A Complete Optical Character Recognition Methodology for Historical Documents

G.Vamvakas, B.Gatos, N. Stamatopoulos, and S.J.Perantonis
Computational Intelligence Laboratory, Institute of Informatics and Telecommunications,
National Center for Scientific Research "Demokritos",
GR-153 10 Agia Paraskevi, Athens, Greece
http://www.iit.demokritos.gr/cil/,
{gbam, bgat, nstam, sper}@iit.demokritos.gr

Abstract

In this paper a complete OCR methodology for recognizing historical documents, either printed or handwritten without any knowledge of the font, is presented. This methodology consists of three steps: The first two steps refer to creating a database for training using a set of documents, while the third one refers to recognition of new document images. First, a pre-processing step that includes image binarization and enhancement takes place. At a second step a top - down segmentation approach is used in order to detect text lines, words and characters. A clustering scheme is then adopted in order to group characters of similar shape. This is a semi-automatic procedure since the user is able to interact at any time in order to correct possible errors of clustering and assign an ASCII label. After this step, a database is created in order to be used for recognition. Finally, in the third step, for every new document image the above segmentation approach takes place while the recognition is based on the character database that has been produced at the previous step.

1. Introduction

The large amount of documents, either modern or historical, that we have in our possession nowadays, due to the expansion of digital libraries, has pointed out the need for reliable and accurate systems for processing them. Historical documents are of more importance because they are a significant part of our cultural heritage. During the last decades a lot of research has been done in the field of Optical Character Recognition (OCR). Numerous commercial products have been released that convert digitized documents into text files, usually in ASCII format. Although these products process machine printed documents successfully, when it comes to handwritten documents the results are not satisfactory enough. Moreover, such products are unable to process historical documents due to their low quality, lack of standard alphabets and presence of unknown fonts.To this end, recognition of historical documents is one of the most challenging tasks in OCR.

In the literature, historical document processing is mainly focused on document retrieval. Word-spotting techniques for searching and indexing historical documents have been introduced. In [1], word images are grouped into clusters of similar words by using image matching to find similarity. Then, by annotating "interesting" clusters, an index that links words to the locations where they occur can be built automatically. In [2] and [3] holistic word recognition approaches for historical documents are presented based on scalar and profile-based features and on matching word contours respectively. Their goal is to produce reasonable recognition accuracies which enable performing retrieval of handwritten pages from a user-supplied ASCII query. In [4], a word spotting technique based on combing synthetic data and user feed-back for keyword searching in historical printed documents is described.

However, transforming whole historical documents into text files is a much more difficult task. To the best of our knowledge, there are not a lot of works following this approach. Moreover, all related works tend to be focused on the unique characteristics of the corresponding historical document they process, such as content and writing style. In [5] and [6], OCR systems were developed respectively for the recognition of characters used in the Christian Orthodox Church Music notation. In [7], a

segmentation-free approach for recognizing old Greek handwritten documents especially from the early ages of the Byzantine Empire is presented. The basic characteristic of these documents is that there is no space between two consecutive words. Choudhury *et al.* [8] introduces an open-source programming framework for building systems that extract information from digitized historical documents empowering the document experts themselves to develop systems with reduced effort [9][10].

In this paper, an off-line recognition system for either machine printed or handwritten historical documents is presented. It consists of a pre-processing stage where documents are converted into binary images, a top – down segmentation technique that extracts the characters, the creation of a database by the extracted characters and a recognition stage where the database is used for converting any document into text file. The flowchart of this methodology is shown in Figure 1. The main advantage of this methodology is the fact that neither any knowledge of the fonts in advance nor the existence of a standard database is needed. So it can be applied to different types of documents and even deal with characters or ligatures that do not appear frequently. Depending on the type of historical documents that we want to process a database that assists the recognition procedure can be created. The more historical books are processed, the more databases are created and the more information from our cultural heritage is gained. Moreover, combination of several approaches from the OCR field such as binarization and segmentation or even from more general pattern recognition issues such as image enhancement and clustering, leads to a complete recognition system for historical documents either printed or handwritten. The remaining of the paper is organized as follows. Pre-processing is described in Section 2 while segmentation, database creation and recognition are discussed in Sections 3, 4 and 5 respectively. Experimental results are shown in Section 6 and finally conclusions are drawn in Section 7.

