A Scale Space Approach for Automatically Segmenting Words from Historical Handwritten Documents

R. Manmatha, *Member*, *IEEE Computer Society*, and Jamie L. Rothfeder

Abstract—Many libraries, museums, and other organizations contain large collections of handwritten historical documents, for example, the papers of early presidents like George Washington at the Library of Congress. The first step in providing recognition/retrieval tools is to automatically segment handwritten pages into words. State of the art segmentation techniques like the gap metrics algorithm have been mostly developed and tested on highly constrained documents like bank checks and postal addresses. There has been little work on full handwritten pages and this work has usually involved testing on clean artificial documents created for the purpose of research. Historical manuscript images, on the other hand, contain a great deal of noise and are much more challenging. Here, a novel scale space algorithm for automatically segmenting handwritten (historical) documents into words is described. First, the page is cleaned to remove margins. This is followed by a gray-level projection profile algorithm for finding lines in images. Each line image is then filtered with an anisotropic Laplacian at several scales. This procedure produces blobs which correspond to portions of characters at small scales and to words at larger scales. Crucial to the algorithm is scale selection, that is, finding the optimum scale at which blobs correspond to words. This is done by finding the maximum over scale of the extent or area of the blobs. This scale maximum is estimated using three different approaches. The blobs recovered at the optimum scale are then bounded with a rectangular box to recover the words. A postprocessing filtering step is performed to eliminate boxes of unusual size which are unlikely to correspond to words. The approach is tested on a number of different data sets and it is shown that, on 100 sampled documents from the George Washington corpus of handwritten document images, a total error rate of 17 percent is observed. The technique outperforms a state-of-the-art gap metrics word-segmentation algorithm on this collection.

Index Terms—Segmentation, document and text processing, document analysis, handwriting analysis, document indexing, smoothing, optical character recognition.

1 Introduction

Many libraries, museums, and other organizations contain large collections of handwritten historical documents. These include the papers of the early presidents like George Washington at the Library of Congress, Isaac Newton's papers at Cambridge University, and the collections of field biologists like Joseph Grinnell in the Museum of Vertebrate Zoology at the University of California Berkeley. These collections are valuable for scholars, researchers, and even the common man. George Washington's letters, for example, provide insight into the founding of the United States of America. Joseph Grinnell's field notes contain useful scientific information on the fauna of Yosemite a hundred years ago. Some of these collections are very large and it is essential that tools for navigating or searching them be available. One approach to this problem is to use manual transcription—however, because it is tedious and expensive, a complete manual transcription of all historical documents is infeasible. Having computers automatically generate indices or provide a search tool would be a cost-effective alternative and would allow access to these important documents.

Recently, there has been some work on this problem. Rath et al. [25] demonstrated a system to retrieve 1,000 pages of George Washington's handwritten document images using text queries (it uses the segmentation described here). It involves using a statistical *relevance model* which learns the joint probability of image features being paired with annotations (text labels). The relevance model is learned using a training set of annotated word images and may then be used to probabilistically annotate the word images with (ASCII) words. Retrieval can then performed using these probabilistic annotations. Manmatha et al. [15], [16] proposed a technique called "word spotting" for historical documents which involves clustering different instances of the same word. The word images have links back to the original page images, providing an index similar to the index at the back of a book. Tomai et al. [28] discuss how to map transcripts of a historical document onto a page image given that a transcript is available. Govindaraju and Xue [7] and Lavrenko et al. [12] investigate the problem of handwriting recognition in historical documents.

All of these papers assume that the words have been correctly segmented from their documents. There has been a fair amount of work on segmentation for specialized systems, like those for postal address or bank check

• *The authors are with the Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, Amherst, 140 Governors Dr., Amherst, MA 01003.*
E-mail: {manmatha, jrothfed}@cs.umass.edu.

Manuscript received 26 Mar. 2004; revised 29 Oct. 2004; accepted 15 Dec. 2004; published online 13 June 2005.
Recommended for acceptance by V. Govindaraju.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0146-0304.

