A NEW FRAMEWORK BASED ON SIGNATURE PATCHES, MICRO REGISTRATION, AND SPARSE REPRESENTATION FOR OPTICAL TEXT RECOGNITION

Reza Farrahi Moghaddam, Fereydoun Farrahi Moghaddam, and Mohamed Cheriet

Synchromedia Laboratory for Multimedia Communication in Telepresence, École de technologie supérieure, Montreal (QC), Canada H3C 1K3
Tel.: +1(514)396-8972, Fax: +1(514)396-8595
reza.farrahi@synchromedia.ca, imriss@ieee.org, ffarrahi@synchromedia.ca, mohamed.cheriet@etsmtl.ca

ABSTRACT
A framework for development of segmentation-free optical recognizers of ancient manuscripts, which work free from line, word, and character segmentation, is proposed. The framework introduces a new representation of visual text using the concept of signature patches. These patches which are free from traditional guidelines of text, such as the baseline, are registered to each other using a micro-scale registration method based on the estimation of the active regions using a multilevel classifier, the directional map. Then, an one-dimensional feature vector is extracted from the registered signature patches, named spiral features. The incremental learning process is performed using a sparse representation using a dictionary of spiral feature atoms. The framework is applied to the George Washington database with promising results.

1. INTRODUCTION
Optical recognition of handwritten text of ancient manuscripts is one of the difficult document image processing tasks. Changes in the writing hand, skewness, and also changes in the baseline are some of the key problems. It has as well been observed that, in old manuscripts, the writers have not well distinguished between inter-word spaces and intra-word spaces (inter-letters, for example) [1]. This phenomenon seems to be universal for all the languages [2].

Alphabetic representation of words and phrases in modern languages had a big influence on the direction that document image processing has followed. This can be confirmed by the huge number of work devoted to recognition of characters and letters as a sub-layer of word recognition blocks [3]. However, the trend of Optical Character Recognition (OCR) has imposed several constraints that have an impact on the performance and applicability to ancient manuscripts. For example, segmentation of word images into letter images is one of major requirements which in turn requires accurate line, word and character segmentation of input document image. The character-level segmentation can lead to a deadlock as in some cases a part of a letter is only presented, or two or more letters are fused into each other and created a ligature, etc.

Various approaches have been considered in order to relax the constraints of character-based methods [1, 4, 5, 6, 7, 8]. One direction is to use a primitive- or grapheme-based representation in order to create a grammar for the huge set of possible character shapes based on a limited set of graphemes [1]. However, graphemes require a sub-character segmentation, and this segmentation could be an ill-posed problem because the ground-truth data is at the character level, not at the grapheme level.

Another possible direction is using the whole letter-blocks (connected components or sub-words) as the basic units of representation [2, 4, 5, 9]. The main problem with these approaches, which we call them Optical Shape Recognition (OSR), is the dramatic increase in the number of objects from a few tens (the number of characters and their variations) to a few hundred thousands (the number of possible letter-blocks). However, one possible solution to this curse of classes is to keep the class at the character level, but change the samples (observations) to the letter-block level [4]. In other words, each object (character) is separately learned, in the form of a binary descriptor, on the samples (the letter-blocks), and then all the binary descriptors are combined at the end to retrieve the letters of a letter-block. Usually, a skeleton-based or curve-based representation of the letter-blocks is used in order to reduce their complexity. Although the binary descriptors approach is promising, it is sensitive to the segmentation of letter-blocks and also to the accuracy of their skeletons. For some languages, such as Arabic, Persian, and Urdu languages, which include a well-defined letter-block construction in their script grammar, the OSR approaches are a good choice. However, for Latin-based languages, in which the letter-blocks are more the result of variation in the performance of the writer, the results of OSR could be poor. To address this, we propose a new class of recognizers, which are completely independent of the segmentation process, and therefore, they can be considered as segmentation-free methods.