2. Image Binarization and Enhancement

Binarization is the starting step of most document image analysis systems and refers to the conversion of the gray-scale image to a binary image. Since historical document collections are most of the times of very low quality, an image enhancement stage is also essential.

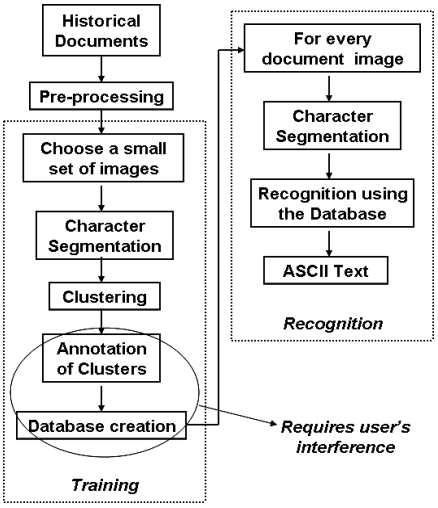
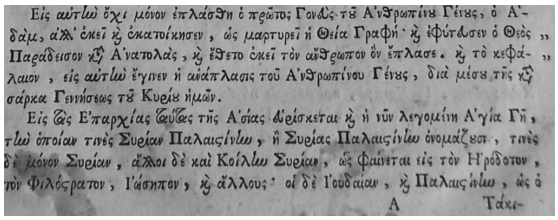
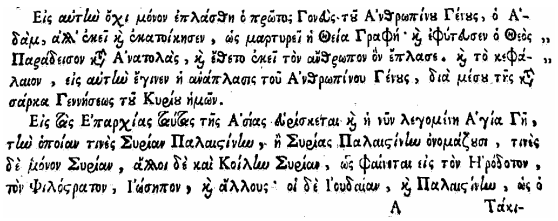


Figure 1. Flowchart of the proposed OCR methodology.

The scheme used for image binarization and enhancement is described in [11] and consists of five distinct steps: a preprocessing procedure using a low-pass Wiener filter, a rough estimation of foreground regions using Niblack's approach [12], a background surface calculation by interpolating neighboring background intensities, a thresholding by combining the calculated background surface with the original image and finally a post-processing step that improves the quality of text regions and preserves stroke connectivity.



(a)



(b)

Figure 2. Image Binarization. (a) Original Image and (b) Binarized Image.

3. Segmentation

After document binarization a top-down segmentation approach is applied. First lines of the documents are detected, then words are extracted and finally words are segmented in characters.

3.1. Text Line Detection

The text line segmentation methodology is based on [13] and consists of three distinct steps. The first step includes, connected component extraction and average character height estimation of the binary image. In the second step, a block - based Hough transform is used for the detection of potential text lines while a third step is used to correct possible splitting, to detect text lines that the previous step did not reveal and, finally, to separate vertically connected characters and assign them to text lines.

3.2. Word and Character Segmentation

Once the text lines have been located, projection profiles based on [14] in order to detect the words are used. Then the following method is adopted in order to separate them into letters. The algorithm is based on the segmentation algorithm described for touching numerals in [15]. The basic idea is that we can find possible segmentation paths linking the feature points on the skeleton of the word and its background.

Firstly, the average character height (*AH*) of the document using the technique proposed in [16] is calculated. We consider that the width of a letter cannot be less than *MinCharWidth* = 1/2**AH* and more than *MaxCharWidth*=3/2**AH*. Then, the connected components (CCs) of a word are detected and the following steps are applied to all CCs that have their height to width ratio less or equal to 0.5, in order to separate them into letters (Figs. 3a-1b).

Step 1: Calculate the skeleton of the CC and its background (Fig. 3c).

Step 2: Classify the skeleton in four different segments: (1) ***Top-segment*** (upper part of the background region), (2) ***Bottom-segment*** (lower part of the background region), (3) ***Stroke-segment*** (black pixels of the CC), (4) ***Hole-segment*** (hole-region of the background) (Fig. 3d).

