou, N. Fakotakis and G. Kokkinakis. 2000. *A slant removal algorithm*. Patt. Recog., Vol. 33, No. 7, pp. 1261–1262, Jul. 2000.

[53] G. Nicchiotti and C. Scagliola. 1997. *Generalised projections: a tool for cursive handwriting normalisation*. Proc. 5th ICDAR, pp.729-732, Sep.1997.

[54] F. Kimura, M. Shridhar, and Z. Chen. 1993. *Improvements of a lexicon directed algorithm for recognition of unconstrained handwritten words*. Proc. 2nd ICDAR., pp. 18–22, Oct. 1993.

[55] L. Simoncini and Zs.M. Kovács-V. 1995. *A System for Reading USA Census'90 Hand-Written Fields*. Proc. 3th ICDAR, Vol. II, pp.86–91, Aug.1995.

[56] Y. Ding, F. Kimura, Y. Miyake and M. Shridhar. 2000. *Accuracy Improvement of Slant Estimation for Handwritten Words*. Proc. 15th ICPR., Vol. 4, pp.527–530, Sep. 2000.

[57] A. D. S.Britto JR., R. Sabourin, E. Lethelier, F. Bortolozzi, and C.Y. Suen. 2000. *Improvement in handwritten numeral string recognition by slant normalization and contextual information*. Proc. 7th IWFHR., pp.323–332, Sep. 2000.

[58] A. Vinciarelli and J. Luettin. 2001. *A new normalization technique for cursive handwritten words*. Pattern Recognition Letters, vol. 22, no. 9, pp. 1043–1050, 2001.

[59] E. Taira, S. Uchida, H. Sakoe. 2004. *Nonuniform Slant Correction for Handwritten Word Recognition*. IEICE Trans. Inf. & Syst., Vol. E87-D, no. 5. May 2004.

[60] H. Hase, M. Yoneda, T. Shinokawa and C.Y. Suen. 2001. *Alignment of free layout color texts for character recognition*. Proc. 6th ICDAR, pp. 932-936, Seattle, Washington, Sept. 2001.

[61] C. Huang and S. N. Srihari. 2008. *Word Segmentation of Off-line Handwritten Documents*. Proc. Document Recognition and Retrieval(DRR) XV, IST/SPIE Annual Symposium, San Jose, CA, January 2008.

[62] M. Feldbach and K. D. Tonnies. 2003. *Word segmentation of handwritten dates in historical documents by combining semantic a-priori-knowledge with local features*. Proc. of the 7th Int. Conference on Document Analysis and Pattern Recognition, 2003.

[63] U. V. Marti and H. Bunke. 2001. *Text line segmentation and word recognition in a system for general writer independent handwriting recognition*. Proc. of the 6th Int. Conference on Document Analysis and Pattern Recognition ,pp.159–163, 2001.

[64] R. Manmatha and J.L. Rothfeder. 2005. *A scale space approach for automatically segmenting words from historical handwritten documents*. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8), pp.1212–1225, August 2005.

[65] U. Mahadevan and R. C. Nagabushnam. 1995. *Gap Metrics for Word Separation in Handwritten Lines*. Proceedings of the Third International Conference on Document Analysis and Recognition, pp. 124-127, Montreal, (ICDAR 1995). Aug. 1995.

[66] G. Seni and E. Cohen. 1994. *External word segmentation of off-line handwritten text lines*. Pattern Recognition, 27:41-52, 1994.

[67] S. Srihari and G. Kim. Penman. 1997. *A system for reading unconstrained handwritten page images*. Symposium on document image understanding technology (SDIUT 97), pages 142-153, April 1997.

[68] G. Kim and V. Govindaraju. 1998. *Handwritten Phrase Recognition as Applied to Street Name Images*. Pattern Recognition, vol. 31, no. 1, pp. 41-51, 1998.

[69] Manmatha, R. and Srimal, N.. 1999. *Scale space technique for word segmentation in handwritten manuscripts*. the Proceedings of the Second International Conference on Scale-Space Theories Computer Vision (Scale Space 99), pp. 22-33. 1999.

[70] Sung-Hyuk Cha and Sargur N. Srihari. *Multiple feature integration for writer verification*. Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition. 2000.

[71] A. Bensefia, T. Paquet, L. Heutte. *Grapheme Based Writer Verification*. In 11th Conference of the International Graphonomics Society, IGS'2003, Scottsdale, Arizona. 2003.

[72] Andreas Schlapbach and Horst Bunke. *Using HMM Based Recognizers for Writer Identification and Verification*. Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition. 2004.

[73] Sargur N. Srihari et al.. *A Statistical Model For Writer Verification.* Proceedings of the Eighth International Conference on Document Analysis and Recognition. 2005.

[74] CedarTech. *www.cedartech.com/products_cedarfox.html*

[75] IAM Handwriting Database *http://iamwww.unibe.ch/ fkiwww/iamDB*

[76] ImageJ *http://rsb.info.nih.gov/ij*

[77] Jimi *http://java.sun.com/products/jimi/*

[78] Brijesh Verma. 2003. *A Contour Code Feature Based Segmentation For Handwriting Recognition.* Proceedings of the Seventh International Conference on Document Analysis and Recognition. 2003.

11 Abstrakt

Názov práce: Porovnávanie ručne písaných textov
Autor: Miroslava Božeková
Katedra: Katedra aplikovanej informatiky
Vedúci diplomovej práce: Doc. RNDr. Milan Ftáčnik, CSc.

Abstrakt: Cieľom tejto práce je prispieť k riešeniu nasledujúceho problému. Ako vstup máme jeden alebo viac naskenovaných obrázkov s ručne písaným textom a našou úlohou je vytvoriť metódy ktoré určia, či obrázky sú napísané tou istou osobou alebo nie. Sústreďujeme sa na základný problém - porovnávanie dvoch obrázkov a rozhodnutie či sú dva obrázky napísané rovnakou osobou alebo nie - takzvaná verifikácia autora. Predstavujeme tri prístupy pozostávajúce z predspracovania a extrakcie príznakového vektora v spojení so zhlukovaním grafém. Prvý prístup je založený na príznakovom vektore, druhý je kombináciou prvého prístupu a Kohonenových samoorganizujúch sa máp a nakoniec tretí spája prvý prístup a modifikované hierarchické zhlukovanie. Urobili sme experimenty na 100 obrázkoch od 40 rôznych autorov. Používame obrázky z databázy IAM Handwriting Database, ktorá obsahuje 1539 naskenovaných strán s textom od 657 autorov. V týchto experimentoch sme testovali tri spomenuté prístupy na dvoch vstupných obrázkoch. Pomocou tretieho prístupu sme dosiahli 96,5 % úspešnosť. V tejto práci predstavujeme našu implementáciu a taktiež výsledky.

Kľúčové slová: rukopis, verifikácia autora, graféma, zhlukovanie