

# A Segmentation-Free Recognition Technique to Assist Old Greek Handwritten Manuscript OCR

Basilios Gatos<sup>1</sup>, Kostas Ntzios<sup>1,2</sup>, Ioannis Pratikakis<sup>1</sup>, Sergios Petridis<sup>1</sup>,  
T. Konidakis<sup>1</sup>, and Stavros J. Perantonis<sup>1</sup>

<sup>1</sup> Computational Intelligence Laboratory, Institute of Informatics and Telecommunications,  
National Research Center "Demokritos",

153 10 Athens, Greece

{bgat,ntzios,ipratika,petridis,tkonid,sper}@iit.demokritos.gr

<http://www.iit.demokritos.gr/cil>

<sup>2</sup> Department of Informatics & Telecommunications,

National & Kapodistrian University of Athens,

Athens, Greece

ntzios@di.uoa.gr

**Abstract.** Recognition of old Greek manuscripts is essential for quick and efficient content exploitation of the valuable old Greek historical collections. In this paper, we focus on the problem of recognizing early Christian Greek manuscripts written in lower case letters. Based on the existence of hole regions in the majority of characters and character ligatures in these scripts, we propose a novel, segmentation-free, fast and efficient technique that assists the recognition procedure by tracing and recognizing the most frequently appearing characters or character ligatures. First, we detect hole regions that exist in the character body. Then, the protrusions in the outer contour outline of the connected components that contain the character hole regions are used for the classification of the area around holes to a specific character or a character ligature. The proposed method gives highly accurate results and offers great assistance to old Greek handwritten manuscript OCR.

## 1 Introduction

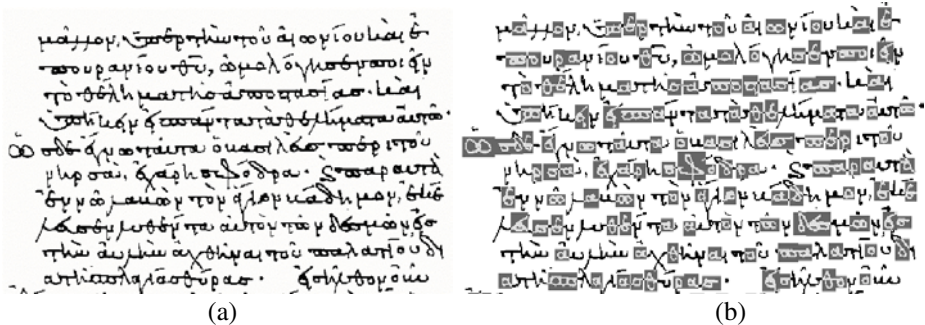
Recognition of old Greek manuscripts is essential for quick and efficient content exploitation of the valuable old Greek historical collections. In this paper, we focus on early Christian Greek manuscripts written in lower case letters. The Sinaitic Codex Number Three, which contains the Book of Job, constitutes the main axis of our research (see Fig. 1a).

The field of handwritten character recognition has made great progress during the past years [1]. Many methods were developed in an attempt to satisfy the need for such systems that exists in various applications like automatic reading of postal addresses and bank checks, form processing etc. [2,3]. For handwritten character recognition two main approaches can be identified: The global approach [4,5] and the segmentation approach [6,7]. The global approach entails the recognition of the whole

word while the segmentation approach requires that each word has to be segmented into letters. Although the global approach can be found in the literature as “segmentation-free” approach, it involves a word detection task. Some approaches that do not involve any segmentation task are based on the concept and techniques of occluded object recognition [8,9]. According to those approaches, significant geometric features, such as short line segments, enclosed regions and corners, are extracted from the image of the entire page.

Traditional techniques for handwritten recognition cannot be easily applied to entail early Christian Greek manuscripts written in lower case letters, since they explore several unique characteristics, such as:

- Continuous connection between characters even between different words.
- Largely standardized scripts since they are immediate predecessors of early printed books.
- Character ligatures are very common.
- Hole regions appear in the majority of character and character ligatures. As shown in Fig. 1b, hole regions appear in letters “α”, “ο”, “σ”, “ε”, “δ”, “ω”, “π”, “θ”, “φ” and in letter ligatures “ας”, “ες” etc. These constitute 60% of all characters of a typical old Greek manuscript.