Step 3: Locate the feature points of the skeleton (Fig. 3e). The different kinds of feature points are as follows: (1) ***Fork point***: The point on a segment which has more than two connected branches, (2) ***End point****:* The point on a segment that has only one neighbor pixel, (3) ***Corner-point***: The point on a segment where the curvature of the segment changes sharply.

Step 4: In this step all the possible segmentation paths are constructed (Fig. 3f). We simultaneously apply two different searches, downward search and upward search. In downward search, we construct all the segmentations paths which start from the feature points on the top-segment. Each segmentation path should start from a feature point on the top-segment, pass through one or two feature points on the stroke-segment and end at a feature point on the bottom-segment. The distance between two feature points (top-stroke, stroke-bottom or stroke-stroke) must be less than 0.8**AH*. Therefore, if no one feature point on the stroke-segment matches a feature point on the top-segment, a vertical path is constructed starting from this feature point on the top-segment until it touches the bottom-segment. Also, if no one feature point on the bottom-segment matches a feature point on the stroke-segment, a vertical path is constructed starting from this feature point on the stroke-segment until it touches the bottom-segment. A similar process is applied to upward search in order to construct possible segmentation paths from bottom-segment to top-segment. A segmentation path must satisfy the following constraints:

- Its length must be less than *AH*.
- Its width must be less than 1/3* *AH*.
- The ration between the count of foreground pixels and the count of background on the segmentation path must be smaller than 3.
- There must be no stroke-segment between two matched feature points.
- If it is a vertical path, it must be cut the stroke-segment only in one point.

Step 5: After locating all the possible segmentation paths the best ones are selected (Fig. 3g). In order to achieve this, starting from the beginning of component or from the last segmentation path that was selected, we take into consideration only segmentation paths that result to characters with width in the limit of [*MinCharWidth, MaxCharWidth*]. Among these, the best segmentation path is selected as the one that minimizes the following criteria:

- The divergence of resulting letter's width from the expected width (*AH*).
- The divergence of resulting letter's height from the expected height (*AH*).
- The length of the segmentation path.
- The width of the segmentation path.

We repeat this process until the CC cannot be segmented into other letters or no other possible segmentation paths exist.

Once the characters within the word have been located, in order to merge pieces of a broken character we calculate all the CCs of the word which have width less than *MinCharWidth* and then we merge them with the nearest character.

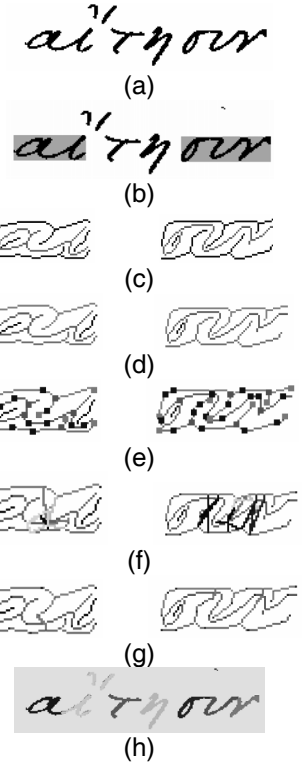


Figure 3: Example of character segmentation (a) a single word (b) CCs (c) skeleton of CCs (d) Classify of skeletons (e) Feature points (f) Possible segmentation paths (g) Best segmentation paths (h) Finally result.

4. Character Database

At this step, we choose a representative set of images for training and from these images characters are extracted following the segmentation procedure described in Section 3 (Fig. 4).Since class labeling of these characters is not available, no OCR methodology based on supervised learning that requires a training set of labeled patterns can be applied. Thus, our concern is first to "reveal" the organization of characters into "sensible" clusters (groups). Then, these clusters, after performing all required procedures to correct possible errors are labelled and, finally, the character database is created.

4.1. Feature Extraction

We first normalize all binary character images to a *N*x*N* matrix with respecting the original aspect ratio. In our case *N* is set to 60.

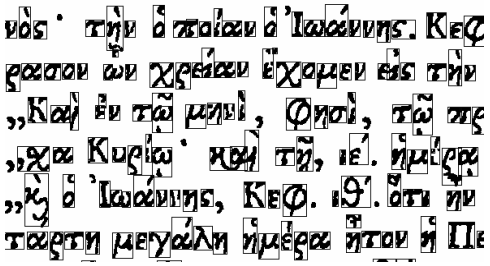


Figure 4. Character segmentation results.

