Automatic Single Page-based Algorithms for Medieval Manuscript Analysis

Ying Yang, Yale University, USA
Ruggero Pintus, CRS4, Italy
Enrico Gobbetti, CRS4, Italy
Holly Rushmeier, Yale University, USA

We propose three automatic algorithms for analyzing digitized medieval manuscripts: *text block computation*, *text line segmentation* and *special component extraction*, by taking advantage of previous clustering algorithms and a template matching technique. These three methods are completely automatic, so that no user intervention or input is required to make them work. Moreover, they are all per-page based; that is, unlike some prior methods–which need a set of pages from the same manuscript for training purposes–they are able to analyze a single page without requiring any additional pages for input, eliminating the need for training on additional pages with similar layout. We extensively evaluated the algorithms on 1771 images of pages of 6 different publicly available historical manuscripts, which differ significantly from each other in terms of layout structure, acquisition resolution, and writing style, etc. The experimental results indicate that they are able to achieve very satisfactory performance, i.e., the average precision and recall values obtained by the *text block computation method* can reach as high as 98% and 99%, respectively.

Categories and Subject Descriptors: I.7.5 [**Document and Text Processing**] Document analysis; I.7.5 [**Document and Text Processing**] Graphics recognition and interpretation

Additional Key Words and Phrases: Document layout analysis, medieval manuscripts, text block computation, text line segmentation, logical component extraction

ACM Reference Format:

Ying Yang, Ruggero Pintus, Enrico Gobbetti and Holly Rushmeier. 2015. Automatic Single Page-based Algorithms for Medieval Manuscript Analysis. *ACM JOCCH* *, *, Article * (September 2016), 22 pages.
DOI: 0000001.0000001

1. INTRODUCTION

Over recent years, a large number of historical manuscripts have been digitized and made public, successfully building numerous digital libraries all over the world. For massive collections, there is a pressing need for automatic computer-aided techniques that are able to perform prompt and intelligent document analysis in order to extract various types of information [Pintus et al. 2015], such as text lines and capital letters. Given the extracted information of interest, scholars can then carry out their manuscript studies more efficiently and in greater depth. While manual or semi-automatic techniques can be used for performing analysis on smaller datasets, once the datasets reach a certain

This work was supported by the Digitally Enabled Scholarship with Medieval Manuscripts (DESMM) project funded by the Mellon Foundation. This work was partially supported by the Sardinian Regional Authorities under project VIGEC. Author's address: Y. Yang, H. Rushmeier, Yale University, Department of Computer Science, 51 Prospect Street, New Haven, CT, USA, 06511; email: ying.yang.yy368@yale.edu, holly.rushmeier@yale.edu; R. Pintus, E. Gobbetti; Visual Computing Group, CRS4, Sardegna Ricerche Edificio 1, C.P. 25, 09010 Pula (CA), Italy; email: ruggero.pintus@gmail.com, gobbetti@crs4.it.

DOI: 0000001.0000001

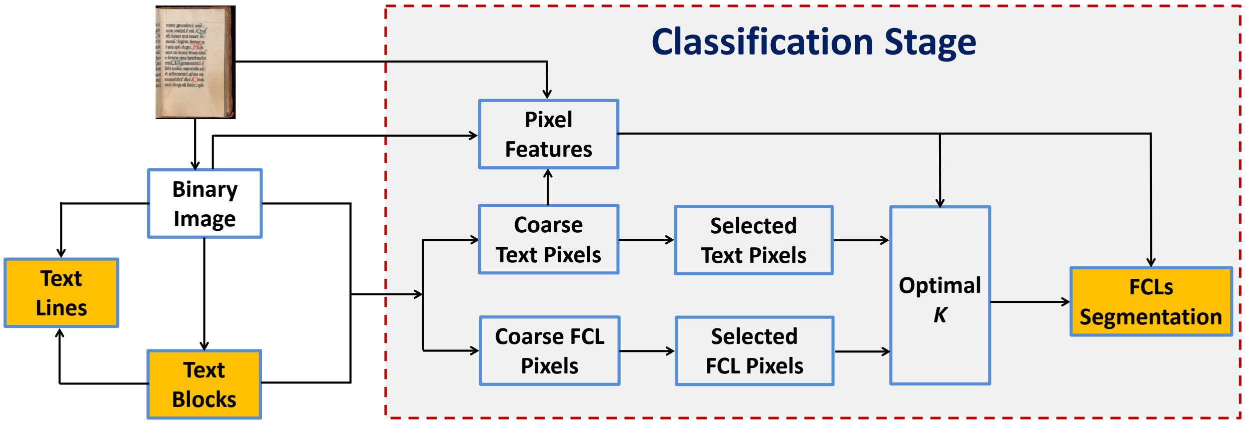

Fig. 1. Algorithmic pipeline. Given an image, we present three automatic algorithms: *text block computation*, *text line segmentation* and *FCLs extraction* for it, based on using template matching and clustering techniques. Manuscript images courtesy of the Yale University [BeineckeMS10].

size these techniques become unfeasibly expensive in terms of both time and labor. For instance, when dealing with large-scale datasets that contain a considerable degree of variability in manuscript physical structure, non-automatic techniques are not as desirable since, in such a context, they usually require distinctive parameter settings for producing good results.

Although some automatic methods [Chen et al. 2014] [Pintus et al. 2014] for analyzing medieval manuscripts have recently been proposed, they can only work on a per-book/manuscript basis. In other words, they require the availability of multiple pages from the same manuscript in order to train a manuscript-dependent classifier. The dependency issue could restrict their range of applicability, i.e., they, when applied to deal with a database that contains pages of structure-distinctive manuscripts, will likely fail due to the difficulty of obtaining a generalized classifier that is reasonable for all the data in the dataset. Therefore, algorithms that can work on a per-page basis are in high demand.

Two complicating factors are (i) that medieval manuscripts generally have sophisticated physical structures, such as flexible writing style and holes; and (ii) that they have undergone significant degradation, due to aging, frequent handling and storage conditions. Thus, it is generally more challenging to design an algorithm for analyzing medieval manuscripts than modern machine-printed documents. Expectedly, the methods [Park et al. 2001] specially designed for modern machine-printed documents are unlikely to produce reasonable results when applied to historical manuscripts [Pintus et al. 2015]. Despite potential difficulties, attempts have been made towards developing computer-assisted techniques for layout analysis of medieval manuscripts. These previous works concentrate mainly on word spotting [Rath and Manmatha 2003], word segmentation [Louloudis et al. 2009] and text line extraction [Pintus et al. 2015].

We develop three fully automatic, per-page based algorithms for analyzing medieval manuscripts: (i) *text block computation*, (ii) *text line segmentation* and (iii) *special component extraction*. By "medieval manuscripts" we refer to professionally prepared books prior to the advent of mechanical printing. Such books were prepared by professionals, who as a step in manuscript preparation ruled the parchment before writing, so that they generally have regular layouts and stable features [De Hamel 1992]. Thousands and thousands of such books professional produced by hand have survived and are an object of study by scholars. For the Book of Hours, which we use for our tests, there are at least 800 surviving copies. Scholars, such as the scholar we worked with, are interested in finding variations in these book copies that convey individuality in their production [Brantley 2009].

The algorithmic pipeline is illustrated in Fig. 1. Note that by special components, we mean those semantically meaningful elements that are not text. Since the special components in our test dataset are almost solely figures and capital letters (see Fig. 9), we shall abbreviate them as FCLs. The text blocks and lines are extracted based on analyzing a projection profile derived from its corresponding binary image. Although the use of binary images likely results in weak robustness against factors such as noise, our methods cope with this limitation by using the reliable text leading/height [Pintus et al. 2015] as the a priori knowledge about the page's physical structure. As demonstrated by our experiments on different manuscripts, the presented approaches can achieve satisfactory robustness. Similar to previous methods [Chen et al. 2014; Grana et al. 2009], we also formulate the extraction of FCLs as a clustering problem, but with two main distinctions. First, we utilize both *unsupervised* and *supervised* learning algorithms for improved performance, while prior methods often take into account *supervised* learning only. Second, the proposed algorithm is a per-page based algorithm that can carry out single page-based training, which is implemented through a novel conversion that transforms the outputs of unsupervised learning into the inputs of supervised learning. By contrast, to the best of our knowledge, prior methods perform multiple-page based training, which requires information contained in few pages from the same manuscript during the training process. As such, they fail to produce reasonable results when working on a database of digital images from multiple distinct books.

The current paper is a significantly extended version of our previous work on color analysis [Yang et al. 2015]. We in this paper use similar image template matching idea presented in [Yang et al. 2015] to identify text pixels of a given page image and also use the same constraints described in [Yang et al. 2015] to determine if an FCL candidate is a *real* or *valid* FCL. The new material includes: (i) text block computation; (ii) text line segmentation; and (iii) new approach to identifying FCLs. In sum, the main contributions of this paper are summarized as follows:

—Three automatic, per-page based algorithms for analyzing medieval manuscripts.

—A demonstration of how to combine *unsupervised* and *supervised* learning algorithms properly for classification purposes.

—Extensive evaluation of our proposed algorithms on a dataset of 1771 images of pages of 6 structure-distinctive medieval manuscripts.

Overall, the purpose of our paper is to create a reliable framework for performing document layout analysis on medieval manuscripts. Although some assumptions regarding medieval text height/width are made, the framework is highly modular, and some steps are independent of the assumptions so that it is adaptable to other writing styles by, for example, finding a new assumption for a particular type of manuscripts.

The rest of this paper is organized as follows. Section 2 reviews relevant literature. Sections 3 and 4 cover the extraction of text blocks and text lines respectively. The algorithm for localizing and extracting FCL is described in detail in Section 5. The experimental results are presented and discussed in Section 6, and we give a brief conclusion in Section 7. In order to aid in clarity, we have included the frequently used symbols in this paper in Table I.

2. RELATED WORK

There are many excellent works on analyzing historical documents and they focus on various aspects of analysis, such as word matching [Rath and Manmatha 2003] [Rodríguez-Serrano and Perronnin 2009], word segmentation [Manmatha and Rothfeder 2005] [Louloudis et al. 2009], text line extraction [Pintus et al. 2015] [Louloudis et al. 2008] [Saabni et al. 2014] and figure extraction [Grana et al. 2009].

Table I. Frequently used symbols.