**Fig. 1.** (a) Early Christian Greek manuscript - The Sinaitic Codex Number Three; (b) Identified characters or character ligatures that contain hole regions.

Due to the continuous connection between characters we developed a segmentation-free recognition technique to assist an Old Greek Handwritten Manuscript OCR. Based on the existence of hole regions in the majority of characters and character ligatures, we propose a technique for tracing and recognition of characters that contain holes. The proposed methodology does not aim at a complete character recognition system for old Greek manuscripts. It rather aims at an assessment of the recognition procedure by tracing and recognizing the most frequently appearing characters or character ligatures, using a segmentation-free, quick and efficient approach. The method has been developed in the framework of the Greek GSRT-funded R&D project, D-SCRIBE, which aims to develop an integrated system for digitization and processing of old Greek manuscripts.

## 2 Methodology

The proposed methodology consists of several distinct stages. Initially, we trace hole regions that exist in character bodies. We suggest a novel fast algorithm based on processing the white runs of the initial b/w image. This algorithm permits the extraction of the character hole regions but rejects hole regions of larger dimension, such as holes inside frames, diagrams, etc. At the next stage of our approach, all character hole regions are initially grouped into several categories according to the distance between them. In this way, character hole regions are classified as: an isolated hole, two horizontal neighboring holes, three horizontal neighboring holes, etc. The final stage of our approach includes the final classification of hole regions or groups of hole regions into a character or a ligature according to the protrusions that exist on the outer contour outline of the connected components that contain the character hole regions.

### 2.1 Character Hole Region Detection

There exist several hole detection algorithms mainly based on contour following and distinguishing external and internal contours [10,11]. We suggest a novel fast algorithm for hole detection based on processing the white runs of the b/w image. Below follows a step-by-step description of the proposed algorithm.

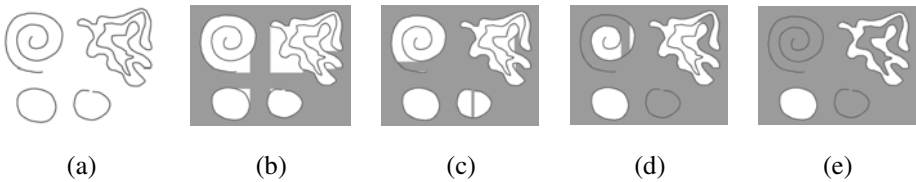
*Step 1.* All horizontal and vertical image white runs that neighbor with image borders or are of length greater than  $L$  are flagged, where  $L$  is a characteristic length reflecting character size.

*Step 2.* All horizontal and vertical white runs of not flagged pixels that neighbor with the flagged pixels of step 1 are flagged as well.

*Step 3.* Repeat step 2 until no pixel remains to be flagged.

*Step 4.* All remaining white runs of not flagged pixels belong to image holes. Possible holes with very small white run lengths are ignored.

The proposed algorithm for hole detection extracts only the character hole regions and not other hole regions of larger dimension, with white run length greater than  $L$ , such as holes inside frames, diagrams etc. An example of the proposed hole detection algorithm is demonstrated in Fig.2.

















**Fig. 2.** Hole detection algorithm demonstration: (a) Original image;(b)-(e) Resulting image after 1,2,5 and 19 iterations, respectively.

## 2.2 Hole Grouping

Hole regions are grouped together according to their proximity and topology. Table 1 shows the dictionary of the hole patterns, including their configuration and the corresponding character or character ligature.