Before continuing with our proposed framework, it is worth to state three other groups of the state-of-the-art methods. The first one is the class of the line-based recognition methods [6]. Interestingly, these methods are able to avoid the word and character segmentations. However,
they usually extract their features from a sliding window on the line, and this makes them somehow grapheme- or subgrapheme-based methods. At the same time, line-level preprocessing of the input, such as local baseline extraction and correction, and skew correction, is very critical for these methods. The second appealing class is known by its slogan of alphabet soup [7]. In these methods, the characters are not the basic units, but instead are discovered by analysis and integrating the soup of characters suggested by the character detectors incorporated in the method. Similar to the line-based methods, they break many of the limitations of the traditional approaches to the recognition. However, all these methods suffer from being linear in that sense that they follow and learn a sequence of the objects along an imaginary line of text. In contrast, in our proposed framework, this implicit assumption is removed, and spatial relations of observations (which we call signature patches) determine the underlying text. The third class uses a patch representation of the isolated characters from input images [8]. They used a Markov random field (MRF) to learn the labels of a patch based on its neighbors. In order to localize the patches, they use an optimization process along the vertical and horizontal directions which could be computationally very expensive. Also, the method can be sensitive to the variations of the strokes which are common in the handwritten text, especially ancient handwritten texts. Therefore, generalization of the method from printed text to handwritten text cannot be confirmed. In contrast, our proposed method can easily absorb variations in the strokes. This is because of its sampling nature. Instead of creating a rigid and unique representation, the signature patches provide several redundant and nonunique sampled representations which can implicitly absorb the variations.

In our proposed framework, called segmentation-Free Optical Recognition (FOR), the objects are considered to be the letters (can also be generalized to combinations of a few letters). However, the observations, which we call them signature patches, are segmentation-free sample patches taken from the document image.

2. PROBLEM STATEMENT

A set of document images is provided: \( \{ U_\omega \}_{\omega=1}^{n_d} \), where \( n_d \) is the number of document images. Also, the binarized and skeleton of binarized images are available: \( \{ U_{\omega, \text{bin}} \}_{\omega} \) and \( \{ U_{\omega, \text{skl}} \}_{\omega} \). On each image, the region of each character (or ligature, in a generalized form) is identified and assigned to the Unicode of that character. Also, the center of mass of each region is calculated and is referred to as a reference point. The reference points are considered as the ground truth. The goal is to build a solution that can generate the same label (Unicode) as the ground truth at each reference point. The set \( \{ U_\omega \} \) will be divided in training and testing subsets.

The proposed framework estimates the Unicode label around a region on an image by generating and identifying some patches, we refer to as signature patches. The details of definition of signature patches and related operations, such as micro registration, and sparse representation learning are discussed in the following sections.

3. SIGNATURE PATCHES

Signature patches introduced here are designed in order to balance the distribution of information between data and model parts of a solution. In contrast to a pixel-based solution, which has the simplest data (pixel) and most complicated models, a solution at a higher level of data can relax the complexity of the model. At the same time, having data represented at very high levels (graphemes, letters, subwords, or words) despite reducing the complexity of model increases the complexity of data in such a way that an embedded solution with its own data and model may be required in order to overcome the complexity of data.

In our approach, the signature patches are placed at an intermediate level to provide a balance between data and model. At the same time, representation of data in the form of signature patches enables us to relax many of constrains of other forms of representation (letter-level representation, for example). Also, its model complexity is far below the complexity of other simple representations (pixel-wise or wavelet representation) which makes signature representation practical.

In the signature-patches representation, at each moment, a region of interest \( u_{\text{ROI}} \) is processed. The \( u_{\text{ROI}} \) is represented by a set of highly overlapping patches which remember their location on the \( u_{\text{ROI}} \). In the case of text recognition, as the background-hosted patches do not carry significant amount of information, they are ignored. In other words, only those patches that anchor on the skeleton pixels (\( u_{\text{ROI}, \text{skl}} \)) are permitted. The size of patches can be determined by the manuscript parameters, such as the average stroke width \( w_s \), the average line height \( h_t \) and the average text height \( h_t \) [10]. In this work, we use the following equation to determine the size of signature patches:

\[
 w_{\text{sig}} = \left\lceil \frac{h_t}{4} \right\rceil
\]