Symbol	Meaning
H	text height/leading
W	text width
K	number of colors/clusters
$\mathbf{B}$	binary image
$\mathbf{S}$	matrix of original matching scores
$\mathbf{S}'$	matrix of updated matching scores
$\lceil\cdot\rceil$	ceiling function
$\|x\|$	absolute value of x

For concision we will only review the most relevant here. For a more exhaustive comparison of layout analysis algorithms, we refer the reader to recent surveys and contests [Nagy 2000] [Likforman-Sulem et al. 2007] [Antonacopoulos et al. 2009] [Stamatopoulos et al. 2013].

Text Line Extraction. Projection profiles have been extensively used for extracting text lines. Manmatha et al. [Manmatha and Rothfeder 2005] compute the projection profile by summing up the pixel values line-by-line. Next, the profile is smoothed with a Gaussian filter to reduce noise sensitivity. Finally, text lines are found by detecting the peaks of the smoothed profile. Arivazhagan et al. [Arivazhagan et al. 2007] propose a skew-resistant method, where an initial set of candidate lines is obtained from the piece-wise projection profile of the input image, and the lines traverse around any obstructing handwritten connected component by associating it with the line above or below. However, there are many problems that are extremely common with these project profile based methods, such as a high sensitivity to noise, inconsistent inter-line spacing, and text-line skew variability.

The Hough transform has also been successfully exploited for text line segmentation [Louloudis et al. 2008] [Likforman-Sulem et al. 1995] [Fletcher and Kasturi 1988]. Likforman-Sulem et al. [Likforman-Sulem et al. 1995] propose a hypothesis-validation scheme. That is, they extract the best text line hypothesis in the Hough domain, while checking the validity of the hypothesis in the image domain. Alternatively, Louloudis et al. [Louloudis et al. 2008] employ block-based Hough transform to detect potential text lines.

Image smearing-based algorithms include both the fuzzy [Shi and Govindaraju 2004] and adaptive RLSA (Run Length Smoothing Algorithm) [Nikolaou et al. 2010]. In some approaches, the RLSA measure is computed for every pixel, yielding an RLSA-based grayscale image. Then, this grayscale image is binarized and the text lines are extracted from the binary image. Other text line extraction methods [Pintus et al. 2015] are based on feature training and testing.

Although some advanced algorithms [Shi et al. 2009] [Koo and Cho 2012] [Ryu et al. 2014] have been recently proposed to extract text lines, they focus on dealing with the images that are of high contrast between background and foreground. Consequently, they generally do not work when used to perform text line extraction for images of medieval manuscripts, where the contrast between background and foreground could be quite low.

As compared to our method, some of the existing algorithms focus on different classes of documents. For instance, the work by Garz et al. [2013] deals with documents that have been stored improperly so that the pages are not flattened, while we consider the thousands of medieval manuscripts stored in libraries that are not severely damaged as in the special case given here. Bar-Yosef et al. [2009] propose an algorithm that works for documents with large skew, which does not occur for the professionally prepared books we are studying. Arvanitopoulos et al. [2014] study a different class of algorithms for cutting out irregular shapes of text, that are useful for applications such as handwritten correspondence, but are not relevant to the analysis of professionally prepared medieval books.

Our proposed method computes text lines by analyzing projection profiles. Indeed, similar ideas have been used in previous algorithms [Manmatha and Rothfeder 2005], but with two main distinctions. While prior methods work by analyzing whole images, our proposed algorithm analyzes text blocks instead. In addition, we use the reliable text leading [Pintus et al. 2015] as the a priori knowledge about the page physical structure; however, prior algorithms do not generally employ or consider this useful information. It is this distinction that allows for our great success in producing such projection profiles which can be analyzed with less effort and which can therefore generate better results.

Text Block Extraction. In the past, algorithms have been presented which have been capable of text block segmentation. While Jain and Yu [Jain and Yu 1998] concentrate on geometric layout analysis of printed technical journal pages, Baechler et al. [Baechler et al. 2010] describe a semi-automatic tool for historical manuscripts. More recently, Pintus et al. [Pintus et al. 2014] propose a text block extraction method for medieval manuscripts. However, this method is per-book/manuscript based, requiring the availability of a set of pages from the same manuscript in order to train a classifier for identifying text pixels.

While Asi et al. [2014] treat a problem in variation in text layout (i.e. writing in curved lines in varies blocks around main text blocks) in ancient (not medieval) texts that do not occur in the class of professionally prepared medieval books that we study, Shafait et al. [2008] consider the segmentation used by various OCR methods applied to mechanically printed books. The problems considered in [Mehri et al. 2014] are not present in the class of documents we consider. As we can see, none of these papers is dealing with the same type of documents or the same problem we are interested in the paper.

Motivated by [Yang et al. 2015], we utilize a template matching technique to obtain texts and proceed to extract text blocks by finding connected components. Our proposed text block extraction method works on a per-page basis and as such requires only the page being analyzed as the input, with no reference to other supporting pages.

Special Component Extraction. Extracting or identifying special, non-text components from historical manuscripts is becoming an increasingly important aspect in contemporary document analysis. This is generally considered as a clustering/segmentation problem, where each image pixel is classified into one of the pre-defined groups such as text, background or decoration [Chen et al. 2014].

Chen et al. [Chen et al. 2014] develop a layout structure segmentation algorithm, where each pixel is represented by a vector containing features based on the coordinates, color and texture information gathered from the area surrounding the pixel. By taking advantage of the SVM (Support Vector Machine), Grana et al. [Grana et al. 2009] propose a system for automatically extracting graphical elements from historical manuscripts and then identifying significant pictures from them. Yang et al. [Yang et al. 2015] introduce an algorithm for automatically estimating a reasonable number of clusters and demonstrate the importance of using the cluster number in FCL extraction.

A common issue among methods which employ *supervised* learning techniques is again that they work on a per-book basis and consequently have limited applicability. Following [Yang et al. 2015], we propose a per-page based algorithm to efficiently address this problem. The proposed method differs from [Yang et al. 2015] in two aspects. First, we modify the K computation strategy so that both k-means and EM algorithm are used, while the original method uses either individually. Second, while [Yang et al. 2015] only takes into account *unsupervised* learning, our method reasonably incorporates both *unsupervised* and *supervised* learning techniques. As demonstrated by the experimental results, these improvements result in better classification performance.

3. TEXT BLOCK COMPUTATION

As Fig. 2 illustrates, we propose a two-stage procedure for extracting the text blocks. In the first stage, we obtain rough text blocks by analyzing projection profiles. In the second stage we perform text block

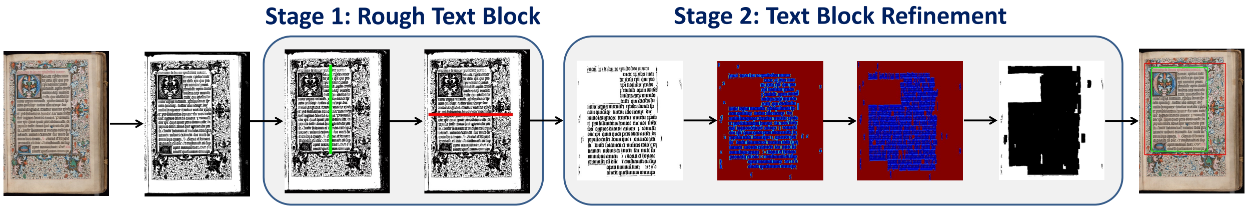

Fig. 2. Two-stage text block computation pipeline. Firstly, we analyze the project profiles to yield the rough text block, whose vertical and horizontal ranges are highlighted by green and red lines, respectively. Second, we propose a refinement strategy to remove unwanted non-text regions from the obtained rough block and to produce a final text block. The rightmost figure shows the text blocks before (red) and after (green) refinement. Manuscript images courtesy of the Yale University [BeineckeMS310].

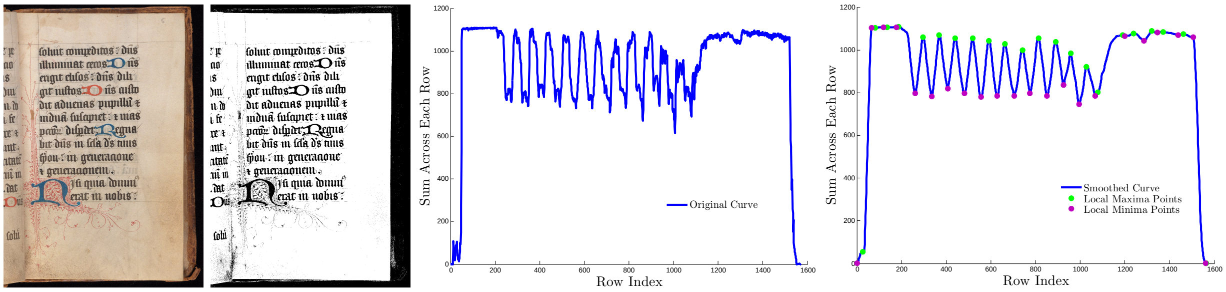

Fig. 3. From left to right: original image, binary image, original projection profile obtained from summing up pixel values along rows and smoothed projection profile. Manuscript images courtesy of the Yale University [BeineckeMS10].

refinement by taking full advantage of a template matching technique to remove unwanted non-text regions from the text blocks obtained, which yields better blocks.

Here we assume that a *valid* text block always satisfies the following constraint: its height and width must be larger than $2 \cdot H$ and $\lceil 1/4 \cdot W_{\text{im}} \rceil$, respectively. Here, H and W_{im} represent the text leading and image width, respectively. We compute H using the ATHENA [Pintus et al. 2013].

3.1 Stage I: Rough Text Blocks

Given a color image, we compute its corresponding binary matrix $\mathbf{B}$ using the image binarization method presented in [Yang et al. 2015]. This method performs image binarization under the assumption that there are more *background* pixels than *foreground* ones. Generally, the projection profile derived from summing along rows of $\mathbf{B}$ differs significantly in shape for *text* and *non-text* regions. Taking into account this fact, we obtain a smoothed projection profile via performing the unweighted moving average using adjacent $\lceil H/2 \rceil$ neighbors of each profile point on the original profile (see Fig. 3).

Next, we compute the local maximum and minimum points of the smoothed projection profile, subject to the constraint that the minimum distance between any two local maximum or minimum points must be larger than or equal to $\alpha \cdot H$. The observation is that the minimum points likely correspond to text areas, while the image segment between any two adjacent maximum/minimum points is generally the text lines themselves. As such, we put a constraint on the distance between two adjacent maximum points. That is, it should approximately equal to the text height, H. For conservative purposes, α is introduced to relax the constraint; $\alpha = 0.7$ was used in our tests.