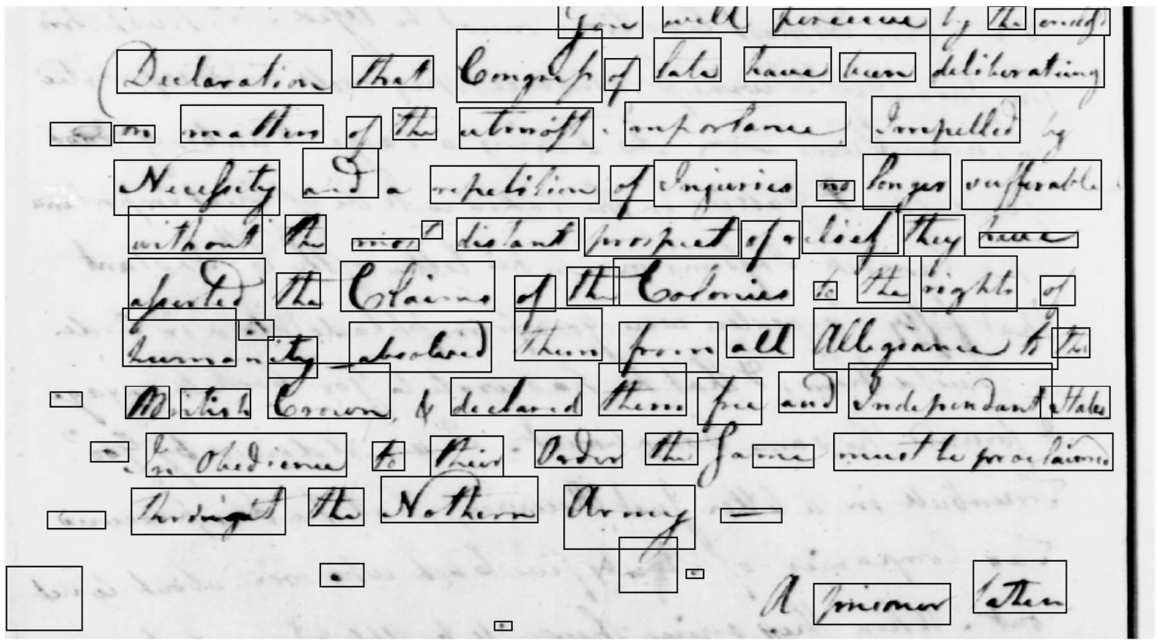


Fig. 1. Scale space segmentation result on a noisy image from the George Washington collection. Note the good quality segmentation despite the presence of faded ink and extensive bleed-through.

recognition. Full page segmentation schemes for handwriting are few and have mainly been tested on clean modern documents which are written specifically for testing document analysis systems [19]. Even for such pages, the word segmentation problem is often difficult. For example, Marti and Bunke [21] discuss the problem of segmenting lines in modern handwriting. They mention that their recognition system avoids the "difficult problem of segmenting a line of text into individual words." Historical documents are even more difficult to segment. They contain a lot of noise due to degradation, dirt, margins, and other artifacts. The process of scanning documents from microfilm or microfiche often adds black margins and bleed-through. Furthermore, these documents can be seriously degraded due to aging. In this paper, which is a significant expansion of the preliminary work introduced in [17], we describe a novel scale space algorithm for segmenting handwritten (historical) manuscripts.[1] Fig. 1 shows an example of the output of the scalespace segmentation on part of George Washington's page. Notice the ink fading and extensive bleed-through present on this page.

The input to the system is a gray-level document image (see Fig. 2 for a visual description of this process). The image is processed to remove margins and horizontal lines that are likely to interfere with later operations (see Section 6). The page is then dissected into lines using projection analysis techniques modified for gray scale images. The projection function is smoothed with a Gaussian filter (low pass filtering) to remove maxima, which cause false alarms, and the positions of the local maxima (i.e., space between the lines) are detected. The line images are smoothed and then convolved with second order anisotropic Gaussian derivative filters at multiple scales to create a scale space and the *blob*-like features which arise from this representation give us the focus of attention regions (i.e., words in the original document image). At small scales, the blobs correspond to character like features. At larger scales (Fig. 5), the blobs correspond to words. Increasing the scales further leads to the blobs merging into even larger units. One of the key issues is automatically selecting the scale for these blob-like features. An efficient technique for scale selection is described whereby the correct scale for blob extraction is obtained by finding the scale maxima of the blob extent. A connected components analysis of the blob image followed by a reverse mapping of the bounding boxes allows us to extract the words. The box is then extended to include the ascenders and descenders. Our approach to word segmentation is novel as it is the first

1. It is interesting to note that scale space techniques were first proposed in the context of optical character recognition by Iijima (see [31]).

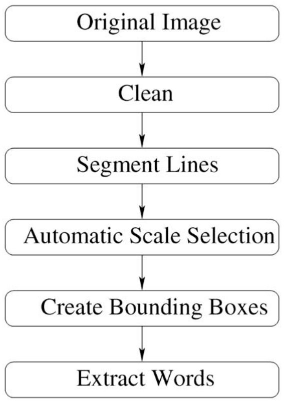


Fig. 2. Illustration of the entire process. Lines are segmented in the gray-level document image. A scale space image is created for each line and scale selection determines the correct scale at which blobs correspond to words.

algorithm which utilizes the inherent scale space behavior of words in gray-level document images.

The scalespace algorithm is compared with a gap-metrics algorithm on a modern database of multiple writers [19] and it is shown that, on a subset from this modern data set, the gap metrics algorithm performs slightly better (5 percent error) than the scalespace algorithm (9 percent error). On a different subset from the same data set, the two algorithms have comparable performance (13 percent error). However, on a database of 100 documents sampled from George Washington's historical manuscripts, the scalespace algorithm performs much better than the gap metrics algorithm (17 percent error versus 32 percent error). We also show that the scalespace algorithm works well on a 10 page set from Joseph Grinnell's field notes.

It is worthwhile briefly comparing this algorithm to a typical gap metrics algorithm for segmentation [20]. The gap metrics algorithm requires thresholding to produce a binary image (which is nontrivial for historical manuscripts). To segment lines into words, the gap metrics algorithm first divides lines into connected components, then computes the convex hulls of these connected components. Since some of these convex hulls may be characters, the gaps between the convex hulls are sorted and all gaps greater than some threshold are classified as word gaps—on the assumption that word gaps are longer than character gaps. The threshold varies with the line and the choice of how to select this threshold is one of the tricky issues in the algorithm. The scalespace algorithm presented here uses gray-level images. The line images are smoothed into blobs reminiscent of the convex hulls. The blobs correspond to character-like entities at smaller scales and to word-like entities at larger scales and the main issue is to select the appropriate scale so that the blobs correspond to words. The size of the convex hulls in a gap metrics algorithm is actually determined by the binarization step. In historical documents, the ink between characters in a word may sometimes be faded and this can lead to the binarization algorithm breaking the word into multiple units. On the other hand, in the scalespace algorithm, the size of the blobs is automatically adjusted to correspond to words by selecting the correct scale—thus, nearby characters usually coalesce into the same blob even if the ink connecting characters is faded.