We employ two types of features presented in [17]. The first set of features is based on zones. The image is divided into horizontal and vertical zones, and for each zone we calculate the density of the character pixels (Fig. 4).

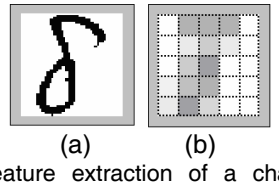


Figure 4. Feature extraction of a character image based on zones. (a) The normalized character image. (b) Features based on zones. Darker squares indicate higher density of character pixels.

In the second type of features, the area that is formed from the projections of the upper and lower as well as of the left and right character profiles is calculated. Firstly, the center mass (x_t,y_t) of the character image is found.

Upper/lower profiles are computed by considering, for each image column, the distance between the horizontal line $y=y_t$ and the closest pixel to the upper/lower boundary of the character image (Fig. 5b).

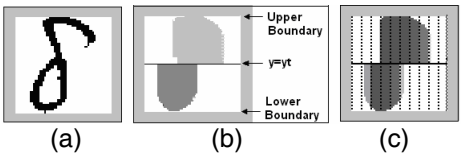
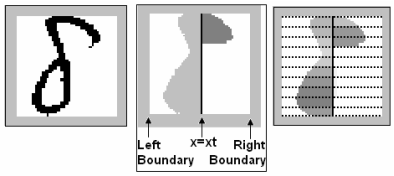


Figure 5: Feature extraction of a character image based on upper and lower character profile projections. (a) The normalized character image. (b) Upper and lower character profiles. (c) The extracted features. Darker squares indicate higher density of zone pixels.

This ends up in two zones (upper, lower) depending on y_t. Then both zones are divided into vertical blocks. For all blocks formed we calculate the

area of the upper/lower character profiles. Figure 5c illustrates the features extracted from a character image using upper/lower character profiles.

Similarly, we extract the features based on left/right character profiles (Fig. 6).



(a) (b) (c)

Figure 6: Feature extraction of a character image based on left and right character profile projections. (a) The normalized character image. (b) Left and right character profiles. (c) The extracted features. Darker squares indicate higher density of zone pixels.

In case or features based on zones the character image is divided into 5 horizontal and 5 vertical zones, thus resulting to 25 features. In case of features based on character (upper/lower) projection profiles the image is divided into 10 vertical zones, therefore the number of features is 20. Similarly, the image is divided into 10 horizontal zones so the number of features corresponding to features based on left/right projection profiles is also 20. Combination of features based on zones and features based on character profile projections led to the feature extraction model that uses a total of 65 features. So, every character is represented by a 65-dimensional feature vector F_i where $i = 1, 2, 3, 4 \ldots 65$.

4.2. Clustering

In our approach the well-known k-Means clustering algorithm [18] is used. An advantage of this algorithm is its computational simplicity. Also, as with all algorithms that use point representatives, k-Means is suitable for recovering compact clusters. Figure 7 illustrates the k-Means clustering algorithm.

Step 1: Initially choose the number of clusters to be k say.
Step 2: Choose arbitrary a set of k instances as initial centres of the clusters.
Step 3: Each instance is assigned to the cluster which is closest.
Step 4: The cluster centroids are recalculated either after each instance assignment, or after the whole cycle of re-assignments.
Step 5: GOTO Step 3 until no change occurs between two successive iterations.

Figure 7: The k-Means clustering algorithm.

4.3. Annotating Clusters

After feature extraction and clustering algorithm described in Sections 4.1 and 4.2 take place all characters are grouped into k clusters. Let each cluster represented as C_i where $i = 1, 2, 3 \ldots k$. The goal here is to turn these clusters into classes. Each class, C_i' where $i = 1, 2, 3 \ldots l$, has then to be assigned to an ASCII label. Unfortunately, this task cannot be automatic, so a tool has been developed that requires the user's interference.

The user is provided with an appropriate tool to handle clustering errors. Figure 8 is a screenshot of the developed tool. On the left side clusters created by the system are displayed to the user. Since no cluster has an ASCII code, initially all clusters are labeled as '?'. On the right side all the character instances are shown. The user is able to label clusters as shown in Figure 9.