(1)

where \( w_{\text{sig}} \) is the size of signature patches and \( \lceil \cdot \rceil \) stands for the Ceiling function. The number of patches are determined based on the number of text pixels given by the \( u_{\text{ROI}, \text{bin}} \). The number of signature patches \( n_{u_{\text{ROI}}, \text{sig}} \) is calculated as follows:

\[
 n_{u_{\text{ROI}}, \text{sig}} = \left\lceil \frac{n_{u_{\text{ROI}, \text{bin}}}}{2 w_{\text{sig}} w_s} \right\rceil + 1
\]

(2)

A set of signature patches of \( u_{\text{ROI}} \) is denoted by \( \{ P_{x_t, w_{\text{sig}}} \} \), where \( P_{x_t, w_{\text{sig}}} \) is a patch of size \( (2w_{\text{sig}} + 1)^2 \) centered at pixel \( x_t \) of \( u_{\text{ROI}} \). It is worth noting that if some part of \( P_{x_t, w_{\text{sig}}} \) goes beyond the area of \( u_{\text{ROI}} \) that area is added to \( P_{x_t, w_{\text{sig}}} \), and no cropping is employed.

The positions of central pixels \( \{ x_t \} \) are determined in an optimization process which tries to minimize the overlapping area of the signature patches. The following cost function is used to determine an optimal set of central pix-
els \{x_i\}_{i=1}^{n_{uROI}}:\)

\[
x_i = \underset{x_j \in \Omega_{uROI} \land \text{OV}(P_{x_i}, P_{x_j}) = 1}{\arg \max} \sum_{i,j} \left( \text{OV}(P_{x_i}, P_{x_j})^2 \right)
\]

where OV\((P, P')\) gives the overlapping area of two patches \(P\) and \(P'\). The optimal pixels are obtained using a heuristic optimization method. It is worth noting that, in contrast to many applications, although we look for an optimal solution, we do not seek for the global optimal solution. In other words, the set of signature patches of a single \(u_{ROI}\) is not considered to be unique, and several sets are possible. A small displacement of a set of optimal \(x_i\) pixels could provide another solution with almost the same level of optimality. When an instance of a \(u_{ROI}\) is processed, a single set of these patches obtained with optimizing equation (3) is used. If \(u_{ROI}\) is processed another time, another set generated by solving equation (3) may be obtained. As stated before, \(u_{ROI}\) are selected randomly in order to generate as many as possible signature representations. Therefore, a \(u_{ROI}\) will be selected several times during the training process. This nature of the framework enables it to generate an over-complete and redundant collection of samples to be incrementally learned later in the sparse representation section. A sample set of signature patches for a \(u_{ROI}\) is shown in Figures 1 and 2.

![Figure 1](Image 67x171 to 276x242)

Fig. 1. A sample of signature patches generated for a \(u_{ROI}\). The patches are shown as red squares. The segmentation-free nature of the framework can easily be seen from the figure as parts of two word are being processed in one \(u_{ROI}\).

![Figure 2](Image 67x338 to 276x396)

Fig. 2. The set of signature patches extracted from the Figure 1.

4. CORE OF SIGNATURE PATCHES

The nature of document images, which are macro sets of pixels with spatial relations at micro scale (a scale of a few pixels), require registration of a large number of observations at the scale of a few pixels. This can be seen as the implicit drive to use various guiding lines, such as baseline and corpus line, and also to use line and word segmentation. Although our proposed framework is segmentation-free by definition, it also requires a registration between its observations (the signature patches). In other words, the signature patches extracted from the same location on the document image can be shifted a few pixels with respect to another set of signature patches extracted from the same location.

In order to resolve this problem, each signature patch is registered in such a way that its center of activities is at the middle of its patch. The new patch obtained after this translation transform is called its core. In order to determine the center of activities of a patch, we introduce and use the directional map (DM) which is a multi-level classifier [11]. By definition, the directional map of an image has a value between 0 and 1 on the skeleton pixels of that image, and is Not A Number (NaN) otherwise. The 0 and 1 values correspond to \(-\pi/2\) and \(\pi/2\) orientations of the stroke respectively. An example of the DM is shown in Figure 3(b).