This paper is organized as follows: Section 1.1 briefly discusses the problems in word segmentation. Section 2 reflects upon some work related to automatic word segmentation. This is followed by a section on the characteristics of the George Washington collection. Section 4 introduces some scale space theory. Section 5 discusses in detail the three different preprocessing techniques we employed to remove margins and horizontal lines. Section 6 discusses how we use projection analysis techniques to segment the document into lines. Section 7 briefly discusses the theory of scale selection and our automatic scale selection technique. Section 8 discusses postprocessing, including how we extract words from blobs. Section 9 discusses experiments and results using the different approaches to the scalespace algorithm and also compares the results of the scalespace algorithm to a gap metrics algorithm on both a modern database of multiple writers and on George Washington's manuscripts. We also show results on another small historical database.

1.1 Word Segmentation

Modeling the human cognitive processes to derive a computational methodology for handwritten word segmentation with performance close to the human visual system is quite complex due to the following characteristics of handwritten text. The handwriting style may be cursive or discrete. In the case of discrete handwriting, characters have to be combined to form words. Unlike machine printed text, handwritten text is not uniformly spaced. The size of characters in a header is generally larger than the average size of the characters in the body of the document—this is a scale problem. Ascenders and descenders are frequently connected and words may be present at different orientations. Documents are often degraded due to noise, artifacts, aging, or for other reasons. Another problem is the presence of background handwriting or bleed-through.

2 Related Work

The successes of offline handwriting have been in domains where the vocabulary is small or domain constraints can be applied—for example, bank check recognition and postal address recognition [22]. There is little work in offline handwriting on large vocabulary work (but see, for example, [21]). The existing work in this area focuses on modern handwriting. Many full-page offline handwriting recognition systems are usually tested by having pages specifically written and created for the purpose of testing these systems and, thus, are constrained in how they are created [19]. Real historical documents pose many challenging problems—including their age and condition, the uneven ink on the papers (George Washington's manuscripts were often written with a quill pen and so the ink even within a word is sometimes variable), and the variability introduced by real writers.

Thus, there is little work on full page segmentation, with most of the previous work in handwriting focused on specialized domains like postal addresses and bank checks. Indeed, even for modern writing, the problem of full page word segmentation can be a difficult one. For example, Marti and Bunke [21] describe their full-page recognition system on the IAM database [19] of modern writers. The IAM database consists of text copied with care by a large number of writers. A ruler was used to ensure that the lines are straight and horizontal and the ascenders from one line do not touch the descenders of another line. Even for this database, they [21] mention that their recognition system avoids the "difficult problem of segmenting a line of text into individual words." A number of other recognition schemes have also avoided the issue of segmentation by considering well-segmented patterns [1] or using words written in boxes whose location is known [4]. Senior and Robinson [27] describe how the pages are written so that the words are well separated and that the "segmentation algorithm takes a simple approach, looking for the gaps between lines and words." Casey and Lecolinet [3] and Plamondon and Srihari [24] provide surveys of the various segmentation and recognition schemes and Nagy [22] discusses papers published in *TPAMI* on document analysis during the last 20 years.

Most techniques today have focused on identifying gaps using geometric distance metrics between connected components, an approach known as gap metrics. Seni and Cohen [26] evaluate eight different distance measures between pairs of connected components for word segmentation in handwritten postal addresses. Mahadevan and Nagabushnam [14] and, more recently, Marti and Bunke [20] used the distance between the convex hulls to distinguish between character and word gaps. While Mahadevan and Nagabushnam's [14] evaluation was done on postal addresses, Marti and Bunke's [20] evaluation is done using a full-page word segmentation algorithm on a database of modern writers [19]. To the best of our knowledge, their evaluation is the largest evaluation on a full-page database of modern writers. Kim et al. [10] present techniques for line separation and then word segmentation using a neural network. Feldbach and Tonnies present a system in [5] using constraints on the semantics to segment the date from church registers using a neural network.

Our main emphasis is on the George Washington collection and, so, it is helpful to briefly mention the characteristics of this and similar historical collections.

3 Characteristics of the George Washington Collection

The data set used here consists of 100 page images from the George Washington collection sampled from different portions of the original collection (at the Library of Congress) which has roughly 140,000 page images scanned at 300 dpi. The papers were scanned from microfilm, presumably for reasons of cost and security (we had no control over the scanning). There are multiple writers in the collection since Washington had aides who helped him write significant portions of the text. However, we do not know who wrote what. The scans are of varying quality with problems, including bleed-through, blotches, and faded ink, all present in varying degrees in different pages. Some of these problems arise because of how the manuscripts were written (e.g., the use of a quill pen sometimes causes the ink within even a word to slowly fade), how they aged, and also how they were preserved or scanned. Ascenders and descenders from adjacent lines may sometimes touch. Given their significance, we may assume that this collection is cared for better than most other historical manuscripts.