Then, let r_i denote the row index of the i-th minimum point, and r_i^- and r_i^+ the row indices of its 2-nearest maximum points such that $r_i^- < r_i < r_i^+$. We can quantitatively describe the above observation and profile shape difference using the ratio defined as follows:

$$d_i = \max\,\{f(r_i)/f(r_i^-),\ f(r_i)/f(r_i^+)\}, \tag{1}$$

where $f(x)$ indicates the value at the point x of the profile. We fix $d_i = 1$ if $r_i^- < r_i < r_i^+$ does not exist. Indeed, smaller d_i corresponds to the regions where text appears (see Fig. 3). The goal now becomes extracting a subset of ratios that corresponds to the text regions. To do so, we apply the k-means++ algorithm [Arthur and Vassilvitskii 2007] ($k = 2$ in this paper) to the ratio set $\{d_1, d_2, ...\}$, yielding k clusters $\mathcal{C}_1$, $\mathcal{C}_2$, ..., $\mathcal{C}_k$. As mentioned earlier that we expect small d_i for the text areas, these areas should hence correspond to the cluster $\mathcal{C}_{k^*}$ given by

$$\begin{array}{ll} \underset{\mathcal{C}_j}{\text{minimize}} & \dfrac{1}{\|\mathcal{C}_j\|} \cdot \displaystyle\sum_{d_i \in \mathcal{C}_j} d_i \\ \text{subject to} & \displaystyle\max_{d_i \in \mathcal{C}_j} \{f^{-1}(d_i)\} - \min_{d_i \in \mathcal{C}_j} \{f^{-1}(d_i)\} > 2 \cdot H \end{array}, \tag{2}$$

where $\|x\|$ stands for the number of elements in a set x and $f^{-1}(x)$ returns the row index associated with x. The constraint in Eq. 2 reflects our assumption that a text block contains more than two text lines.

Given $\mathcal{C}_{k^*}$, we believe that the text block(s) is/are bounded by

$$\Theta = [\min_{d_i \in \mathcal{C}_{k^*}} \{f^{-1}(d_i)\} - \lceil H/2 \rceil\,,\ \max_{d_i \in \mathcal{C}_{k^*}} \{f^{-1}(d_i)\} + \lceil H/2 \rceil]. \tag{3}$$

Notice that $\pm H/2$ in Eq. 3 is due to the fact that a minimum point is located approximately in the middle of a text line (see Fig. 3).

It is easy to see that the above process just prunes the non-text regions along rows. Our current focus is on removing those unwanted regions along columns. Similarly, a column-based projection profile is first created by summing along columns of the binary matrix of the text block computed before. Assuming the block size is $M \times N$, we can obtain a N-sized binary vector v:

$$v_i = \begin{cases} 0 & \text{if } s_i \geq \lambda \cdot M \\ 1 & \text{otherwise} \end{cases}, \tag{4}$$

using the N sum values $\{s_1, s_2, ..., s_N\}$ along columns. The parameter λ in Eq. 4 is a parameter that distinguishes *text* and *non-text* columns. In other words, if the majority of pixels along the i-th column are foreground (black) pixels, we expect a smaller s_i. Moreover, this column is said to be a *text* column and is associated with $v_i = 1$. The parameter λ is found through an iterative process. Starting from $\lambda = 0.98$, we iteratively reduce λ by 0.004, i.e., $\lambda \leftarrow \lambda - 0.004$, and terminate the iteration until there are more than 10 percent of zeros in the vector v.

After this, we update v using the majority-based voting rule. Specifically, for each component v_i of v, we group its H-nearest neighbors and assign $v_i = 0$ if there are more 0's than 1's in the group; otherwise $v_i = 1$.

Since $v_i = 1$ corresponds to a text column, the problem now turns into finding the consecutive 1's within in the updated vector v. We partition v into smaller series of subvectors of consecutive elements $v_i = 1$. For a specific series, we claim that its corresponding columns constitute a text block if and only if it obeys the assumption mentioned earlier.

The text blocks obtained in the first stage are not perfect; i.e., they may contain non-text regions, especially when the given page contains decorations in the margins (see Fig. 2). Nevertheless, the intention of producing rough blocks and preliminarily removing some unwanted image parts from the

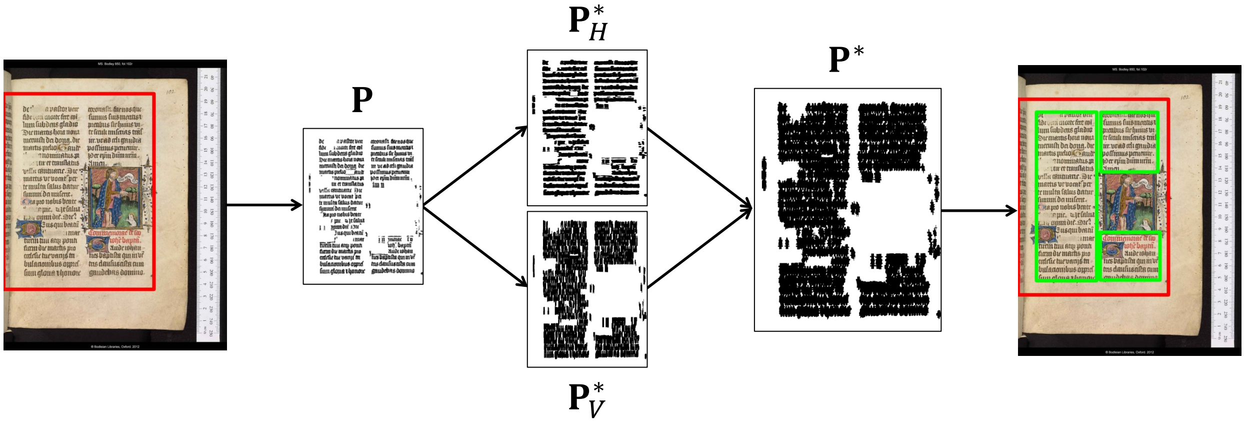

Fig. 4. Illustration of the refinement process for a page with multiple text blocks. Starting from the rough block (highlighted in red), we obtain its binary text mark image $\mathbf{P}$, two variants, $\mathbf{P}^*_H$ and $\mathbf{P}^*_V$, of $\mathbf{P}$ and also $\mathbf{P}^*$. Refined text blocks are computed by extracting connected components from $\mathbf{P}^*$. Manuscript images courtesy of the Oxford University [BodleianMSBodley850].

original is twofold: (i) a better estimate of the stroke width, W, can be computed; and (ii) the speed of the image matching process can be greatly increased.

3.2 Stage II: Text Block Refinement

Motivated by the method presented in [Yang et al. 2015], here we refine the text blocks obtained in the previous subsection under the assumption that the text is rectangularly shaped. This assumption is generally true for old English and Latin manuscripts, although in some cases manuscripts have a highly curved style that will degrade the performance of our method. Fig. 4 shows how the refinement algorithm works for a page with multiple text blocks.

The main idea is that the text of a manuscript page can be identified by using an image template matching technique due to the shape (rectangularly shaped) of text. We refer the reader to [Yang et al. 2015] for more detailed description on the matching process. Briefly, we, given the text height H and text stroke width W, begin with defining a binary text stroke-like template image $\mathbf{T}$:

$$\mathbf{T}(i,j) = \begin{cases} 0 & \text{if} \quad \begin{array}{l} \alpha_H \cdot H \le i \le \beta_H \cdot H \\ \alpha_W \cdot W \le j \le \beta_W \cdot W \end{array} . \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

Here, H and W are computed based on [Pintus et al. 2015] and [Von Gioi et al. 2010], respectively. The two parameters α_W and β_W are fixed at $\alpha_W = 0.5$ and $\beta_W = 1.5$, while the other parameters α_H and β_H are obtained using connected component idea. Then, matching the binary image $\mathbf{B}$ and the template $\mathbf{T}$ yields a matrix $\mathbf{S}$ of matching scores normalized into [0, 1] and furthermore $\mathbf{S}'$ after making some updates to $\mathbf{S}$. Fig. 5 shows the color-coded $\mathbf{S}$ and $\mathbf{S}'$ for an example image.

While Yang et al. [Yang et al. 2015] end up with choosing the pixels that corresponding to $\mathbf{S}'(x,y) < \kappa$ for computing an appropriate number of clusters, we here regard pixels with $\mathbf{S}'(x,y) \ge \kappa$ as text pixels. That is, the binary text mark image $\mathbf{P}$ is given by:

$$\mathbf{P}(x,y) = \begin{cases} 0 & \text{if } \mathbf{S}'(x,y) \ge \kappa \text{ and } \mathbf{B}(x,y) = 0 \\ 1 & \text{otherwise} \end{cases} . \tag{6}$$

κ is a threshold distinguishing text and non-text and is fixed at $\kappa = 0.75$ in [Yang et al. 2015].

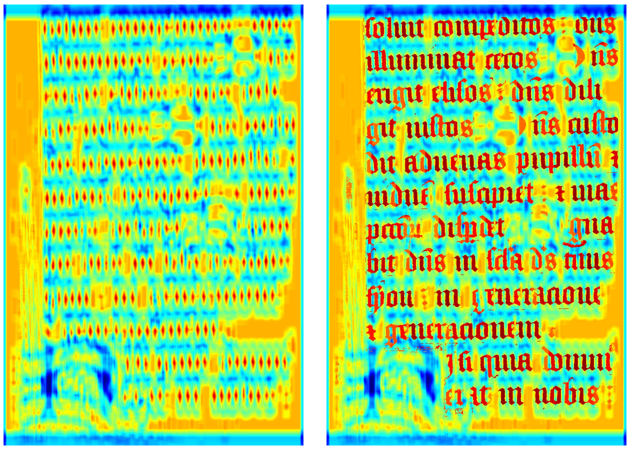

Fig. 5. Visualization of the original matching score map $\mathbf{S}$ and updated map $\mathbf{S}'$ for the text block of Fig. 3 (left). Here, we assign blue and red color to low and high matching scores, respectively.

With $\mathbf{P}$, we can perform text block refinement based on the observation that a text block can actually be considered as a connected component that satisfies certain constraints. Rather than extracting the connected components directly from $\mathbf{P}$, we produce a variant $\mathbf{P}^*$ of it and then work on $\mathbf{P}^*$ instead.