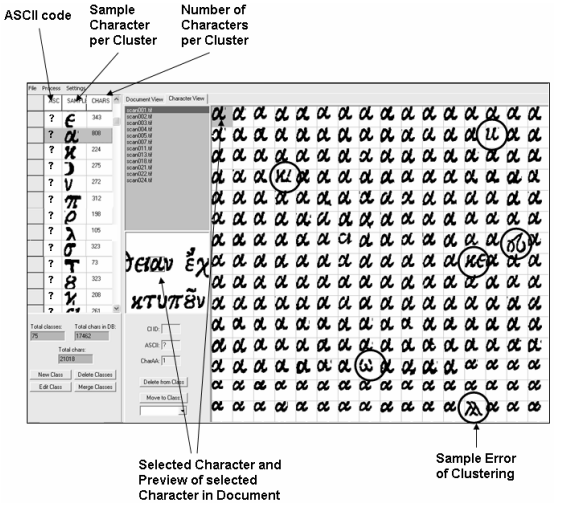


Figure 8: Result of clustering. Characters in circle are wrongly assigned to this cluster.

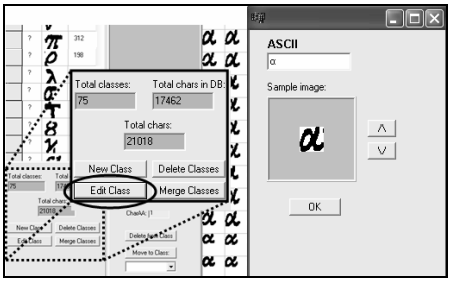


Figure 9: Giving an ASCII code to a cluster.

However, there are few characters which are wrongly placed into each class that need to be removed. The user can either select these misplaced

characters and delete them from the class or select and assign them to another class (see Fig. 10).

Moreover, the tool enables the user to perform a few more tasks to finalize the database such as merging or deleting two or more classes. Finally, characters or ligatures that rarely appear in the documents (see Table 1) can be assigned to classes of their own.

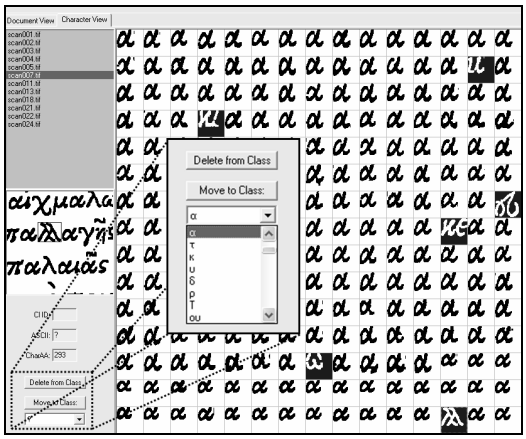


Figure 10: Handling with clustering errors.

Table 1: Samples of rarely appeared characters or ligatures.

Characters or ligatures	ASCII code
	δια
	ευ
	και
	κατά
	σθ
	τους
	τρ
	γαρ

5. Recognition

At this stage every document image that has not participated in the training phase is ready to be converted into a text file. Characters are extracted following the approach described in Section 3, each character is represented as a feature vector according to Section 4.1 and then all characters are classified using the database created in Section 4.3. For this classification problem the Support Vector (SVM) algorithm was used [19].

Formally, the SVM require the solution of an optimization problem, given a training set of instance-label pairs (x_i, y_i), i=1… m, where $x_i \in R^n$ and $y_i \in \{1,-1\}^m$. The optimization problem is defined as follows:

$$\begin{aligned} \min_{\omega,b,\xi} \quad & \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{m}\xi_i \\ subject\ \ to \quad & y_i(\omega^T\phi(x_i)+b) \geq 1-\xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{1}$$

SVM was used in conjunction with the Radial Basis Function (RBF) kernel, a popular, general-purpose yet powerful kernel, denoted as:

$$K(x_i, x_j) \equiv \exp(-\gamma \left\| x_i - x_j \right\|^2) \tag{2}$$

Furthermore, a grid search was performed in order to find the optimal values for both the variance parameter (γ) of the RBF kernel and the cost parameter (C) of SVM (see Eq. 1).