The writing in the Washington papers is roughly straight and horizontal. The actual skew of a line in the 100 page collection varies up to about 2 deg (this implies that a line's vertical position may change by as much as 0.3 inches over the 8 inches width of a page). Many (but not all) historical documents have their writing along straight roughly horizontal lines with small skew since many of these documents were originally written to be read by others and there were no alternative printed or typewritten versions. Our assumption will, therefore, be that the words are approximately written along straight horizontal lines (we will show that our algorithm can tolerate the skew in the George Washington collection without any problems). One could handle larger amounts of skew by using a standard skew detection algorithm [2], [30], but it is not necessary for any of the collections we discuss in this paper. In principle, it may even be possible to handle curved lines by first detecting the curved line, warping it to a straight line, and then using the word segmentation algorithm presented here, but this is beyond the scope of the work here.

4 Scale Space and Document Analysis

Scale space theory deals with the importance of scale in any physical observation, i.e., objects and features are relevant only at particular scales. In scale space, starting from an original image, successively smoothed images are generated along the scale dimension. Koenderinck and van Doorn [11] showed that the Gaussian uniquely generates the linear scale space under the conditions of causality, isotropy, and homogeneity (essentially these conditions lead to the diffusion equation whose unique solution is the Gaussian).[2] A number of other researchers [6], [13] have shown equivalent formulations which show that the Gaussian still uniquely generates the scale space.

We feel that *scale space* also provides an ideal framework for document analysis. We may regard a document to be formed of features at multiple scales. Intuitively, at a finer scale, we have characters and, at larger scales, we have words, phrases, lines, and other structures. Hence, we may also say that there exists a scale at which we may derive words from a document image. We would, therefore, like to have an image representation which makes the features at that scale (words in this case) explicit. The linear scale space representation of a continuous signal with arbitrary dimensions consists of building a one parameter family of signals derived from the original one in which the details are progressively removed. Let $f : \Re^2 \rightarrow \Re$ represent any given signal. Then, the scale space representation $I : \Re^2 \times \Re_+ \rightarrow \Re$ is defined by (see [13]) letting the scale space representation at zero scale be equal to the original signal $I(\cdot; 0) = f$ and, for $t > 0$,

$$I(\cdot; t) = G(\cdot; t) \star f, \tag{1}$$

where $t \in \Re_+$ is the scale parameter and G is the Gaussian kernel which, in two dimensions $(x, y \in \Re)$, is written as

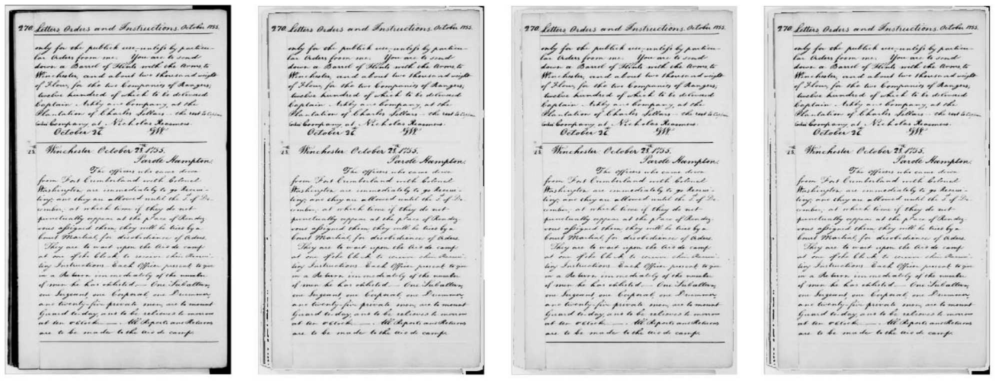
$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{(2\sigma^2)}}, \tag{2}$$

where $\sigma = \sqrt{2t}$. We now describe the various stages in our algorithm.

5 Preprocessing

These handwritten manuscripts have been subjected to degradation such as fading and other artifacts. The images provided to us are scanned versions of microfilm of the original manuscripts. During this conversion, horizontal and vertical black line segments and margins were introduced. Horizontal lines are also present within the text. The purpose of the preprocessing step is to remove some of these margins and lines so that they will not interfere with the later stages.

2. Removing the isotropy condition leads to anisotropic Gaussians.



(a) (b) (c) (d)

Fig. 3. Margins and horizontal lines removed by different techniques. (a) Original image. (b) LOG filtering. (c) Projection profile. (d) Hough transform.

We used three different techniques to detect and remove these margins and compared it with a baseline where the margins were not removed.

1. No Preprocessing: Baseline (lines were not removed).
2. LOG Filtering: Create "blobs" from the entire document using a Laplacian of Gaussian filter and then remove ones which are very large or very long.
3. Projection Profile: Detect peaks in vertical projection profiles (only removes margins).
4. Hough Transform: Use a Hough transform to detect lines.

After detecting the lines using one of the above techniques, the image is binarized. A connected component analysis is run and those components which overlap the chosen lines are mapped back to the intensity image and replaced with the background color (effectively removing these lines). Fig. 3 shows examples of lines removed by each of these three methods.

The following three sections describe these preprocessing techniques in more detail.

5.1 Line Removal Using LOG Filtering

Marr and Hildreth [18] have shown that the intensity changes (edges) are given by the zero crossings of the Laplacian of Gaussian. The sign of the Laplacian may then be used to classify the image into foreground—positive values indicating text and lines—and background indicated by negative values. Therefore, the document image is smoothed and convolved with a LOG filter with a small scale value to preserve all the image structures:

$$I(x, y; \sigma) = \nabla^2 G(x, y; \sigma) \star f(x, y). \tag{3}$$

The Laplacian is then thresholded on the positive values to get a binary image. We observe that these line segments/margins are located only on the periphery of the page. Therefore, we use a connected component analysis over the first and last 20 percent columns to detect margins. A similar analysis is then done on the image rows to detect horizontal lines. The remaining horizontal lines can be removed in the subsequent steps. The line segments are eliminated based on aspect ratio, height, and length filters.