To generate $\mathbf{P}^*$, we compute both horizontal-direction and vertical-direction distance maps of $\mathbf{P}$. The distance map along horizontal direction is defined as follows:

$$\mathbf{D}_H(x,y) = \min_{\substack{y' \\ \mathbf{P}(x,y')=0}} \{|y-y'|\}. \tag{7}$$

We set $\mathbf{D}_H(x,y) = W + H$ if $\mathbf{P}(x,y') = 0, \forall y'$ does not exist. Next, we compare $\mathbf{D}_H$ against the threshold W, resulting in a binary image $\mathbf{P}^*_H$:

$$\mathbf{P}^*_H(x,y) = \begin{cases} 0 & \text{if } \mathbf{D}_H(x,y) \leq W \\ 1 & \text{otherwise} \end{cases}. \tag{8}$$

Similarly, we can obtain the vertical-direction distance map $\mathbf{D}_V$ and furthermore its corresponding binary image $\mathbf{P}^*_V$ with the threshold $H/4$. Fig. 2 shows the color-coded maps for $\mathbf{D}_H$ and $\mathbf{D}_V$. Then, $\mathbf{P}^*$ is produced after performing element-by-element bitwise AND (&) operation between $\mathbf{P}^*_H$ and $\mathbf{P}^*_V$, i.e., $\mathbf{P}^* = \mathbf{P}^*_H \& \mathbf{P}^*_V$.

Finally, we extract all the connected components from $\mathbf{P}^*$ and consider each of them as a *valid* refined text block if it meets the size-based constraints mentioned at the beginning of Section 3.

The intention of generating $\mathbf{P}^*$ is to "close" the gaps between non-connecting text pixels and hence to increase the accuracy of obtaining expected text blocks. In other words, $\mathbf{P}^*_H$ and $\mathbf{P}^*_V$ are two variants of $\mathbf{P}$ produced by adding extra zero-valued pixels into $\mathbf{P}$. This addition attempts to merge certain adjacent, yet isolated pixels or closes their "gaps" along both directions, forming one or few larger connected components in $\mathbf{P}^*$. Regarding the two thresholds used here, they are an indication that what gaps can be filled up with zeros. As an example, considering two foreground pixels at (x_1, y_1) and (x_1, y_2) with

$$y_2 = \arg\max_{\substack{y \\ \mathbf{P}(x_1,y)=0}} \{|y_1 - y|\}, \tag{9}$$

we believe that the two pixels (x_1, y_1) and (x_1, y_2) are unlikely from the same text block if $\mathbf{D}_H(x_1, y_1) > 2 \cdot W$; and thus they should not be connected. The dis-connectivity is implemented because $\mathbf{D}_H(x_1, y) > W$, $\exists y \in [y_1, y_2]$ holds, assuming without loss of generality that $y_1 < y_2$ and hence $\mathbf{P}^*_H(x_1, y) = 1$, $\exists y \in [y_1, y_2]$. Otherwise, $\forall y \in [y_1, y_2]$, $\mathbf{D}_H(x_1, y) \leq W$ and hence $\mathbf{P}^*_H(x_1, y) = 0$.

4. TEXT LINE SEGMENTATION

The extraction of text lines is based on analyzing the text blocks computed in the previous section. Specifically, we extract the text lines from the binary images of the text blocks, rather than from the given whole image. Again, the main observation is that the spacing between any two adjacent text lines results in wave-like fluctuation in the row-based projection profile.

Following the same idea and using the same parameter setting as mentioned in Section 3.1, we sum up the pixel values along rows of the binary image of each text block, smooth these sum values and compute the local maximum points. Afterward, we extract the image regions bounded by any two adjacent maximum points and deem them as text lines.

Although our text line segmentation algorithm is a binary image-based method, we achieve satisfactory performance (precision value of up to 93.20% and 99.62% as shown in Tables IV and V, respectively). This is because it works on a text block basis with the a priori knowledge about the page's physical structure; i.e., the height of each text line segment is expected to be approximately H.

5. FCL EXTRACTION

Next, we move on to the most challenging part, where the focus will be on identifying and extracting FCL, if existing. Since FCL are generally colored and shaped distinctively from text (see Fig. 9), we formulate this as a clustering problem. The algorithmic pipeline is shown in Fig. 1 and the implementation details are described as follows.

5.1 Feature and Optimal K Computation

We use the same features as used in [Yang et al. 2015] and the feature vector for each foreground pixel is composed of 60 components associated with color and statistical characteristics, such as the mean, standard deviation, skewness, energy and entropy of the color information.

As the number of colors, K, varies over pages, we should use content-adaptive K as the number of clusters when using clustering algorithms to classify pixels. Otherwise, with fixed K, the unexpected underfitting and overfitting issues likely occur.

To compute a content-adaptive K, we follow the strategy in [Yang et al. 2015] with some modifications. In [Yang et al. 2015], K is estimated by minimizing a function that assesses the quality of a candidate clustering. That is, the expected/appropriate K should correspond to the best clustering quality. Our proposed method first selects a subset of foreground pixels corresponding to $\mathbf{S}'(x, y) < \kappa$ and clusters them into different groups. Then, the clustering quality is measured using the Davies-Bouldin method [Davies and Bouldin 1979]. Finally, we compute K as

$$K = \left\lceil (K^{\text{kmeans}} + K^{\text{EM}})/2 \right\rceil . \tag{10}$$

where K^{kmeans} and K^{EM} are the optimal cluster number K corresponding to good clustering quality, when using K-means and the EM algorithm for the Gaussian mixture model to perform clustering. K^{kmeans} is estimated as

$$K^{\text{kmeans}} = \begin{cases} \lceil (K^+ + K^-)/2 \rceil + 1 & \text{if } |K^+ - K^-| \leq 3 \\ K^+ + 1 & \text{otherwise} \end{cases} , \tag{11}$$

where K^+ and K^- denote the indices of the first and second minimum elements of $\{DB_{\tilde{K}} : \tilde{K} \in [K_{\min}, K_{\max}]\}$. $K_{\min}$ and $K_{\max}$ define the range of K; that is, the number of colors that one manuscript page generally uses. We fix $K_{\min} = 1$ and $K_{\max} = 7$. $|K^+ - K^-| \leq 3$ in Eq. 11 means that both K^+ and K^- should be reliable due to their small difference and thus should contribute to K^{kmeans}. The computation procedure for its corresponding K^{EM} is analogous, so we omit the details.

Two noticeable changes to the original method presented in [Yang et al. 2015] are (i) that we combine both the k-means and EM clustering algorithms to compute the expected K in Eq. 11, while [Yang et al. 2015] uses each individually and (ii) that we use an improved equation 11 to compute K^{kmeans}.

5.2 Feature Clustering and FCL Extraction

The clustering algorithms used include the k-means, SVM and EM algorithm for the Gaussian mixture model. FCL are extracted according to the binary mask image $\mathbf{I}$, which is derived from combining all the binary mask images $\{\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_8\}$ that are produced based on the results of these clustering methods. Each zero-valued pixel in each mask image corresponds to a unit/element of a FCL. The algorithmic details are described as follows. Note that we shall only consider the *foreground* pixels in the following unless otherwise stated.

Mask Image $\mathbf{I}_1$. To compute the first binary mask image $\mathbf{I}_1$, we apply the k-means algorithm with $k = K$ (computed using Eq. 10) clusters to classify the image pixels, resulting in K groups. However, since the computed K could be inaccurate, the clustering results may be not satisfactory and consequently the cluster labels may have to be amended [Yang et al. 2015]. We update K following [Yang et al. 2015].

To distinguish the text and FCL pixels, we take into account the two observations: (i) generally there are significantly more text pixels than FCL ones and (ii) the text pixels are associated with larger matching scores. Considering the pixels with the matching score $\mathbf{S}'(x,y) \geq \kappa = 0.85$, we respectively count the number of occurrences of each label in $\{1, 2, \ldots, K\}$ for these pixels to produce the count set $\{n_1, n_2, \ldots, n_K\}$. With the observation in mind, we can thus assume that the text has been assigned the label

$$i^* = \arg\max_i \{n_i\}. \tag{12}$$

and hence that the pixels without the label i^* are the elements of FCL.

The binary mask $\mathbf{I}_1$ is produced as follows. We first initialize it with ones. Next, we perform $\mathbf{I}_1(x,y) = 0$ if and only if its label is not i^*, i.e., it belongs to the set $\{1, 2, \ldots, K\} - \{i^*\}$. This way, $\mathbf{I}_1(x,y) = 0$ is expected to correspond to the *FCL* pixels, while $\mathbf{I}_1(x,y) = 1$ is for the *text* and *background* pixels.

Mask Image $\mathbf{I}_2$. The computation process for the second mask image $\mathbf{I}_2$ is exactly the same as for $\mathbf{I}_1$, except that we use the EM algorithm rather than the k-means method.

Mask Image $\mathbf{I}_3$. To generate $\mathbf{I}_3$, we combine both the k-means and SVM clustering algorithms. That is, we employ the clustering results resulting from k-means to assist the SVM training. First, the k-means algorithm with K clusters is first applied to classify the image pixels. Next, we train a SVM classifier with Radial Basis Function (RBF) kernel using the features computed in Section 5.1 and k-means based clustering labels. In our experiments, the *training pixels* include those that correspond to $\mathbf{S}'(x,y) < \kappa = 0.75$ (likely non-text pixels) and half of those pixels (likely text pixels) that have $\mathbf{S}'(x,y) \geq \kappa = 0.85$ and the label i^*. The half pixels are randomly chosen.

After the SVM-based classifier has been computed, we perform pixel classification and subsequently produce $\mathbf{I}_3$, following the strategy as mentioned in the $\mathbf{I}_1$ computation.

Mask Image $\mathbf{I}_4$. Simply replacing the k-means algorithm with the EM algorithm for computing $\mathbf{I}_3$ will yield the fourth binary mask image $\mathbf{I}_4$.

Mask Image $\mathbf{I}_5$. To compute the fifth mask image $\mathbf{I}_5$, we combine the k-means, EM algorithm and SVM together. To train a better SVM classifier, we utilize the first two algorithms with K clusters to identify more reliable training pixels. A pixel is deemed as *reliable* if the two labels from the two unsupervised methods are corresponding to each other. Note that there is no need for the two methods to assign the same label to the pixel, i.e., label consistency is not required.