**Table 1.** Hole patterns dictionary.

Pattern ID	1	2	3	4	5	6
Pattern	o	oo	ooo	oooo	o o	o oo
Characters or character ligatures	 ( $\alpha$ ),  ( $\omega$ )  ( $\epsilon$ ),  ( $\sigma$ )  ( $\rho$ ),  ( $\delta$ )	 ( $\pi$ ),  ( $\omega$ ),  ( $\epsilon\sigma$ )	 ( $\sigma\pi$ ),  ( $\epsilon\pi$ )	 ( $\alpha\pi\omega$ )	 ( $\theta$ )	 ( $\phi$ )

## 2.3 Character Detection

Feature extraction is applied to characters that contain one or more holes. In order to calculate these features it is needed to isolate the connected component into characters that contain one or more holes. The method we use for character isolation takes advantage of the characteristics of the outer contours' pixels of the component and the contours' pixels of the hole. The coordinates of these pixels can be extracted by using a contour following algorithm [12]. Fig. 3 shows three components with 1 hole each.

Let  $\mathcal{H} = \{(x_i^{\mathcal{H}}, y_i^{\mathcal{H}}), i \in [1, n]\}$ , be the set of the pixel coordinates, that composes the contour of the hole and  $C = \{(x_i^C, y_i^C), i \in [1, m]\}$ , be the set of the pixel coordinates belonging to the outer contour of the character.

We define the distance between the hole and the connected character component as:

$$D(\mathcal{H}, C) = \max_i \left\{ \min_j \left\{ \sqrt{(x_i^{\mathcal{H}} - x_j^C)^2 + (y_i^{\mathcal{H}} - y_j^C)^2} \right\} \right\} \quad (1)$$

Calculating  $D(\mathcal{H}, C)$  and using pixel coordinates which compose the contour of the hole, it is easy to isolate the pixels of the connected character component contained in a bounding box  $W$  with the following top-left  $(x_{TL}, y_{TL})$  and bottom right corner coordinates  $(x_{BR}, y_{BR})$ :

$$x_{TL} = \begin{cases} \min_i (x_i^C) & \text{If } \min_i (x_i^C) > \min_i (x_i^{\mathcal{H}}) - 2 \cdot D(\mathcal{H}, C) \\ \min_i (x_i^{\mathcal{H}}) - 2 \cdot D(\mathcal{H}, C) & \text{otherwise} \end{cases} \quad (2)$$

$$y_{TL} = \begin{cases} \min_i (y_i^C) & \text{If } \min_i (y_i^C) > \min_i (y_i^{\mathcal{H}}) - 2 \cdot D(\mathcal{H}, C) \\ \min_i (y_i^{\mathcal{H}}) - 2 \cdot D(\mathcal{H}, C) & \text{otherwise} \end{cases}$$

$$x_{BR} = \begin{cases} \max_i(x_i^C) & \text{If } \max_i(x_i^C) < \max_i(x_i^H) + 2 \cdot D(\mathcal{H}, C) \\ \max_i(x_i^H) + 2 \cdot D(\mathcal{H}, C) & \text{otherwise} \end{cases}$$

$$y_{BR} = \begin{cases} \max_i(y_i^C) & \text{if } \max_i(y_i^C) < \max_i(y_i^H) + 2 \cdot D(\mathcal{H}, C) \\ \max_i(y_i^H) + 2 \cdot D(\mathcal{H}, C) & \text{otherwise} \end{cases}$$

Fig. 4 shows the contours of the component with the respective windows  $W$  around the holes.



Fig. 3. An image of three independent components that contains a Greek word.

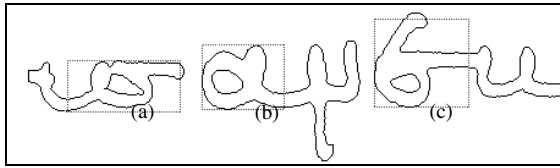


Fig. 4. The contours of the component with the respective window  $W$  around the holes. a) Greek character “σ”, b) Greek character “α”, c) Greek character “ε”.