6. Experimental Results

For our experiments, old Greek Christian documents from the 17th century were used. During training phase two sets of 10 representative documents each, were used. One set of 10 printed documents *TrainSet1* and one set of 10 handwritten documents *TrainSet2*. In order to evaluate the performance of the segmentation procedure described in Section 3 for text line detection and word segmentation, we manually marked and extracted the ground truth on these train sets. The performance evaluation is based on counting the number of matches between the text lines and words detected by the algorithm with those in the in the ground truth [20]. A match is considered only if the matching score is equal to or above the evaluator's acceptance threshold T_α. The performance is recorded in terms of detection rate (*DR*) and recognition accuracy (*RA*), while as an overall measure the F-measure (*FM*) which is a weighted harmonic mean of detection rate and recognition accuracy is used (see Eq.3).

$$FM = \frac{2 \cdot DR \cdot RA}{DR + RA} \tag{3}$$

A global performance metric *SM* is extracted by calculating the average values for *FM* metric for text line and word segmentation. Table 2 shows the results where the acceptance threshold is set to T_α=90.

From these training sets a typewritten database *TWDB* and a handwritten database *HWDB* were created respectively following the methodology discussed in Section 4. For both cases, regarding the clustering step, *k* is set to 65 assuming that 65 clusters is a good starting point for the database creation. Table 3 depicts the results of training phase. *TWDB* is a

subset of *TrainSet1* and *HWDB* is a subset of *TrainSet2*.

Table 2: Evaluation of Segmentation.

Machine printed Documents (*TrainSet1*)				
	DR	RA	FM	SM
Text lines	98.4%	98.0%	98.2%	95.8%
Words	97%	90.3%	93.5%	
Handwritten Documents (*TrainSet2*)				
	DR	RA	FM	SM
Text lines	98.0%	98.6%	98.3%	94.2%
Words	88.7%	91.6%	90.1%	

Table 3: The TWDB and HWDB databases.

Machine printed Documents		
TrainSet1	65 unlabeled clusters	17462 chars
TWDB	67 labeled classes	13966 chars
Handwritten Documents		
TrainSet2	65 unlabeled clusters	8304 chars
HWDB	51 labeled classes	6758 chars

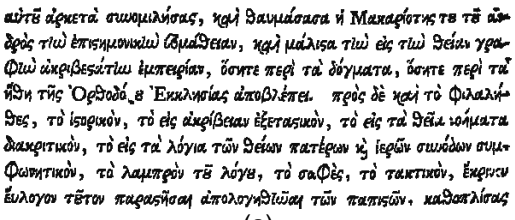
Moreover, in order to test the credibility of our databases as well as the effectiveness of our features two tests were carried out. Each database was split to a test set and a training set. The test set consists of 1/5 of each class while the rest is used for training. Table 4 shows the outcomes of these tests using the SVM with RBF. The variance parameter γ is set to 0.3 and the cost parameter C to 300.

Table 4: Evaluation of the TWDB and the HWDB databases.

TWDB		
Test Set	Train Set	Recognition Rate
2793 chars	11173 chars	95.44%
HWDB		
Test Set	Train Set	Recognition Rate
1351 chars	5407 chars	94.62 %

The overall recognition rates of our methodology are calculated using the *Edit Distance*, also known as *Levenshtein Distance* (*LD*) [21]. The *LD* between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single character. It is used to measure the similarity between the text file produced by our methodology and the ground truth text file that has been typed for a document image. Figures 11 and 12 show the text files in ASCII format for a printed and a handwritten document using the *TWDB* and the *HWDB* databases.

Finally, 5 printed and 5 handwritten documents, different from the ones used in training phase, formed two test sets, *TestSet1* and *TestSet2* respectively. Table 5 shows the recognition rates using the *LD*.