5.2 Margin Removal with Projection Profiles

Projection profiles, sometimes called projection histograms, have been used widely in document imaging [29], [24]. However, most of this work uses binary images, while we use gray-scale images. Computing a projection profile is much faster than using a LOG filter or a Hough Transform. Let $f(x, y)$ be the intensity value of a pixel (x, y) in a gray-scale image, I. Then, we define the vertical projection profile as

$$P(x) = \sum_{x=0}^{W} f(x, y), \tag{4}$$

where W is the width of the image. We use the projection profile to determine the amount of ink in a column. More ink is represented by a local minimum in the projection profile. We locate all local minima and then threshold to detect the margins.

5.3 Line Removal with the Hough Transform

The Hough Transform, introduced by P.V.C. Hough [9], is a well-known tool for detecting shapes in images, such as straight lines. The Hough Transform operates on the edge image by accumulating counts for the parameters of a line in polar coordinates. For our implementation, we considered the highest 70 percent of the values in the Hough Accumulator to be lines.

6 Line Segmentation

The George Washington manuscripts contain lines which are approximately straight and close to horizontal. Projection profile techniques have been widely used in line and word segmentation for machine printed documents [8]. In this technique, a 1D function of the pixel values is obtained by projecting the binary image onto the horizontal or vertical axis. We use a modified version of the same algorithm extended to gray-scale images (see (4)).

Fig. 4a shows an image and Fig. 4b shows its projection profile. The distinct local peaks in the profile correspond to the white space between the lines and distinct local minima correspond to the text (black ink). Line segmentation, therefore, involves detecting the position of the local

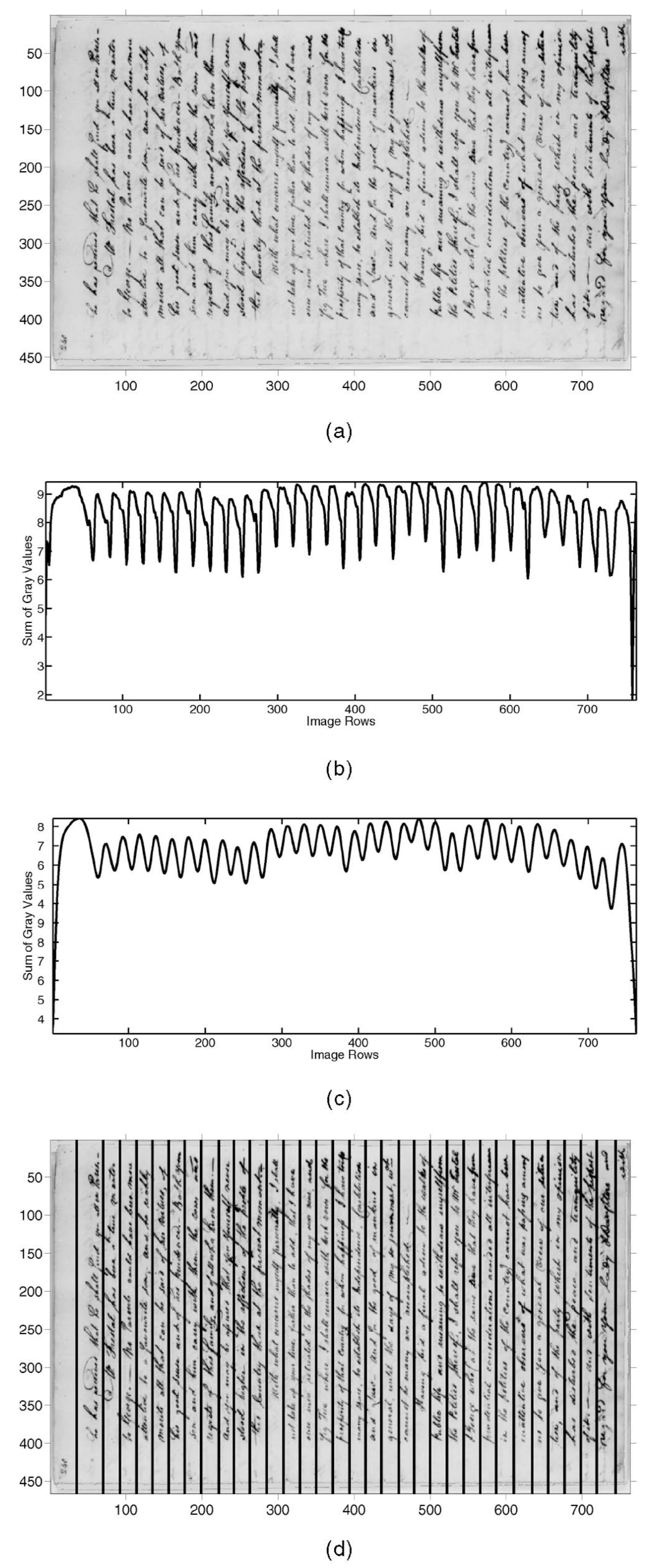


Fig. 4. Line segmentation steps. (a) A rotated image. (b) Projection profile. (c) Smoothed projection profile. Note unique peaks. (d) Line segmentation by finding peaks in the smoothed projection profile.