![Figure 3](Image 316x253 to 429x366)

Fig. 3. a) A sample signature patch. b) The DM map. c) The standard deviation map of (b). d) The core of (a).

Having the DM of a signature patch, the standard deviation of the DM is calculated using the grid-based modeling [10] on a scale of the \(w_s\). Then, the center of mass (CM) of this standard deviation map is calculated and used as the center of activities of the patch:

\[
x_{i,CM} = CM \left( \sigma_{G;w_s;P_{i,DM}} + 0.1 \max \left( \sigma_{G;w_s;P_{i,DM}} \right) P_{i,SKL} \right)
\]

where \(\sigma_{G;w_s;P}\) is the standard deviation map calculated using a grid of scale \(w_s\), \(CM(\cdot)\) is the center of mass function. The second term in equation (4) is added for numerical stability in cases when the standard deviation is very small. The transform which brings this CM to the center...
of the patch gives us the core. If the standard deviation is zero on the patch, the CM of the skeleton image will be used. The calculated standard deviation distribution of Figure 3(b) is shown in Figure 3(c). By transforming \( x_i, \text{CM} \) to the center of the patch, the core patch is obtained (Figure 3(d)).

The core patches of the signature patches of Figure 2 are shown in Figure 4.

![Fig. 4. The cores of the signature patches in Figure 2.](image)

### 5. SPIRAL FEATURES

In this section, the spiral features are introduced. A patch is converted to its spiral feature vector by arranging its pixels in a clockwise spiral way starting from its central pixel. An example of how the spiral feature vector of an imaginary patch is constructed is shown in Figure 5. The arrangement can be seen as a sequence of moves along horizontal and vertical directions. The corresponding feature vector of a signature patch \( P_x \) is represented by \( f_x \) or \( f_i \).

![Fig. 5. A sample of spiral features extraction from a 7 x 7 patch (\( w_{\text{sig}} = 3 \)). A few of the first moves are highlighted in color. Also, the position of each pixel in the spiral feature vector is stated on each pixel.](image)

The spiral feature vector has many advantages: First, it adds an importance weight to the pixels (for example, setting higher importance for the central pixels). In this work, this option is not used, and all pixels have the same weight. Second advantage of the spiral features is ability to compare two patches of different sizes. This can simply be performed by comparing just the same length of two spiral feature vectors (which will automatically be the length of the shorter vector). As the pixels at the end of feature vector correspond to the pixels far from the center, the amount of information carried by them would be minimal.

In the case of non-square patches, if a move is not possible, that move will be skipped and next move will be performed. However, in some applications, imaginary moves can also be considered.

Having the spiral feature vectors and their labels (obtained based on the reference points and character regions of the ground truth; see section 6.1), the learning process is discussed in the next section.

### 6. SPARSE REPRESENTATION USING DICTIONARY ATOMS

This section briefly reviews the learning process used in the proposed framework. Because of the nature of the signature patches which are large in number, and also are generated gradually by observation, a sparse representation with incremental learning is chosen. A sparse representation is a representation in which a signature patch is represented by a linear combination, with a small number of non-zero coefficients, of some elementary patches called atoms. An over-complete dictionary comprises the atoms. The dictionary is over-complete if the number of its atoms is higher than the minimal dimension required to represent all possible patches. This is usually the case because, not only, the minimal dimension is much less than the dimension of the feature vector space, it is also unknown.

In the proposed sparse representation, the model is constructed by a dictionary of spiral feature vectors. The dictionary is learned in an unsupervised process. Every feature vector in the dictionary, which is an observed feature vectors selected to be in the dictionary, is called an dictionary atom, for example, \( a_j \) where \( j \) is its index in the dictionary. In each observation iteration, a ROI region on the document images is selected, and a set of signature patches is generated. Then, the spiral feature vectors of cores of those signature patches are considered as new observations. The dictionary size \( n_{\text{dic}} \), which is variable and changes over time, is determined by the \( L_1 \) distance (taxi-cab distance) between the union set of already-chosen dictionary atoms and the new observations. The \( n_{\text{dic}} \) is determined by considering a threshold distance of \( d_{\text{thr}} = 2 w_{\text{sig}} \). If the number of those new observations that have a distance to their nearest neighbor in the union set greater than \( d_{\text{thr}} \), then the size of dictionary will be:

\[
 n_{\text{dic}} = \tilde{n}_{\text{dic}} + n_{\text{union}}^{(+)}
\]

where \( \tilde{n}_{\text{dic}} \) is the old dictionary size. In an optimization process, based on the calculated dictionary size, a new set of dictionary atoms is selected from the union set:

\[
 \{a_j\}_{j=1}^{n_{\text{dic}}} = \arg \min \sum_i ||a_i - f_i||_1 \\
 \{a_k\} \subset \{f_i\} \in \{m\} = \bigcup \{a_j\} \\
 \{a_k\} \subset \{f_i\} \in \{m\} = \bigcup \{a_j\} \\
\]

![Image 67x568 to 276x639](image)
where \( \{ \tilde{a}_j \}_j \) is the set of old atoms, \( \{ f_i \}_i \) is the set of new observations, \( | \cdot |_1 \) is the \( L_1 \) norm, \( \bigcup \) is the union operator, \( n_{\text{dic}} \) is the new size of the dictionary, and \( a_l \) is the nearest neighbor of \( f_l \) in \( \{ a_k \} \).

The incremental learning is achieved by selecting random \( u_{\text{ROI}} \) regions from the training document images, and building the dictionary. The optimization process is performed using a heuristic optimizer which take advantage of the curvature of the feature space [12]. As can be seen, the learning process from each set of observations is highly independent from the others. Therefore, a parallel implementation is used (\( n_{\text{para}} = 4 \) parallel processes in our experiments). Each process grows its dictionary based on its observations independently from the other processes for a number of iterations (100 in our experiments), and then all independently-grown dictionaries are combined in a common dictionary in an optimization process similar to equation (6):

\[
\{ a_j \}_j^{n_{\text{dic}}} = \arg\min_{\{ a_k \}} \sum_i | | a_l - f_l | |_1 \quad (7)
\]

where \( n_{\text{para}} \) is the number of parallel processes, \( \{ a_j^{(p)} \}_j \) is the dictionary of \( p^{th} \) process. Then, the common dictionary is distributed to the processes.

The second part of the learning process, which is performed simultaneously with the first part described above, is assigning the Unicode labels to the dictionary atoms. For each dictionary atom, two vectors are considered: one to represent its associated Unicode labels, and the second one to carry the weight of each associated label. In our experiments, we consider at most 8 labels for each atom. In each iteration, a new observation gets some labels based on the reference points and character regions near it (see section 6.1). Then, these labels are propagated to its associated dictionary atom by increasing the weights of those labels in the atom’s weight vector. Iteratively, the label vector of each atom and its weight vector grow and converge to some steady profiles. For those atoms which disappear in an iteration, the corresponding labels and weights are transferred to the new atom. In the testing step, where the labels of the new observations is required in the absence of reference points and character regions, the labels of the corresponding dictionary atoms of the new observation and its spatial neighbors are weighted averaged to obtain the labels of different ranks. The weights of the atoms are normalized to their max before averaging in order to reduce the effect of imbalanced training data.

In the following subsection, the process of generating observation is discussed.

### 6.1. Generating the observations

Generating a set of observations, which is an \( u_{\text{ROI}} \) region and a set of its signature patches, is the core of the proposed framework. We use the following process to achieve this. First, one of the reference points and its nearest neighbor reference points are considered (the red and green marks in Figure 6(a)). Then, using the character regions of these points, several boundary points are extracted with a distance of a fraction of \( u_s \) to each other on the edge of each character region (blue points in Figure 6(b)). As can be seen, these points are displaced toward their reference point in order to avoid ambiguity on the borders of character regions. Next, the bounding box of all these points is constructed and considered as an \( u_{\text{ROI}} \) (yellow box in Figure 6(b)). To have a complete set of reference points and boundary points, all reference and boundary points which are in the bounding box are added to the ground truth of the \( u_{\text{ROI}} \) (purple points in Figure 6(c)).