Since the labels from the two unsupervised algorithms may not be consistent, we first need to compute Σ, which is composed of a set of label correspondences. In other words, the task is to find a permutation of the k-means labels (EM labels) so that the permuted labels correspond to the EM labels (k-means labels). Let $\{n_1^{\text{kmeans}}, n_2^{\text{kmeans}}, \ldots, n_K^{\text{kmeans}}\}$ and $\{n_1^{\text{EM}}, n_2^{\text{EM}}, \ldots, n_K^{\text{EM}}\}$ denote the numbers of occurrences of each k-means and EM label of the *training pixels*, which are obtained using the same way as used for computing $\mathbf{I}_3$. Assuming the k-means label i corresponds to the EM label i', then their occurrences n_i^{kmeans} and $n_{i'}^{\text{EM}}$ are expected to be similar; that is, the difference $|n_i^{\text{kmeans}} - n_{i'}^{\text{EM}}|$ should be small. Computing the differences between any two elements in $\{n_1^{\text{kmeans}}, n_2^{\text{kmeans}}, \ldots, n_K^{\text{kmeans}}\}$ and $\{n_1^{\text{EM}}, n_2^{\text{EM}}, \ldots, n_K^{\text{EM}}\}$ yields a $K \times K$ dissimilarity matrix $\mathbf{M}$

$$\mathbf{M}(i,j) = |n_i^{\text{kmeans}} - n_j^{\text{EM}}|. \tag{13}$$

A good label correspondence is found if the sum of all its dissimilarities is small. In our case, the aim is therefore to find a constrained minimum assignment through column and/or row permutation to minimize the trace of $\mathbf{M}$. Mathematically,

$$\begin{aligned} \text{minimize} \quad & \text{trace}(\mathbf{M}) \\ \text{subject to} \quad & \arg\max_i \{n_i^{\text{kmeans}}\} \leftrightarrow \arg\max_i \{n_i^{\text{EM}}\}, \end{aligned} \tag{14}$$

where $\leftrightarrow$ stands for a correspondence between two labels. The constraint in Eq. 14 reflects our assumption that there are more text pixels in the training set and thus that their labels should correspond to each other. To solve this optimization problem, various optimization algorithms, such as [Stošić et al. 2011], can be used. We in this paper use the Hungarian algorithm [Kuhn 1955], which results in a set of label correspondences Σ.

Given Σ, we carefully prune the pixels in the training set with the aim of obtaining more reliable training pixels for improved training and hence better clustering. For each pixel in the original training set, we find both its k-means and EM labels, denoted by i^{kmeans} and i^{EM}. If $(i^{\text{kmeans}}, i^{\text{EM}}) \in \Sigma$, this would indicate that $(i^{\text{kmeans}}, i^{\text{EM}})$ is a pair with label consistency and that i^{kmeans} is corresponding to i^{EM}, so that we consider it as a training-reliable pixel and then add it to the new training set. Otherwise, we ignore it if $(i^{\text{kmeans}}, i^{\text{EM}}) \notin \Sigma$. Finally, with the new training data and their labels, we obtain $\mathbf{I}_5$ using the same way the mask $\mathbf{I}_3$ is computed, but the only difference is to use the new training data.

Mask Image $\mathbf{I}_6$. The sixth mask image $\mathbf{I}_6$ might be produced, depending upon if the number K of clusters needs to be updated when computing $\mathbf{I}_1$. That is, if the k-means-based labels require amendment, the K computed by Eq. 10 needs to be updated accordingly and hence we will produce $\mathbf{I}_6$. Let $K_{\text{new}}^{\text{kmeans}}$ ($K_{\text{new}}^{\text{kmeans}} < K$) denote the new number of clusters. Given $K_{\text{new}}^{\text{kmeans}}$ as the input, we just go through the same procedure as we compute $\mathbf{I}_3$, yielding $\mathbf{I}_6$.

Mask Image $\mathbf{I}_7$. Similarly, we check if there are any incorrectly labeled pixels by the EM algorithm, and if so, compute the new cluster number $K_{\text{new}}^{\text{EM}}$ ($K_{\text{new}}^{\text{EM}} < K$). Next, the seventh mask image $\mathbf{I}_7$ can be produced in the same way $\mathbf{I}_4$ is computed using $K_{\text{new}}^{\text{EM}}$ instead of K.

Mask Image $\mathbf{I}_8$. We generate the eighth mask image $\mathbf{I}_8$ following the same strategy as used for computing $\mathbf{I}_5$. Since the process of producing $\mathbf{I}_5$ involves the label correspondence/matching, the number of the unique k-means-based labels must be equal to that of the unique EM algorithm-based labels. This would mean that $\mathbf{I}_8$ will be generated if and only if $K_{\text{new}}^{\text{kmeans}} = K_{\text{new}}^{\text{EM}} < K$.

Final Mask Image I. The final binary image $\mathbf{I}$ is constructed after combining all the existing masks $\{\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_8\}$. Specifically, $\mathbf{I}$ is computed by comparing the numbers of the pixel values 0 and 1 as follows:

$$\mathbf{I}(x,y) = \begin{cases} 0 & \text{if} \sum_i (\mathbf{I}_i(x,y) == 1) < \sum_i (\mathbf{I}_i(x,y) == 0) \\ 1 & \text{otherwise} \end{cases}. \quad (15)$$

Notice again that $\{\mathbf{I}_6, \mathbf{I}_7, \mathbf{I}_8\}$ may not exist and also that a background pixel at the position (x, y) is not considered as an FCL pixel so that $\mathbf{I}_i(x, y) = 1$ and hence $\mathbf{I}(x, y) = 1$, $\forall i \in \{1, 2, \ldots, 8\}$.

It is worth mentioning here that we have observed that the amount each mask image $\mathbf{I}_i$ contributes varies over pages, indicating that that it is not reasonable to claim $\mathbf{I}_i$ is consistently better than $\mathbf{I}_j$ in terms of pixel classification. Moreover, presented above is just an efficient strategy for generating $\mathbf{I}$, but we do not focus on investigating and comparing other possible strategies that may use $\{\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_8\}$ in a different manner.

Given the final binary mask $\mathbf{I}$, we are able to extract FCL from it. In other words, FCL are considered as those connected components of $\mathbf{I}$ that satisfy the constraints described in [Yang et al. 2015].

6. EXPERIMENTAL RESULTS

In this section, we perform a comprehensive evaluation of the proposed algorithms on a public dataset with 1819 images of pages of 7 different manuscripts from the Yale Universitys Beinecke Rare Book and Manuscript Digital Library [Beinecke 2014a], the Oxford Universitys Bodleian Library and the Walters Art Museum. The dataset is available to download at [Beinecke 2014b]. The data are very heterogeneous, in terms of layout structure (e.g., number of columns and text density), conservation (e.g., ageing, ink bleed-through and noise), acquisition resolution and writing styles. We implemented these algorithms using MATLAB, C++ and the OpenCV library [OpenCV 2013]. The computational time depends on the dimensions of the test images. For a 3128×2274-sized image, our non-optimized implementation takes approximately 11 seconds, 12 seconds and 3.5 minutes respectively to extract text blocks, text lines and FCL on a PC running on an 8 Intel Core i7-3630QM CPU 2.40GHz processor with 12GB memory.

6.1 Evaluation Methodology

We evaluate the performance of our algorithms in both *pixel-level* and *object-level*. Like [Pintus et al. 2014], we report the results in *Precision* and *Recall* values, given by:

$$\begin{cases} \text{Precision} & = \dfrac{\text{TP}}{\text{TP} + \text{FP}} \\ \text{Recall} & = \dfrac{\text{TP}}{\text{TP} + \text{FN}} \end{cases}, \quad (16)$$

where TP, FP and FN indicate true-positive, false positive and false-negative, respectively. Note that we take into account all pixels when computing TP, FP and FN for *pixel-level* evaluation.

By pixel-level, we mean that we consider *pixel classification accuracy* when computing TP, FP and FN. In doing so, we compare the results automatically generated by the proposed algorithms against a ground truth dataset. To create the ground truth data, we, for each algorithm proposed, randomly select 5 images from each manuscript, forming a dataset of 105 images. Next, we obtain a single binary mask by manually segmenting each image in the dataset into either text blocks, text lines or FCL. Fig. 6 shows our manually produced ground truth masks for an example image. It is worth mentioning here that the pixel-level based evaluation strategy is only practical for a small set of images (105 in

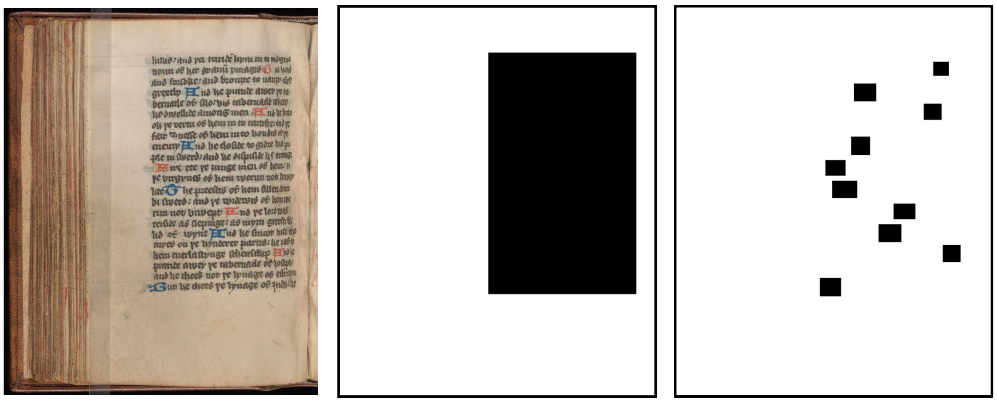

Fig. 6. Illustration of the manually created mask images for evaluating text block computation algorithm (middle) and FCL extraction algorithm (right) for an example manuscript page image (left). Manuscript images courtesy of the Yale University [BeineckeMS360].

our experiments), since the ground truth data for the test data used in the experiments is not available yet and also since creating them is a very tedious and time-consuming process.

Given a ground truth mask $\mathbf{P}^{\text{man}}$ and its corresponding mask $\mathbf{P}^{\text{auto}}$ resulting from one of our algorithms, we analyze the pixel value pairs $(\mathbf{P}^{\text{man}}(x, y), \mathbf{P}^{\text{auto}}(x, y))$ at each position (x, y). That is, TP, FP and FN correspond to (0, 0), (1, 0) and (0, 1), respectively.

By object-level, we mean that TP, FP and FN are computed based on the *object classification accuracy*, which is obtained by visually assessing the automatically-generated results. As these results are visualized by drawing rectangles or lines, we just need to go through all the test images to perceptually verify the result correctness and obtain the resulting TP, FP and FN. For instance, to evaluate the FCL results, we presented them to two users, who were asked to give the TP, FP and FN. In order for an extracted FCL to be deemed as TP, there must be more than 80% overlap between it and its corresponding rectangular box. Notice that the object-level evaluation emphasizes the capability of successfully detecting the objects (text block, text lines and FCL) rather than the pixel-level detection accuracy. Although this strategy is a binary judgment (correct or incorrect) [Pintus et al. 2015], it makes extensive validation possible while efficiently evaluating the algorithm performance.