maxima. However, the projection profile has a number of false local maxima and minima. The projection function $P(y)$ is therefore smoothed with a Gaussian (low pass) filter to eliminate false alarms and reduce sensitivity to noise. This smoothed profile is shown in Fig. 4c. The local minima (troughs) of the smoothed projection profile correspond roughly to the mid-points of the lines and the local maxima (peaks) to the space between lines. The local maxima may be obtained by setting the derivative of the projection profile to zero. Since convolution is a linear operation, the smoothing and the derivative operation may be combined into one step by convolving the projection profile with a Gaussian derivative. That is,

$$d/dy * G(y;\sigma) * P(y) = \frac{dG(y;\sigma)}{dy} * P(y). \tag{5}$$

Finally, the lines are segmented at the peaks as shown in Fig. 4d. The line segmentation technique is robust to variations in the size of the lines and has been tested on a wide range of handwritten pages. After segmenting these lines, we begin our word segmentation by creating a scale space of the line images for blob analysis.

7 Blob Analysis

Now we examine each line image individually to extract the words. A word image is composed of discrete characters, connected characters, or a combination of the two. We would like to merge these subunits into a single meaningful entity which is a word. This may be achieved by forming a blob-like representation of the image. A blob can be regarded as a connected region in space. One way of forming a blob is to use a Laplacian of a Gaussian (LOG) [13]. We have used a differential expression similar to a LOG for creating a multiscale representation for blob detection. However, our differential expression differs in that we combine second order partial Gaussian derivatives along the two orientations at different scales. In the next section, we present the motivation for using an anisotropic derivative operator.

7.1 Nonuniform Gaussian Filters

In this section, some properties which characterize writing are used to formulate an approach to filtering words. In [13], Lindeberg observes that maxima in scale occur at a scale proportional to the spatial dimensions of the blob. If we observe a word, we may see that the spatial extent of the word is determined by the following:

1. The individual characters determine the height (y dimension) of the word and
2. the length (x dimension) is determined by the number of characters in it.

A word generally contains more than one character and has an aspect ratio greater than one. As the x dimension of the word is larger than the y dimension, the spatial filtering frequency should also be higher in the y dimension as compared to the x dimension. This domain-specific knowledge allows us to move from isotropic (same scale in both directions) to anisotropic operators. We choose the x dimension scale to be larger than the y dimension to correspond to the spatial structure of the word. We define the anisotropic Gaussian filter as

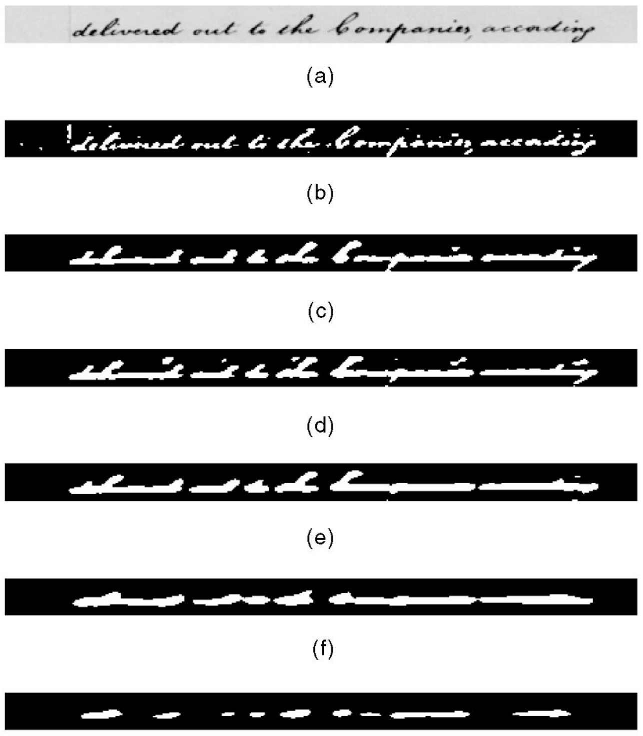


Fig. 5. A line image and the corresponding blobs at different scales. At low scales, blobs correspond to character-like entities—(a). With increasing scale, they correspond more to word-like entities. (a) A line image. (b) Blobs: $\sigma_y = 1, \sigma_x = 2$. (c) Blobs: $\sigma_y = 2, \sigma_x = 4$. (d) Blobs: $\sigma_y = 2, \sigma_x = 8$. Blobs correspond to words. (e) Blobs: $\sigma_y = 2.55$, $\sigma_x = 10.2$. Blobs correspond to words. (f) Blobs: $\sigma_y = 4, \sigma_x = 16$. (g) Blobs: $\sigma_y = 5, \sigma_x = 20$.

$$G(x, y; \sigma_x, \sigma_y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2}\right)}. \qquad (6)$$

We may also define a multiplication factor η by $\eta = \frac{\sigma_x}{\sigma_y}$.