### 2.4 Feature Estimation

The features are computed in order to identify all segments that belong to a protrusion as it is delineated by the outer contour outline of the isolated character. Feature extraction is applied in two modes; the so-called *vertical* and *horizontal* mode. The vertical mode is used to describe the protrusible segments that may exist on the top and the bottom of the character while the horizontal mode is used to describe the protrusible segments that may exist on the right side of the character (see Fig. 4). The feature set which is composed of 9 features  $\mathcal{F}=(f_1, f_2, \dots, f_9)$  express the probability of a segment being part of a protrusion. Features  $f_1, f_2, f_3$  describe the protrusible segments that may appear on the top of the character,  $f_4, f_5, f_6$  describe the protrusible segments that may exist at the bottom of the character and the last three features describe the protrusible segments that may exist on the right side of the character. In our approach we have not taken into account segments that may belong to left protrusions, due to our observation that in all cases they correspond to a letter ligament rather than the main body of a character. A sound example is given in Fig. 4a in the case of Greek character “σ”.

Feature estimation is employed in the following steps:

- *Step 1: Bounding box division into blocks*

In vertical mode we compute the mean value  $Y$  of all  $y^{\mathcal{H}}$ -coordinates of  $H$  set, which is denoted as:

$$Y = \frac{1}{n} \sum_{i=1}^n y_i^{\mathcal{H}} \quad (3)$$

We divide  $W$  into three areas of equal width and assign a divide line  $F(x)=Y$  as it is shown in Fig 5a, resulting in 6 blocks  $R_1, \dots, R_6$ .

Furthermore, for the horizontal mode we compute the mean value  $X$  of all  $x^{\mathcal{H}}$ -coordinates of  $H$  set, which is denoted as:

$$X = \frac{1}{n} \sum_{i=1}^n x_i^{\mathcal{H}} \quad (4)$$

Similarly, we divide  $W$  into three areas of equal height and assign a divide line  $F(y)=X$  as it is shown in Fig 6a, resulting in extra 6 blocks  $R_7 \dots R_{12}$ .

- *Step 2: Block correction*

In order to describe the pixels of a protrusible segment, ignoring the other pixels of the character, the process must be restricted to the upper side of blocks  $R_i$  with  $i \in [1,3]$ , to the lower side of blocks  $R_i$  with  $i \in [4,6]$  and to the most right side of blocks  $R_i$  with  $i \in [7,9]$ . Blocks  $R_{10}, R_{11}, R_{12}$  are not used for feature extraction because any protrusion in these regions is a letter ligature rather than a dominant part of the character. Thus, for each block, we compute an offset ( $P_i$ ) to the divide line at the corresponding mode (see Fig 5b, 6b). This leads to a new assignment of the block area that is different for each block that initially partitioned the bounding box. The corrected blocks will delimit the area for the block-based feature computation.

The offset, at the vertical mode ( $P_{y_i}$ ) and the horizontal mode ( $P_{x_i}$ ) is calculated as follows:

$$P_{y_i} = \begin{cases} \min_j (y_j^{\mathcal{H}_{R_i}}) - D(\mathcal{H}_{R_i}, C) & (\mathcal{H}_{R_i} \neq \text{null}) \vee (i \in [1,3]) \\ \max_j (y_j^{\mathcal{H}_{R_i}}) + D(\mathcal{H}_{R_i}, C) & (\mathcal{H}_{R_i} \neq \text{null}) \vee (i \in [4,6]) \\ D(\mathcal{H}, C) & \mathcal{H}_{R_i} = \text{null} \end{cases} \quad (5)$$

$$P_{x_i} = \begin{cases} \max_j (x_j^{\mathcal{H}_{R_i}}) + D(\mathcal{H}_{R_i}, C) & (\mathcal{H}_{R_i} \neq \text{null}) \vee (i \in [7,9]) \\ D(\mathcal{H}_{R_i}, C) & \mathcal{H}_{R_i} = \text{null} \end{cases}$$

where  $\mathcal{H}_{R_i} = \{(x_j^{\mathcal{H}_{R_i}}, y_j^{\mathcal{H}_{R_i}}) \subset \mathcal{H}, j \in [1, n_{R_i}], i \in [1,9]\}$  are the sets of the pixel coordinates, which consist the part of the hole contour depicted in block  $R_i$ .