It is worth mentioning that the groundtruth data for the whole dataset is not available yet and also that creating the groundtruth data, though very useful, for such a big dataset is not considered a contribution of our work. Thus, we can only create the groundtruth data for a small portion of the dataset, which were randomly selected to avoid biased selection.

Another important reason of performing evaluation on groundtruth data is to demonstrate that the results by object-level evaluation (used for extensive evaluation when groundtruth data is not available) are reliable. Indeed, by comparing Tables II and III, Tables IV and V, and Tables VI and VIII, we can see that the results by pixel-level evaluation match well with those by object-level evaluation, indicating we would expect similar results when evaluating on the groundtruth data for the whole dataset. Therefore, even though the groundtruth data we tested on is for 105 images, the results obtained on them can still reliably reflect the overall performance of the proposed methods.

6.2 Text Block Computation Results

We applied the text block computation algorithm to the randomly selected 35 images with 42 text blocks. As we can see from Table II showing the resulting pixel-level evaluation results, the proposed

Table II. Pixel-level evaluation results for text block computation.

Manuscript Name	Blocks	TP	FP	FN	Precision	Recall
BeineckeMS10	5	9.54×10^6	7.54×10^4	5.62×10^5	99.22%	94.44%
BeineckeMS109	5	4.05×10^6	4.13×10^4	1.57×10^5	98.99%	96.27%
BeineckeMS310	5	2.97×10^7	6.10×10^5	4.10×10^5	97.99%	98.64%
BeineckeMS360	5	5.28×10^6	8.24×10^4	1.56×10^5	98.46%	97.13%
Osborna44	5	8.22×10^6	2.09×10^5	4.45×10^5	97.52%	94.86%
BodleianMSBodley850	12	3.34×10^6	1.97×10^5	1.57×10^5	94.42%	95.50%
Walters34	5	1.82×10^7	5.24×10^5	1.19×10^6	97.20%	93.85%
# Manuscripts – 7	42	7.83×10^7	1.73×10^6	3.04×10^6	97.84%	96.26%

Table III. Object-level evaluation results for text block computation. We compare these results against those obtained by the state-of-the-art method [Pintus et al. 2014].

Manuscript Name	Blocks	TP	FP	FN	Precision	Recall
					[Pintus et al. 2014]/Our	[Pintus et al. 2014]/Our
BeineckeMS10	173	173	0	0	NA/100%	NA/100%
BeineckeMS109	250	246	1	4	91.01%/99.60%	98.83%/98.40%
BeineckeMS310	277	275	4	2	94.56%/98.57%	90.06%/99.28%
BeineckeMS360	370	370	0	0	67.26%/100%	97.17%/100%
Osborna44	470	467	2	3	77.73%/99.57%	95.25%/99.36%
BodleianMSBodley850	457	447	22	10	NA/95.31%	NA/97.81%
Walters34	48	40	0	8	NA/100%	NA/83.33%
# Manuscripts – 7	2045	2018	29	27	82.64%/99.01%	95.33%/96.88%

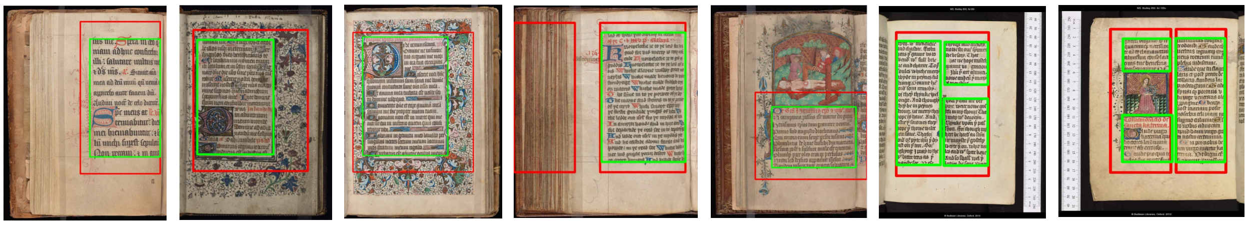

Fig. 7. Block computation results for several example page images, which are respectively from *BeineckeMS10*, *BeineckeMS109*, *BeineckeMS310*, *BeineckeMS360*, *Osborna44* and *BodleianMSBodley850* (from left to right). The two rightmost images show pages from the *BodleianMSBodley850* manuscript. The green and red rectangles show the results before and after text block refinement. Manuscript images courtesy of the Yale University [BeineckeMS10 ; BeineckeMS109 ; BeineckeMS310 ; BeineckeMS360 ; Osborna44] and the Oxford University [BodleianMSBodley850].

algorithm can achieve very satisfactory results, with an overall precision value of up to 97% and a recall value of up to 96%.

By contrast, we also evaluated the algorithm on all the test images with 2045 text blocks in order to further demonstrate its efficiency. After visually checking the results, we list the numerical statistics in Table III. It is easy to see from this table that we can achieve up to 99% precision and 96% recall.

To visualize the detected text blocks, we draw the rectangles over their own images. Fig. 7 illustrates the text block computation results for few example images from 7 physical appearance-distinct manuscripts. This figure demonstrates that we are able to identify the unwanted image parts from the coarse text blocks and remove them to produce the satisfactory refined blocks. Moreover, the rightmost image shows that the proposed method is still successful even when used to detect the text blocks with a few text lines. From these results produced on the massive number of page images, we can conclude that the proposed method is capable of automatically extracting text blocks.

Table IV. Pixel-level evaluation results for text line segmentation.

Manuscript Name	Lines	TP	FP	FN	Precision	Recall
BeineckeMS10	60	9.23×10^6	4.55×10^5	1.33×10^6	95.30%	87.41%
BeineckeMS109	100	4.06×10^6	5.54×10^5	5.43×10^5	87.99%	88.20%
BeineckeMS310	88	1.28×10^7	6.50×10^5	2.29×10^6	95.17%	84.82%
BeineckeMS360	110	4.89×10^6	4.73×10^5	5.34×10^5	91.18%	90.15%
Osborna44	90	7.76×10^6	6.86×10^5	8.21×10^5	91.88%	90.43%
BodleianMSBodley850	220	3.43×10^6	2.57×10^5	3.86×10^5	93.03%	89.89%
Walters34	91	4.35×10^8	2.29×10^6	4.72×10^6	99.48%	98.93%
# Manuscripts - 7	759	4.77×10^8	5.37×10^6	1.06×10^7	98.89%	97.83%

6.3 Text Line Segmentation Results

Table IV reports the text line segmentation results obtained from applying the proposed method to the randomly selected 35 images with 759 lines. From this table, we are able to achieve the precision and recall values as high as 98% and 97% across the manuscripts tested. However, due to the skew introduced during document scanning (see Fig. 8), the text lines have inclinations with respect to the horizontal lines. This is in conflict with our assumption that they are parallel to the horizontal lines, hence making the precision and recall by the text line segmentation algorithm smaller than those by the text block computation approach. For improved results, we can apply the de-skewing algorithms to pre-process the images before using our method.

Table V. Object-level evaluation results for text line segmentation. We compare these results against those obtained by the state-of-the-art methods [Pintus et al. 2015; Pintus et al. 2014].

Manuscript Name	Lines	TP	FP	FN	Precision	Recall
					[Pintus et al. 2015]/[Pintus et al. 2014]/Our	[Pintus et al. 2015]/[Pintus et al. 2014]/Our
BeineckeMS10	2076	2076	2	0	100%/NA/99.90%	99.90%/NA/100%
BeineckeMS109	4652	4586	21	66	98.64%/97.48%/99.54%	97.73%/99.00%/98.58%
BeineckeMS310	5264	5161	83	103	98.64%/95.24%/98.42%	97.73%/99.00%/98.04%
BeineckeMS360	8162	8044	0	118	98.61%/92.09%/100%	98.23%/98.19%/98.55%
Osborna44	9317	9134	29	183	99.96%/97.07%/99.68%	90.37%/99.03%/98.04%
BodleianMSBodley850	9695	9409	12	286	NA/NA/99.87%	NA/NA/97.05%
Walters34	681	607	6	74	99.92%/NA/99.02%	95.18%/NA/89.13%
# Manuscripts – 7	39847	39017	153	830	99.17%/95.47%/99.61%	96.79%/98.81%/97.92%

In addition, we apply the algorithm to all the test images, which have a total of 39847 text lines. Table V summaries the object-level evaluation results obtained from visual inspection. Again, the algorithm yields up to 100% precision and recall for some manuscripts, exhibiting very satisfactory performance.

Fig. 8 visualizes the line segmentation results for a few images. As this figure shows, the proposed method is successful in segmenting the text lines, even if the pages are seriously degraded with low quality.

6.4 FCL Extraction Results

To evaluate the performance of the FCL extraction algorithm, we first apply it to the random 35 images with 357 FCL. Table VI shows that in this environment the average pixel-level precision and recall accuracies can reach as high as 98.71% and 96.93%, respectively. Nevertheless, the pixel-level results do not explicitly indicate the accuracy of successful FCL extraction because we cannot infer from them how many FCL have been successfully detected.

In Table VII, we compare the pixel-level based results obtained from using the individual masks $\{\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_8\}$ and their combination $\mathbf{I}$ for FCL extraction. The results indicate that $\mathbf{I}_1$, $\mathbf{I}_2$, $\mathbf{I}_5$, $\mathbf{I}_6$ and $\mathbf{I}_8$

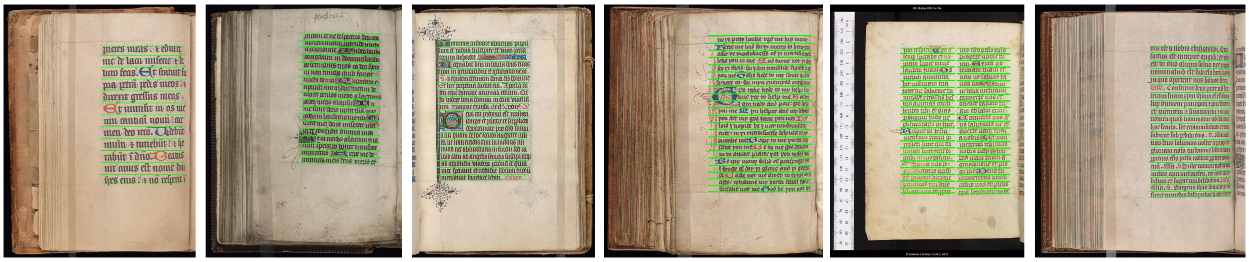

Fig. 8. Text line segmentation results for several example page images, which are respectively from *BeineckeMS10*, *BeineckeMS109*, *BeineckeMS310*, *BeineckeMS360*, *BodleianMSBodley850* and *Osborna44* (from left to right). Manuscript images courtesy of the Yale University [BeineckeMS10 ; BeineckeMS109 ; BeineckeMS310 ; BeineckeMS360 ; Osborna44] and the Oxford University [BodleianMSBodley850].