(a)

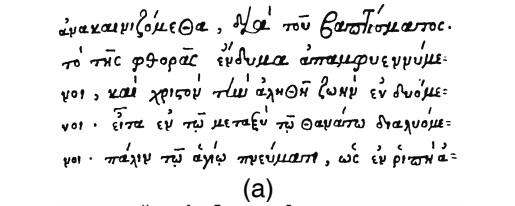
αυτου αρκετα συνομιλησας και θαυμασασα η Μακαριοτης του του αν
δρος την επιστημονικην ευμαθειαν και μαλιστα την εις την θειαν γρα
φην ακριβεσταταν εμπειριαν οσητε περι τα δογματα οσητε περι τα
ηθη της Ορθοδοξου Εκκλησιας αποβλεπει προς δε και το Φιλαλη
θες το ιστορικον το εις ακριβειαν εξεταστικον το εις τα θεια νοηματα
διακριτικον το εις τα λογια των θειων πατερων και ιερων συνοδων συμ
φωνητικον το λαμπρον του λογου το σαφες το τακτικον εκρινεν
ευλογον τουτον παραστησαι απολογηθηναι των παπιστων καθοπλισας

(b)

αυτου αρκετα συομιλησας καη θαυμασασα η Μακαριοτηστου του αν
δρος τημ επιστομονικην ευμαθειαν και μαλιστα την εες την θειαν γρα
φαυ ακριβεστατου εμπειριαν οσητε περι τα δογματα οσητε περι τα
ηθη της Ορθοδοξου Εκκλησας αποβλεπει προς δε και το φιλαλη
θες το ιστορικον το εις ακριβεηαν εξεταστικον το εις τα θεια Μηματα
διακριτουρον το εις τα λογια των θειων πατερων και ιερων συωοδαν συμ
φωνητικον το λαμπρον του λογου το σαφες το τακπκον εκριν
ευλογον τουτον παραστησαι απολωγηθεωση των παπιστων καθοπλισας

(c)

Figure 11: Conversion of a historical printed document into ASCII format. (a) Original document image, (b) Ground truth text file, (c) Result of recognition.



(a)

ανακαινιζομεθα δια του βαπτισματος
το της φθορας ενδυμα απαμφυεννυμε
νοι και χριστον την αληθη ζωην ενδυομε
νοι ειτα εν τω μεταξυ τω θανατω διαλυομε
νοι παλιν τω αγιω πνευματι ως εν ριπη α

(b)

ανακαινιζομεθα δια τον βαπτισρατος
το της τθορας ενδυμα απαμφυενκυμε
νοι και χριστον την αληΘη ξωην ενδυομε
νοι ειτα εν τ μεταξυ τω Θανατα διαλυαμε
νι παλ1ν τω αγιυ πνευματι ως ον ριπη α

(c)

Figure 12: Conversion of a historical handwritten document into ASCII format. (a) Original document image, (b) Ground truth text file, (c) Result of recognition.

7. Conclusions

In this paper a complete OCR methodology that assists the recognition of historical documents is presented. This methodology can be applied to either machine printed or handwritten documents. It requires

neither any knowledge of the fonts nor the existence of standard database because it can adjust depending on the type of documents that we want to process.

Our future work will focus on optimizing the current recognition results by exploiting new approaches for segmentation and new types of features.

Table 5: Recognition Rates using Levenshtein Distance.

	Recognition Rates
TestSet1	83.66%
TestSet2	72.68%

Acknowledgments

This research is carried out within the framework of the Greek Ministry of Research funded R&D project POLYTIMO [22] which aims to process and provide access to the content of valuable historical books and handwritten manuscripts

References

[1] T.M.Rath and R. Manmatha, "Word spotting for historical documents", *International Journal on Document Analysis and Recognition* (*IJDAR*), Vol.9, No 2 – 4, pp. 139 – 152 , 2006.

[2] V. Lavrenko, T. M. Rath, R. Manmatha: "Holistic Word Recognition for Handwritten Historical Documents", *Proceedings of the First International Workshop on Document Image Analysis for Libraries (DIAL'04),*pp 278-287, 2004.

[3] T. Adamek, N. E. O'Connor, A. F. Smeaton, "Word Matching Using Single-Closed Contours for Indexing Handwritten Historical Documents", *International Journal on Document Analysis and Recognition (IJDAR)*, special Issue on Analysis of Historical Documents, 2006.