- *Step 3: Block-based feature computation*

For this step, crucial role is played by the directions of the contour pixels evaluated by considering pairs of adjacent pixels during a clockwise tracing of the outer con-

tour. Hence, for each pixel  $j$  of the contour we introduce as  $s_j$  the local orientation of the contour, taking nominal values from the set  $\{W, SW, S, SE, E, NE, N, NW\}$ . Once the directions are evaluated the proposed feature  $f_i$  is defined as follows:

$$f_i = \frac{1}{D(\mathcal{H}, C)} \sum_{j=1}^{m_i} g_i(s_j^i) \quad (6)$$

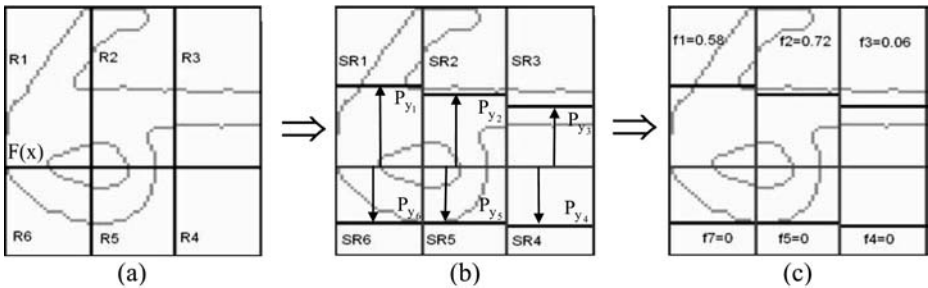
where  $g_i(\cdot)$  is a function depending on the orientation of the pixel and the block considered and  $m_i$  is the total number of pixels of the outer contour in side block  $SR_i$ . The term  $D(\mathcal{H}, C)$  is used as a “normalization” factor allowing for the feature to be independent of the character scaling. The  $g_i(\cdot)$ 's are explicitly defined in Table 2. They determine a pixel template unique for each block  $i$  that expresses the expected local orientation of the corresponding contour pixels. During contour following,  $g_i(\cdot)$  equals 1 when the examined pixel belongs to either a vertical or horizontal protrusible segment, and otherwise it equals 0.

Extension of the above procedure for characters having more than one hole is straightforward. Multiple holes are treated as one after a merging process which preserves the initially topology.

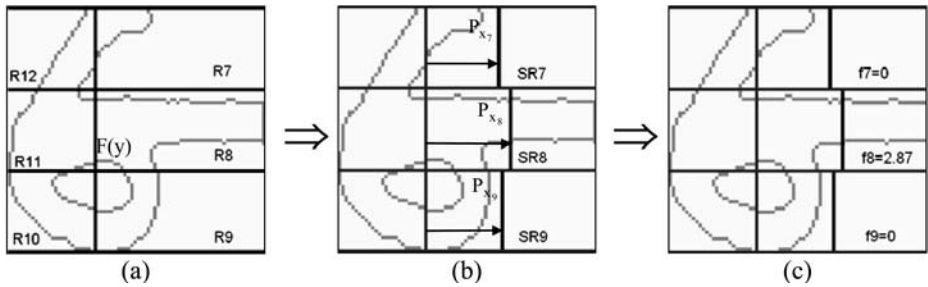
**Table 2.**  $g_i(\cdot)$  in 8-connectivity contour following.

	E	NE	N	NW	W	SW	S	SE
$g_1$	0	0	1	1	0	1	1	1
$g_2$	0	1	1	1	0	1	1	1
$g_3$	0	1	1	1	0	1	1	0
$g_4$	0	1	1	1	0	0	1	1
$g_5$	0	1	1	1	0	1	1	1
$g_6$	0	1	1	0	0	1	1	1
$g_7$	1	1	0	1	1	1	0	0
$g_8$	1	1	0	1	1	1	0	1
$g_9$	1	1	0	1	1	0	0	1

↖	N	↗
W	•	E
↙	S	↘



**Fig. 5.** Vertical mode. a) Region  $R_1, R_2, \dots, R_6$ , defined by the  $F(x)$  b) Sub-regions  $SR_1, SR_2, \dots, SR_6$  defined by  $P_{y_i}$  offsets respectively. c) The values of features  $f_1 \dots f_6$  for the segmented Greek letter “ε”.