Table VI. Pixel-level evaluation results for FCL extraction.

Manuscript Name	FCL	TP	FP	FN	Precision	Recall
BeineckeMS10	17	1.02×10^6	3.38×10^4	5.66×10^4	96.79%	94.83%
BeineckeMS109	37	6.51×10^5	2.53×10^5	2.96×10^4	72.01%	95.65%
BeineckeMS310	77	2.89×10^6	2.66×10^5	2.08×10^5	91.57%	93.29%
BeineckeMS360	50	5.12×10^5	2.33×10^4	1.73×10^4	95.65%	96.73%
Osborna44	41	1.10×10^6	1.34×10^5	6.31×10^4	89.11%	94.58%
BodleianMSBodley850	90	5.05×10^5	6.27×10^4	6.33×10^4	88.94%	88.85%
Walters34	45	3.03×10^7	3.95×10^5	4.96×10^5	98.71%	98.39%
# Manuscripts – 7	357	3.70×10^7	1.17×10^6	9.34×10^5	98.71%	96.93%

Table VII. Comparison of pixel-level evaluation results obtained from using the individual masks $\{\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_8\}$ and their combination $\mathbf{I}_8$ for FCL extraction.

Mask Name	FCL	TP	FP	FN	Precision	Recall
$\mathbf{I}_1$	312	6.29×10^6	1.04×10^6	0.83×10^6	85.78%	88.32%
$\mathbf{I}_2$	312	6.26×10^6	1.18×10^6	0.87×10^6	84.17%	87.83%
$\mathbf{I}_3$	312	6.69×10^6	3.19×10^6	0.43×10^6	67.70%	93.97%
$\mathbf{I}_4$	312	6.64×10^6	3.37×10^6	0.48×10^6	66.29%	93.26%
$\mathbf{I}_5$	312	6.36×10^6	1.25×10^6	0.76×10^6	83.62%	89.32%
$\mathbf{I}_6$	312	1.70×10^6	0.31×10^6	0.37×10^6	84.53%	82.06%
$\mathbf{I}_7$	312	1.40×10^6	0.54×10^6	0.12×10^6	72.35%	91.97%
$\mathbf{I}_8$	312	1.28×10^6	0.36×10^6	0.25×10^6	77.98%	83.90%
$\mathbf{I}$	312	6.68×10^6	7.73×10^5	4.38×10^5	89.63%	93.85%

provide similar *Precision* and *Recall* values. Although $\mathbf{I}_3$, $\mathbf{I}_4$ and $\mathbf{I}_7$ each have relatively lower *Precision*, they can offer higher *Recall*. On the other hand, just as expected, using the mask $\mathbf{I}$, which is the combination of $\{\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_8\}$, yields the best *Precision* and *Recall* values than using any individual mask $\mathbf{I}_j$, $1 \leq j \leq 8$ does.

By contrast, Table VIII reports the precision and recall rates of successful FCL extraction from all the 1819 images with 13668 FCL. Although the pages of these test manuscript, especially the manuscript *BeineckeMS109*, contain a great deal of noise and are of very low quality, our approach still achieves very desirable performance.

We visualize the detected FCL for several images in Fig. 9. Notice that it is quite challenging to extract all the FCL from the second leftmost and rightmost images since some of them look rather

Table VIII. Object-level evaluation results for FCL extraction. We compare these results against those obtained by the method presented in [Yang et al. 2015].

Manuscript Name	FCL	TP	FP	FN	Precision	Recall
					[Yang et al. 2015]/Our	[Yang et al. 2015]/Our
BeineckeMS10	767	724	20	43	87.82%/97.31%	99.23%/94.39%
BeineckeMS109	1386	1333	115	53	88.55%/92.06%	95.80%/96.18%
BeineckeMS310	2443	2330	112	113	90.02%/95.41%	96.33%/95.37%
BeineckeMS360	3503	3257	13	246	98.36%/99.60%	89.93%/92.98%
Osborna44	3386	3144	118	242	NA/96.38%	NA/92.85%
BodleianMSBodley850	1793	1475	62	318	NA/95.97%	NA/82.26%
Walters34	390	323	13	67	NA/96.13%	NA/82.82%
# Manuscripts – 7	13668	12586	453	1082	91.19%/96.53%	95.32/92.08%

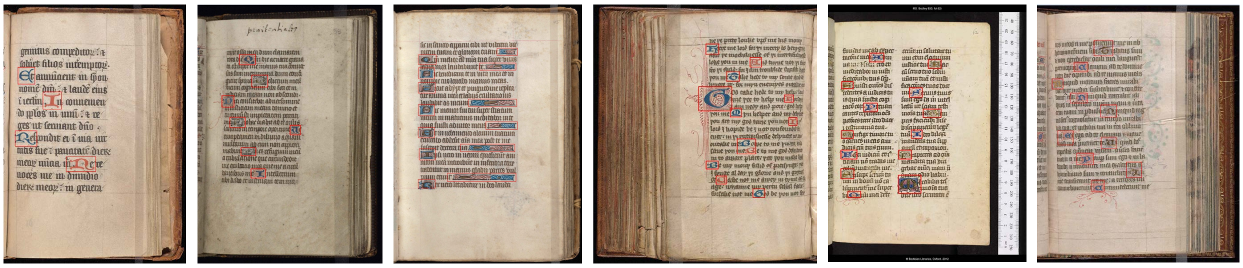

Fig. 9. FCL extraction results for several example page images, which are respectively from *BeineckeMS10*, *BeineckeMS109*, *BeineckeMS310*, *BeineckeMS360*, *BodleianMSBodley850* and *Osborna44* (from left to right). Manuscript images courtesy of the Yale University [BeineckeMS10 ; BeineckeMS109 ; BeineckeMS310 ; BeineckeMS360 ; Osborna44] and the Oxford University [BodleianMSBodley850].

similar to the text. Fortunately, the visualization shows that our proposed algorithm is able to extract all of them, clearly demonstrating its efficiency.

6.5 Algorithm Comparison

We compare our proposed text block computation and text line segmentation algorithms against two state-of-the-art methods in [Pintus et al. 2015; Pintus et al. 2014] and the FCL extraction method against [Yang et al. 2015]. As the comparison results listed in Tables III and V show, our approaches very likely outperform [Pintus et al. 2015; Pintus et al. 2014] in terms of both precision and recall rates. Moreover, Table VIII indicates that, compared to [Yang et al. 2015], the proposed method provides higher precision by anywhere between 1% to 10%, as well as comparable recall values.

Regarding applicability, we believe that our per-page based methods can be used in a wider range of applications because the approaches presented in [Pintus et al. 2015; Pintus et al. 2014] both work on a per-book basis. The method by Pintus et al. [Pintus et al. 2014] requires the availability of few pages from the same manuscript in order to train a classifier. By contrast, our methods are all single-page based with no reference to any other pages.

Fig. 10 illustrates the automatically generated results for some challenging pages. Despite the existence of noise and/or marginal decorations that can create difficulties, the proposed algorithms still succeed in correctly extracting the text block, text lines and FCL.

However, our algorithms could yield unexpected results when applied to pages that are composed of sparse text lines. This is mainly due to the heavy dependence on the text leading H and also the assumption about the spacing between two text lines. Fig. 11 illustrates an example image for which the algorithms are not very successful.

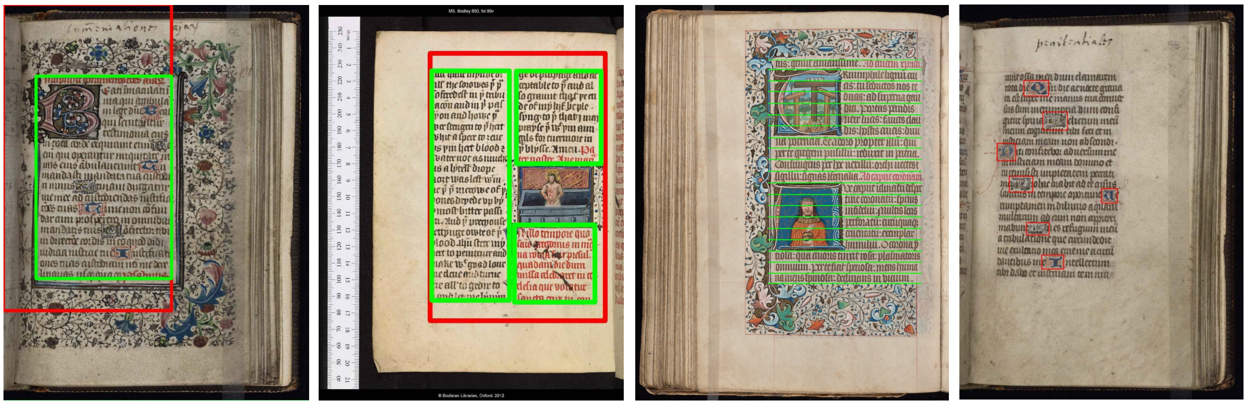

Fig. 10. Results for some challenging pages, which contain a great deal of noise and/or marginal decorations. Manuscript images courtesy of the Yale University [BeineckeMS109 ; BeineckeMS310] and the Oxford University [BodleianMSBodley850].

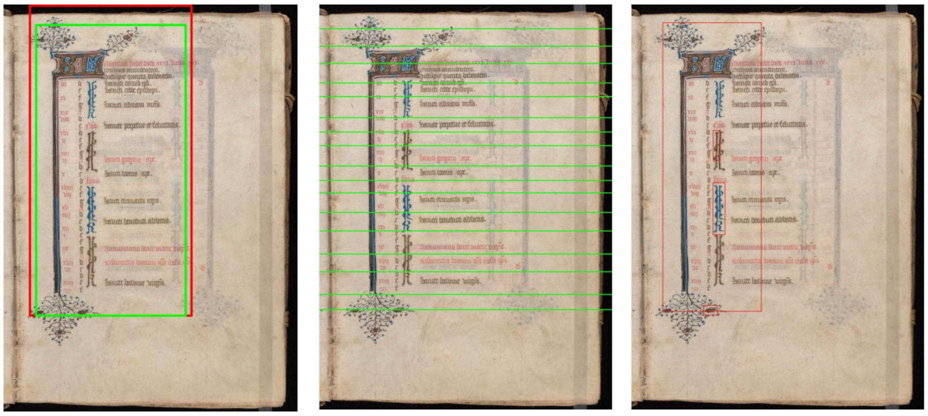

Fig. 11. Failure example page. The text height computed by ATHENA [Pintus et al. 2013] is 158, but its real value is 60. Manuscript images courtesy of the Yale University [BeineckeMS310].