In the training process, when a set of the signature patches of the \( u_{\text{ROI}} \) are generated, their labels are determined using a nearest neighborhood classification process with 3 neighbors. In contrast, in the testing process, the labels of the dictionary atoms will determine the labels of the signature patches in an averaging process. This can be seen as a coloring process of the document image using the signature patches. The label at each reference point is then determined according to the signature patches that cover it.

### 6.2. Application to unannotated documents

Having the dictionary learned, and the weights and labels of the atoms are set, the method can be used to recognize any unannotated document. In a serial or random way, a series of observations is generated on the unannotated document image in the form of bounding boxes. Then, for each observation, its signature patches and their associated spiral features are generated and mapped on the dictionary. Then, having the associated labels of each signature patch, the final label of every signature patch is determined using a weighted average of its labels and also the labels of its neighbor signature patches. Having the final labels, the area of each signature patch on the observation, and also on the main document image, is painted with its final label. In order to generate the final text, these painted regions should be sequenced using some a priori information on the stroke width and text height and width. This task is beyond this work and will be pursued in the future.

### 7. EXPERIMENTAL RESULTS

The proposed framework was applied to some parts of the George Washington database (DWDB). This database comprises of 20 handwritten document images corresponding to George Washington Papers at the Library of Congress, 1741-1799: Series 2, Letterbook 1, pages 270–279 and 300–309.\(^1\)

\(^1\)Available online with transcriptions at: http://memory.loc.gov/ammem/gwhtml/gwseries2.html
Fig. 6. An example of generating an $u_{ROI}$. a) The reference points. b) The bounding box. c) The reference and boundary points which are inside (b).


Although the transcript of the document images in the database is available, their regional annotation has been performed by manually coloring a segmented version of these document images. Roughly, each character is colored by its character label. Then, in an automatic way, the reference points and boundary points are extracted from the character regions. Up to now, two pages of the GWDB database have been annotated.

7.2. Results

In the experiments, the first page is considered as the training set, and three first lines of the second page are considered as the test set. After completion of the annotation process, the training and test sets will be adjusted accordingly. The training process resulted in a dictionary with a converging size of $n_{dic} = 25242$ atoms. The distribution of the labels and their weights in the atoms label vectors are shown in Figures 7 and 8. Although the second and more ranked labels seems to be ignorable, they carry significant information about the neighbor patches. This can be seen from the high-frequent labels at each rank presented in Figure 9; this shows that there are some atoms which carry information up to their 6th ranked labels.

The current dictionary is used to retrieve the labels of the test set. The same procedure is applied to generate the $u_{ROI}$ regions. In order to reduce the impact of ROI generation step, the reference points obtained from the annotation process are used to select ROIs. Despite the small size of the training data, the proposed method was able to achieve an average accuracy of 48.3% in retrieval of labels at the reference points of the test set. The labels at the reference points of the test set are obtained by reading the painted label, which painted as described in section 6.2, on the document image at the coordinates of the reference point. In the future, a process will extract the individual characters just by analyzing the painted labels and without using the reference points. An example of successful retrieval of all labels at the reference points in an $u_{ROI}$ is shown in Figure 10. The authors are working toward training on the whole set in order to achieve higher accuracies.

Fig. 7. The distribution of labels for the dictionary atoms.

Fig. 8. The distribution weights for the dictionary atoms.
8. CONCLUSIONS AND FUTURE PROSPECTS

A novel framework for segmentation-free optical recognition (FOR) is introduced. The framework is built on the signature patches, micro registration, and sparse learning. It has been applied to a portion of the George Washington database with promising results. Application to the whole database is under consideration. Also, the spatial relations between the signature patches will explicitly be incorporated in the framework.

Another direction is to improving the time performance of the learning process by considering distributed (partial) computing in addition to parallel computing in the case of large databases. Also, preprocessing of the document images, for example, application of shock filter to reduce the text intensity variations will be considered. Analysis of the dictionary atoms in order to discover possible (over-complete) basis in order to reduce the size of dictionary is another research direction. Finally, application of manifold learning techniques in order to reduce the complexity of the representation (the spiral feature vectors) by non-linear dimension reduction will be considered.

9. REFERENCES