**Fig. 6.** Horizontal mode. a) Regions  $R_7, R_8, R_9$  defined by  $F(y)$  b) Sub-regions  $SR_7, SR_8, SR_9$  defined by the  $P_{x_i}$  offsets respectively c) The values of features  $f_7, f_8, f_9$  for the segmented Greek letter “ε”.

### 3 Experimental Results

The purpose of the experiments was to test the classification performance of the handwritten manuscripts with respect to the proposed hole detection and feature extraction techniques. The overall experiments used samples coming from three different writers of the Book of Job collection, manually labeled with the correct answers.

#### 3.1 Hole Detection Evaluation

The first series of experiments tested the performance of the hole pattern detection algorithm. Table 3 shows the dictionary of the hole patterns, including the number of pattern occurrences in the sample. Notice that the majority of characters is classified as having one or two adjacent holes. Especially, patterns with id 5 and 6 correspond to exactly one character each, which implies that a detection of these patterns is equivalent to identification of the corresponding characters.

**Table 3.** The dictionary of hole patterns including the number of pattern occurrences.

Pattern ID	1	2	3	4	5	6
Pattern	o	oo	ooo	oooo	o o	o oo
Occurrences	786	130	7	3	30	11

The hole detection algorithm requires no training and thus the totality of the available labeled sample was used as a test set. Table 4 summarizes the results obtained by applying the algorithm, showing the recall and the precision rates for each one of the hole patterns. As seen from Table 4, the performance on both recall and precision is satisfactory. Again, the reader should focus on the results concerning the first two patterns, which correspond to the majority of samples and have thus stronger statistical significance.



**Table 4.** Recall / Precision for the characters or character ligatures in each of the hole patterns.

ID	Recall	Precision
1	95,81	97,42
2	94,61	86,62
3	100	53,85
4	100	100
5	84,37	96,43
6	87,5	100
Overall weighted Recall/Precision	<b>95,24</b>	

### 3.2 Character and Character Ligature Identification

The second series of experiments test the performance of the character and character ligature identification algorithm. The experiments concentrate on characters and character ligatures that correspond to patterns 1 and 2 (Table 1), since these patterns appear in a great variety of characters. The experiment involved two steps: the feature vector extraction step and the training and testing of a statistical classifier. The focus of the experiments was on testing suitability of the extracted features, by measuring the classification performance of popular classification algorithms.

To that end, characters and character ligatures coming from 3 different writers (referenced as im1, im2 and im3, respectively – Table 5) were gathered into two different datasets CL1 and CL2, according to their pattern (one or two holes). Within each dataset, the feature extraction algorithm described in section 2.4 was applied to each character or character ligature. Notice that in this step, there has been no need for splitting the data in reference and test sets, since the feature extraction algorithm required no training. For the classification step, however, such split is necessary in order to measure the generalization performance of the trained classifiers. Thus, a series of different scenarios that the experiments were to be based on, with various ways of splitting has been constructed. Table 5 list the scenarios for CL1 and CL2.

**Table 5.** Training set / Test set configuration for the CL1 dataset.

ID	Samples	Training	Test
CL1-1	754	70 (10% im1, im2)	684 (90% im1, im2)
CL1-2	754	147 (20% im1, im2)	607 (80% im1, im2)
CL1-3	479	147 (20% im1, im2)	332 (100% im3)
CL1-4	402	70 (10% im1, im2)	332 (100% im3)
CL1-5	1086	754 (100% im1, im2)	332 (100% im3)
CL2-1	123	12 (10% im1, im2)	111 (90% im1, im2)
CL2-2	123	24 (20% im1, im2)	99 (80% im1, im2)
CL2-3	73	12 (10% im1, im2)	61 (100% im3)
CL2-4	85	24 (20% im1, im2)	61 (100% im3)
CL2-5	184	123 (100% im1, im2)	61 (100% im3)

The classification step was performed using two well known classification algorithms, K-NN [13] and SVM [13,14]. K-NN was used in two variants, with L1 norm and L2 norm each time. Moreover, exhaustive search took place in order to determine the value of neighbors ( $k$ ) that gave the best score. On the other hand, SVM was used in conjunction with the RBF kernel, a popular, general-purpose yet powerful kernel. Again, a grid search was performed in order to find the optimum values for both the variance parameter of the RBF kernel ( $\gamma$ ) and the cost parameter of SVM ( $c$ ). The results for CL1 and CL2, together with the optimal parameter values, are listed in Table 6 and Table 7 respectively.