Although comparing the proposed methods against other state-of-the-art algorithms allows us to further evaluate the performance of our methods, to the best of our knowledge, there is one major difficulty: the vast majority of the previous algorithms are not completely automatic and independent of the text leading and image resolution. They typically require a parameter set by the user. This manual tuning, which is not present in our approaches, might add a bias in the comparison and evaluation.

7. CONCLUSION

Given an image of a medieval manuscript page, we present three fully automatic, per-page-based layout analysis algorithms: *text block computation*, *text line segmentation* and *special component extraction*. Their automation property enables them to be particularly useful for scenarios where many images need to be analyzed. Moreover, unlike the state-of-the-art methods requiring images of few pages from the same manuscript to work, our proposed algorithms work on a per-page basis, so that they can be used in a wider range of applications.

We carried out an extensive experimental evaluation of the proposed algorithms by testing them on 1819 images of pages of 7 distinct medieval manuscripts. As demonstrated by the results, they can achieve very satisfactory performance; that is, the overall precision and recall values are as high as 99% and 97%, respectively, for the text line segmentation algorithm. Even for the very changeling task of extracting FCLs, our method is able to achieve up to a 96% precision rate. The success is attributed to taking advantage of previous clustering (both unsupervised and supervised) and template matching techniques, as well as the a priori knowledge (the text leading is H) about the page physical structure.

As a direction for future research, we plan to investigate the analysis methods based on superpixels other than single individual pixel. The aim is to make further improvements by regarding as an independent component each superpixel rather than a single pixel. We will further evaluate the performance of the proposed algorithms by comparing them against other state-of-the-art approaches and also testing them on other publicly available datasets. In order to make the evaluation easier to perform, we have developed a program for creating the ground truth data. This will allow us to conduct in-depth analysis on the results and hence to design improved algorithmic strategy. Therefore, our future work also includes creating ground truth data so that algorithm evaluation can be simplified and performed without requiring users to manually judge the correctness of results.

REFERENCES

ANTONACOPOULOS, A., PLETSCHACHER, S., BRIDSON, D., AND PAPADOPOULOS, C. 2009. Icdar 2009 page segmentation competition. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*. 1370–1374.

ARIVAZHAGAN, M., SRINIVASAN, H., AND SRIHARI, S. 2007. A statistical approach to line segmentation in handwritten documents. In *Electronic Imaging 2007*. International Society for Optics and Photonics, 65000T–65000T.

ARTHUR, D. AND VASSILVITSKII, S. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 1027–1035.

ARVANITOPOULOS, N. AND SUSSTRUNK, S. 2014. Seam carving for text line extraction on color and grayscale historical manuscripts. In *14th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 726–731.

ASI, A., COHEN, R., KEDEM, K., EL-SANA, J., AND DINSTEIN, I. 2014. A coarse-to-fine approach for layout analysis of ancient manuscripts. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*. IEEE, 140–145.

BAECHLER, M., BLOECHLE, J.-L., AND INGOLD, R. 2010. Semi-automatic annotation tool for medieval manuscripts. In *2010 International Conference on Frontiers in Handwriting Recognition*. IEEE, 182–187.

BAR-YOSEF, I., HAGBI, N., KEDEM, K., AND DINSTEIN, I. 2009. Line segmentation for degraded handwritten historical documents. In *10th International Conference on Document Analysis and Recognition*. IEEE, 1161–1165.

BEINECKE. 2014a. Beinecke rare book and manuscript library.

BEINECKE. 2014b. Database - download scripts - http://hdl.handle.net/10079/cz8w9v8.

BEINECKEMS10. Beinecke rare book and manuscript library - Yale University.

BEINECKEMS109. Beinecke rare book and manuscript library - Yale University.

BEINECKEMS310. Beinecke rare book and manuscript library - Yale University.

BEINECKEMS360. Beinecke rare book and manuscript library - Yale University.

BODLEIANMSBODLEY850. Bodleian library - Oxford University.

BRANTLEY, J. 2009. The prehistory of the book. *PMLA 124,* 2, 632–639.

CHEN, K., WEI, H., HENNEBERT, J., INGOLD, R., AND LIWICKI, M. 2014. Page segmentation for historical handwritten document images using color and texture features. In *International Conference on Frontiers in Handwriting Recognition*.

DAVIES, D. L. AND BOULDIN, D. W. 1979. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2, 224–227.

DE HAMEL, C. 1992. *Scribes and illuminators*. University of Toronto Press.

FLETCHER, L. A. AND KASTURI, R. 1988. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 10,* 6, 910–918.

GARZ, A., FISCHER, A., BUNKE, H., AND INGOLD, R. 2013. A binarization-free clustering approach to segment curved text lines in historical manuscripts. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 1290–1294.

GRANA, C., BORGHESANI, D., AND CUCCHIARA, R. 2009. Picture extraction from digitized historical manuscripts. In *Proceedings of the ACM International Conference on Image and Video Retrieval*. ACM, 22.

JAIN, A. K. AND YU, B. 1998. Document representation and its application to page decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20,* 3, 294–308.

KOO, H. I. AND CHO, N. I. 2012. Text-line extraction in handwritten chinese documents based on an energy minimization framework. *Image Processing, IEEE Transactions on 21,* 3, 1169–1175.

KUHN, H. W. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly 2,* 1-2, 83–97.

LIKFORMAN-SULEM, L., HANIMYAN, A., AND FAURE, C. 1995. A hough based algorithm for extracting text lines in handwritten documents. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*. Vol. 2. IEEE, 774–777.

LIKFORMAN-SULEM, L., ZAHOUR, A., AND TACONET, B. 2007. Text line segmentation of historical documents: a survey. *International Journal of Document Analysis and Recognition (IJDAR) 9,* 2-4, 123–138.

LOULOUDIS, G., GATOS, B., PRATIKAKIS, I., AND HALATSIS, C. 2008. Text line detection in handwritten documents. *Pattern Recognition 41,* 12, 3758–3772.

LOULOUDIS, G., GATOS, B., PRATIKAKIS, I., AND HALATSIS, C. 2009. Text line and word segmentation of handwritten documents. *Pattern Recognition 42,* 12, 3169–3183.

MANMATHA, R. AND ROTHFEDER, J. L. 2005. A scale space approach for automatically segmenting words from historical handwritten documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27,* 8, 1212–1225.

MEHRI, M., MHIRI, M., HÉROUX, P., GOMEZ-KRÄMER, P., MAHJOUB, M. A., AND MULLOT, R. 2014. Performance evaluation and benchmarking of six texture-based feature sets for segmenting historical documents. In *International Conference on Pattern Recognition*. 2885–2890.

NAGY, G. 2000. Twenty years of document image analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22,* 1, 38–62.

NIKOLAOU, N., MAKRIDIS, M., GATOS, B., STAMATOPOULOS, N., AND PAPAMARKOS, N. 2010. Segmentation of historical machine-printed documents using adaptive run length smoothing and skeleton segmentation paths. *Image and Vision Computing 28,* 4, 590–604.

OPENCV. 2013. OpenCV - open source computer vision library. http://opencv.org/.

OSBORNA44. Beinecke rare book and manuscript library - Yale University.

PARK, H.-C., OK, S.-Y., YU, Y.-J., AND CHO, H.-G. 2001. A word extraction algorithm for machine-printed documents using a 3d neighborhood graph model. *International Journal on Document Analysis and Recognition 4,* 2, 115–130.

PINTUS, R., YANG, Y., GOBBETTI, E., AND RUSHMEIER, H. 2014. A TaLISMAN: Automatic text and LIne segmentation of historical MANuscripts. In *12th EUROGRAPHICS Workshop on Graphics and Cultural Heritage (accepted)*.

PINTUS, R., YANG, Y., AND RUSHMEIER, H. 2013. ATHENA: Automatic text height extraction for the analysis of old handwritten manuscripts. In *Digital Heritage International Congress*. Vol. 1. IEEE, 605–612.

PINTUS, R., YANG, Y., AND RUSHMEIER, H. 2015. ATHENA: Automatic text height extraction for the analysis of text lines in old handwritten manuscripts. *ACM Journal on Computing and Cultural Heritage 8,* 1, 1:1–1:25.

RATH, T. M. AND MANMATHA, R. 2003. Word image matching using dynamic time warping. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. Vol. 2. IEEE, II–521 – II–527.

RODRÍGUEZ-SERRANO, J. A. AND PERRONNIN, F. 2009. Handwritten word-spotting using hidden markov models and universal vocabularies. *Pattern Recognition 42,* 9, 2106–2116.

RYU, J., KOO, H. I., AND CHO, N. I. 2014. Language-independent text-line extraction algorithm for handwritten documents. *Signal Processing Letters, IEEE 21,* 9, 1115–1119.

SAABNI, R., ASI, A., AND EL-SANA, J. 2014. Text line extraction for historical document images. *Pattern Recognition Letters 35*, 23–33.

SHAFAIT, F., KEYSERS, D., AND BREUEL, T. M. 2008. Performance evaluation and benchmarking of six-page segmentation algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 30,* 6, 941–954.

SHI, Z. AND GOVINDARAJU, V. 2004. Line separation for complex document images using fuzzy runlength. In *International Workshop on Document Image Analysis for Libraries*. IEEE Computer Society, 306–306.

SHI, Z., SETLUR, S., AND GOVINDARAJU, V. 2009. A steerable directional local profile technique for extraction of handwritten arabic text lines. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*. IEEE, 176–180.

STAMATOPOULOS, N., GATOS, B., LOULOUDIS, G., PAL, U., AND ALAEI, A. 2013. ICDAR 2013 handwriting segmentation contest. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE, 1402–1406.

STOŠIĆ, M., MARQUES, M., AND COSTEIRA, J. P. 2011. Convex solution of a permutation problem. *Linear Algebra and its Applications 434,* 1, 361–369.

VON GIOI, R. G., JAKUBOWICZ, J., MOREL, J.-M., AND RANDALL, G. 2010. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32,* 4, 722–732.

YANG, Y., PINTUS, R., GOBBETTI, E., AND RUSHMEIER, H. 2015. Automated color clustering for medieval manuscript analysis. In *Digital Heritage*. IEEE.