All of the handwritten documents available to us are written in English and read from left to right. For the above Gaussian, the second order anisotropic Gaussian differential operator $L(x, y; \sigma_x, \sigma_y)$ is defined as

$$L(x, y; \sigma_x, \sigma_y) = G_{xx}(x, y; \sigma_x, \sigma_y) + G_{yy}(x, y; \sigma_x, \sigma_y). \qquad (7)$$

A scale space representation of the line images is constructed by convolving the image with L from (7). Consider a two-dimensional image $f(x, y)$; then, the corresponding output image is

$$I(x, y; \sigma_x, \sigma_y) = L(x, y; \sigma_x, \sigma_y) \star f(x, y). \qquad (8)$$

The main features which arise from a scale space representation are blob-like (i.e., connected regions either brighter or darker than the background). The sign of I may then be used to make a classification of the 3D intensity surface into foreground and background. For example, consider the line image in Fig. 5a. The figures show the blob images $I(x, y; \sigma_x, \sigma_y)$ at increasing scale values. Fig. 5b shows that, at a lower scale, the blob image consists of character blobs. As we increase the scale, character blobs give rise to word blobs (Fig. 5d and Fig. 5e). This is indicative of the phenomenon of merging in blobs. It is seen that, for certain scale values, the blobs and, hence, the words are correctly delineated (e.g., Fig. 5e). A further increase in the scale value may not necessarily cause word blobs to merge together and other phenomenon such as splitting is also observed (this can happen in higher dimensions—see [13]). These figures show that there exists a scale at which it is possible to delineate most words. In the next section, we present an approach to automatic scale selection for blob extraction.

7.2 Choice of Scale

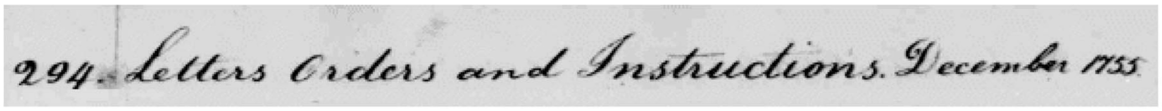
Scale selection is a difficult problem. The solution to this problem depends on the particular application and requires the use of prior information to guide the scale selection procedure. Lindeberg [13] in his classic work on scale selection argued that the maximum response in both scale and space is obtained at a scale proportional to the dimension of the object and, therefore, that the optimal scale of a structure may be determined by looking at the extremum of some quantity or feature. This quantity or feature must be determined for the particular domain or application.

A document image consists of structures such as characters, words, and lines at different scales. There is an appropriate scale associated with the characters, the words, and the lines. Hence, there exists a scale where each of the individual words form a distinct blob and this scale may be determined using Lindeberg's argument by looking for maxima of a particular quantity.

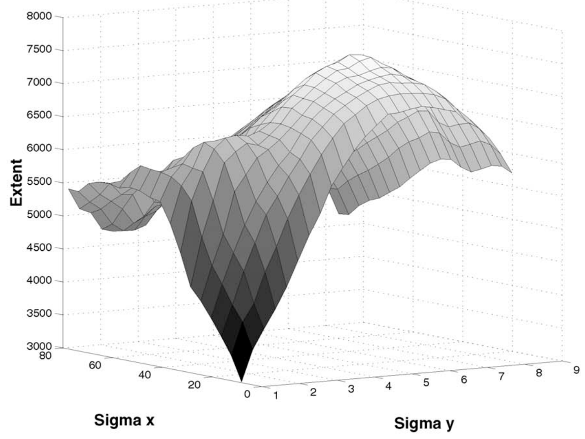
The two parameters σ_y and σ_x capture the spatial scales of a word. Using Lindeberg's criterion, our aim is to find the values of these two scales such that the spatial extent (area) of blobs for a line is maximized. To measure the variation in spatial extent of the blobs over scale, we define ζ_i to represent the extent (area) of a blob i. Then, the total extent of the blobs A for a line is given by $A = \sum_{i=1}^{n} \zeta_i$. Observation reveals (in accordance with Lindeberg's general theory) that, if the scales at which the extent of the blob is maximum are used to filter the line images, the resulting blobs correspond to words. An example is shown in Fig. 6 for a particular line of text. The extent of the blob is plotted against σ_x and σ_y. Note that there is a maximum in the three-dimensional plot—which corresponds roughly to a value of $\sigma_y = 6$ and $\sigma_x = 24$. The figure also shows a cross-section of the three-dimensional plot obtained by keeping the quantity $\eta = \sigma_x/\sigma_y$ constant at a value of 4. It is observed that, if these scale values of the blobs are used, then the resulting blobs correspond to words. This observation has been repeated over many documents and shows that the maximum of the spatial extent (area) of the blobs corresponds to the best scale for filtering.

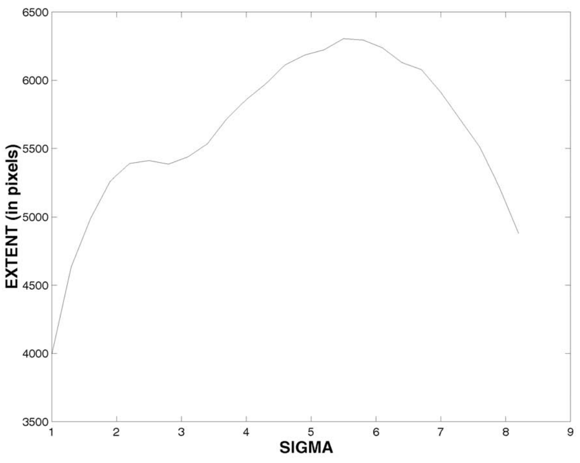
A two parameter optimization problem is very expensive and, therefore, it would be useful to simplify this to maximizing a one-dimensional problem. We had earlier defined the factor η as $\eta = \sigma_x/\sigma_y$. If we can fix η, then the problem reduces to the one-dimensional problem of determining σ_y.