**Table 6.** Performance of the algorithms for the CL1 dataset. Numbers in parenthesis represent the parameters used for achieving the optimum scores. The ID column corresponds to the different scenarios as shown in Table 5. For the SVM kernel, the number of support vectors found is also given.

ID	KNN-L1	KNN-L2	SVM-RBF
CL1-1	90.49 ( $k=1$ )	90.78 ( $k=1$ )	<b>93.42</b> ( $\gamma=0.94$ , $c=50$ , $SVs=45$ )
CL1-2	94.06 ( $k=1$ )	93.73 ( $k=1$ )	<b>95.05</b> ( $\gamma=0.98$ , $c=50$ , $SVs=62$ )
CL1-3	<b>97.89</b> ( $k=2$ )	97.59 ( $k=2$ )	<b>97.89</b> ( $\gamma=0.04$ , $c=50$ , $SVs=63$ )
CL1-4	94.27 ( $k=1$ )	96.08 ( $k=1$ )	<b>97.28</b> ( $\gamma=0.2$ , $c=100$ , $SVs=42$ )
CL1-5	98.19 ( $k=10$ )	98.19 ( $k=4$ )	<b>98.49</b> ( $\gamma=0.8$ , $c=1$ , $SVs=216$ )

**Table 7.** Performance of the algorithms for the CL2 dataset. Numbers in parenthesis represent the parameters used for achieving the optimum scores. The ID column corresponds to the different scenarios as shown in Table 5.

ID	KNN-L1	KNN-L2	SVM-RBF
CL2-1	95.49 ( $k=1$ )	93.69 ( $k=1$ )	<b>94.59</b> ( $\gamma=0.1$ , $c=10$ , $SVs=9$ )
CL2-2	93.93 ( $k=1$ )	92.92 ( $k=1$ )	<b>94.94</b> ( $\gamma=0.1$ , $c=20$ , $SVs=10$ )
CL2-3	<b>100.0</b> ( $k=1$ )	<b>100.0</b> ( $k=1$ )	<b>100.0</b> ( $\gamma=0.1$ , $c=10$ , $SVs=9$ )
CL2-4	98.36 ( $k=6$ )	95.99 ( $k=1$ )	<b>100.0</b> ( $\gamma=0.1$ , $c=10$ , $SVs=11$ )
CL2-5	96.72 ( $k=1$ )	98.36 ( $k=1$ )	<b>100.0</b> ( $\gamma=0.1$ , $c=10$ , $SVs=24$ )

The scores that were achieved in both datasets were very high even in cases where the samples were few. This particular aspect is very encouraging, since it proves the good generalization performance of the algorithms. Furthermore, the fact that the algorithms were able to generalize so well, is also due to the robust feature vector representation scheme. The features selected are suitable for enabling the good overall performance. Moreover, it can be assumed that if new characters were to be added, the performance of the algorithms, concerning the new characters, would still be high.

As a particular case, we present two confusion matrices resulting from specific scenarios, showing the points where maximum confusion between characters based on the extracted features is to be expected. Table 8 shows the confusion matrix for the scenario CL1-1 of Table 5. This scenario involved taking 10% of the samples from the first two images as training set and 90% of the samples of the same images as test set as far as CL1 dataset is concerned. The results correspond to the application of the SVM algorithm to the training and test set. In particular, notice the case of characters

“ $\alpha$ ” and “ $\varepsilon$ ”, which were mutually misclassified and the case of character “ $\delta$ ” which was misclassified 9 times as “ $\sigma$ ”.