[4] T. Konidaris, B. Gatos, K. Ntzios, I. Pratikakis, S. Theodoridis and S. J. Perantonis, " Keyword - Guided Word Spotting in Historical Printed Documents Using Synthetic Data and User feedback ", *International Journal on Document Analysis and Recognition (IJDAR)*, special issue on historical documents, Vol. 9, No. 2-4, pp. 167-177, 2007.

[5] V.G.Gezerlis and S.Theodoridis, "Optical Character Recognition for the Orthodox Hellenic Byzantine music notation", *Pattern Recognition*, Vol.35, pp. 895 – 914, 2002.

[6] L. Laskov, "Classification and Recognition of Neume Note Notation in Historical Documents", *International Conference of Computer Systems and Technologies (CompSysTech)*, 2006.

[7] K. Ntzios, B. Gatos, I. Pratikakis, T. Konidaris and S.J. Perantonis, "An Old Greek Handwritten OCR System based on an Efficient Segmentation-free Approach", *International Journal on Document Analysis and Recognition (IJDAR)*, special issue on historical documents, Vol. 9, No. 2-4, pp. 179-192, 2007.

[8] G.S. Choudhury, T. DiLauro, R. Ferguson, M. Droettboom, I. Fujinaga, "Document Recognition for a Million Books", *D-Lib Magazine*,Vol. 12, No. 3, 2006, www.dlib.org/dlib/march06/03contents.html.

[9] Droettboom, M., K. MacMillan, and I. Fujinaga, "The Gamera Framework for building custom recognition systems", Proceedings *of the Symposium on Document Image Understanding Technologies, (SDIUT)*, pp. 275-86, 2003.

[10] Droettboom, M., K. MacMillan, I. Fujinaga, G. S. Choudhury, T. DiLauro, M. Patton, and T. Anderson, "Using Gamera for the recognition of cultural heritage materials", *Proceedings of the Joint Conference on Digital Libraries, (JCDL 2002)*, pp. 11-17, 2002.

[11] Gatos, B., Pratikakis, I., Perantonis, S.J., "Adaptive degraded document image binarization", *Pattern Recognition*, 39, 317–327, 2006.

[12] Niblack,W., "An Introduction to Digital Image Processing", pp.115–116. Prentice Hall, Englewood Cliffs, NJ, (1986).

[13] G. Louloudis, B. Gatos and C. Halatsis, "Text Line Detection in Unconstrained Handwritten Documents Using a Block-Based Hough Transform Approach", *9th International Conference on Document Analysis and Recognition (ICDAR'07)*, pp. 599-603, Curitiba, Brazil, September 2007.

[14] A. Antonacopoulos, D. Karatzas, "Document Image analysis for World War II personal records", *First International Workshop on Document Image Analysis for Libraries*, Palo Alto, 2004, pp. 336-341.

[15] Yi – Kai Chen and Jhing – Fa Wang, "Segmentation of Single- or Multiple-Touching Handwritten Numeral String Using Background and Foreground Analysis", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, pp. 1304-1317, 2000.

[16] B. Gatos, T. Konidaris, K. Ntzios, I. Pratikakis and S. J. Perantonis, "A Segmentation-free Approach for Keyword Search in Historical Typewritten Documents", *9th International Conference on Document Analysis and Recognition*, Seoul, Korea, August 2005, pp. 54-58.

[17] G. Vamvakas, B. Gatos, S. Petridis and N. Stamatopoulos, "An Efficient Feature Extraction and Dimensionality Reduction Scheme for Isolated Greek Handwritten Character Recognition", *9th International Conference on Document Analysis and Recognition (ICDAR'07)*, pp. 1073-1077, Curitiba, Brazil, September 2007.

[18] Theodoridis, S., and Koutroumbas, K., *Pattern Recognition*, Academic Press, 1997.

[19] Cortes C., and Vapnik, V., "Support-vector network", *Machine Learning*, vol. 20, pp. 273-297, 1997.

[20] B. Gatos, A. Antonacopoulos, N. Stamatopoulos, "Handwriting Segmentation Contest", *9th International Conference on Document Analysis and Recognition (ICDAR)*, Curitiba, Brazil, 2007, pp. 1284-1288.

[21] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals", *Soviet Physics Doklady* 10 (1966):707–710.

[22] POLYTIMO project, http://iit.demokritos.gr/cil/Polytimo, 2007.