**Table 8.** Confusion matrix for the scenario CL1-1 in Table 5.

	$\alpha$	$\varepsilon$	$\theta$	$\sigma$	$\rho$	$\delta$
A	170	9	0	0	1	1
E	6	119	0	0	0	1
$\Theta$	0	1	0	0	0	0
O	1	0	0	155	2	0
$\Sigma$	5	6	0	106	0	0
P	0	0	0	3	65	0
$\Delta$	0	2	0	9	0	22

**Table 9.** Confusion matrix for the scenario CL2-1 in Table 5.

	$\pi$	$\varepsilon\sigma$	$\omega$
$\pi$	48	0	6
$\varepsilon\sigma$	0	2	0
$\omega$	0	0	55

Table 9 shows the confusion matrix for the scenario CL2-1 shown in Table 5. For this scenario 10% of the samples from the first two images were chosen as the training set and 90% of the samples of the same images were used as the test set as far as CL2 dataset is concerned. The only errors produced here were that 6 of the  $\pi$  characters were classified as  $\omega$ . The rest of the characters and character ligatures in this scenario were correctly classified.

## 4 Conclusions and Further Work

In this paper, we present a novel methodology that assists recognition of early Christian Greek manuscripts written in lower case letters. We do not provide a solution for a complete character recognition system but we strive toward an assessment of the recognition procedure by tracing and recognizing the most frequently appearing characters or character ligatures, using a segmentation-free, quick and efficient approach. Based on the observation that hole regions appear in the majority of characters and character ligatures, we propose a recognition technique that consists of several distinct stages. Experimental results show that the proposed method gives highly accurate results that offers a great assistance to old Greek handwritten manuscript interpretation.

Future work involves the detection and recognition of all the remaining old Greek handwritten character and character ligatures that do not include holes, as well as the testing of the performance of the proposed technique for other types of old handwritten historical manuscripts.

## Acknowledgment

The authors would like to thank Mount Sina Foundation for providing us samples from their historical document collection.

This work is supported by the Greek GSRT-funded R&D project, D-SCRIBE, which aims to develop an integrated system for digitization and processing of old Greek manuscripts.

## References

1. Verma, B., Blumenstein, M., Kukarni, S.: Recent Achievements in Off-line Handwriting Recognition Systems. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'98), Melbourne, Australia (1998) 27-33
2. Lam L. et al.: Automatic Processing of Information on Cheques. International Conference on Systems, Man & Cybernetics (1995) 2353-2358
3. Brakensiek, A., Rottland, J., Rigoll, G.: Confidence measures for an address reading system. Seventh International Conference on Document Analysis and Recognition (ICDAR 2003) (2003) 294-298
4. Suen, C.Y. et al.: Building a New Generation of Handwriting Recognition Systems. Pattern Recognition Letters, 14 (1993) 303-315
5. Guillevic D., Suen, C.Y.: HMM Word Recognition Engine. Fourth International Conference on Document Analysis and Recognition (ICDAR97) (1997) 544
6. Kavallieratou E., Fakotakis, N., Kokkinakis, G.: Handwritten character recognition based on structural characteristics. 16<sup>th</sup> International Conference on Pattern Recognition (2002) 139-142
7. Eastwood B. et al.: A Feature Based Neural Network Segmenter for Handwritten Words. International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'97), Australia (1997) 286-290
8. Chen C.H., de Curtins, J.: Word Recognition in a Segmentation-Free Approach to OCR. Second International Conference on Document Analysis and Recognition (ICDAR'93) (2003) 573-576
9. Chen C.H., de Curtins, J.: A Segmentation-free Approach to OCR. IEEE Workshop on Applications of Computer Vision (1992) 190-196
10. Pavlidis, T.: Algorithms for Graphics and Image Processing. Computer Science Press, Rockville, MD (1992)
11. Xia, F.: Normal vector and winding number in 2D digital images with their application for hole detection. Pattern Recognition 36 (2003) 1383-1395
12. Jain, A.: Fundamentals of digital image processing. Prentice Hall (1989)
13. Theodoridis, S., Koutroumbas K.: Pattern Recognition, Academic Press (1997)
14. Chang, C-C., Lin, C.-J., LIBSVM: A library for support vector machines (2001), Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>