Fig. 6 also shows the spatial extent of the blobs as a function of σ_y for a fixed value of $\eta = 4$. The plot again shows that the extent is maximized when $\sigma_y = 5.6$. Filtering the line image with $\sigma_y = 5.6$ and $\sigma_x = \sigma_y * \eta = 22.4$ also


294. Letters Orders and Instructions. December 1755

(a)





(b) (c)

Fig. 6. Three-dimensional plot of spatial extent of the blobs against σ_x and σ_y and a cross-section corresponding to the line image shown in (a). (a) A line image. (b) Extent versus σ_y, σ_x for the line image shown in (a). (c) Extent versus σ_y for constant η ($\eta = 4$).

gives blobs corresponding to words. This suggests that we can simplify the optimization into a one-dimensional problem by fixing η and then estimating σ_y. Note that the response of filters at nearby scales is highly correlated, so the scale parameter is only moderately sensitive, i.e., one does not need to pick the exact maximum. Instead, a range of scales around the maximum is sufficient to perform good word segmentation (see Figs. 5d and 5e).

Our aim is to fix η by optimizing it for the corpus and then solve the problem of maximizing σ_y for each line. The rationale for this approach is that the scales in the x and y direction are very strongly correlated. This is similar to the word height normalization idea in document recognition —words are scaled to be the same height, the presumption being that the aspect ratio of a given word doesn't change a lot.

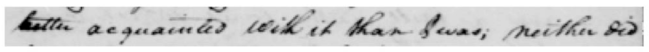
7.3 Selecting η

We can estimate a value of η in a couple of ways. We first note that η is related to the aspect ratio of a word. The maximum value of η will then be close to the average aspect ratio of a word. A manual analysis of several images shows that the average aspect ratio of a word in a document image lies in the range 3.0-5.0. A second analysis of several images reveals that, for constant σ_y, the maxima in extent is obtained for η lying in the range between 3-5. A line image and the corresponding plot are shown in Fig. 7, where the scale maximum obtained is approximately 3.5.

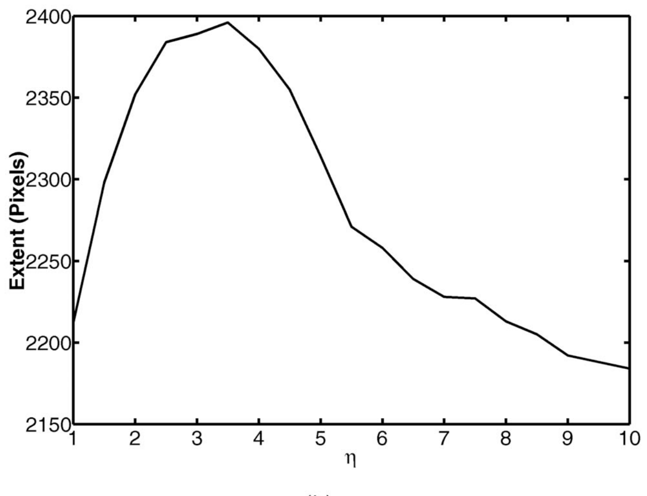
This analysis, along with the observation that the average aspect ratio of the word is between 3-5, allows us to choose a value of η in the range 3-5. Specifically, for further analysis, we choose $\eta = 4$. As noted above, the blob detection process is not highly sensitive to scale and this also means that using the same value of η for the corpus is sufficient.

7.4 Selecting σ_y

Given η, the problem reduces to determining σ_y for each line. The maximum at which σ_y occurs can vary with line height. Fig. 6 showed the plot of extent versus σ_y at constant η for a line with large height. Fig. 8 shows a line image of smaller height and a plot of extent versus σ_y for constant $\eta = 4$. It is observed that there is a maximum, but the maximum occurs at a lower scale since the line height is smaller. In fact, there is a close correspondence between the

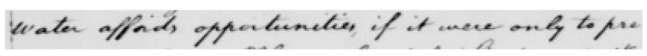

better acquainted with it than I was; neither did

(a)

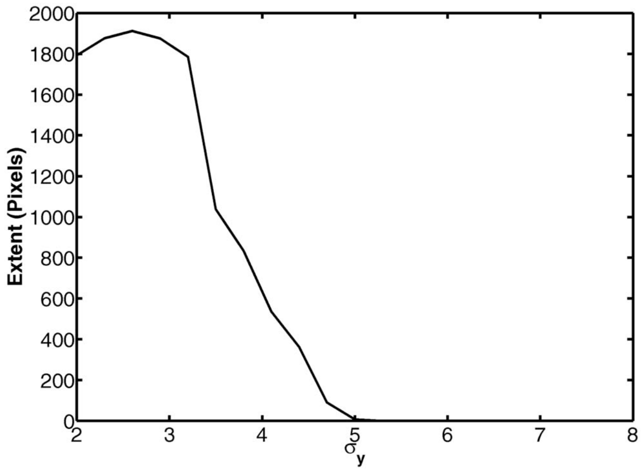


(b)

Fig. 7. Variation of blob extent versus η with constant σ_y. (a) A line image. (b) Plot of extent versus η, $\sigma_y = 2$ for the above image. The maximum is obtained at $\eta = 3.5$.



(a)



(b)

Fig. 8. Variation of extent versus σ_y for constant η. (a) A sample line image with smaller height. (b) Plot of extent versus σ_y with constant $\eta = 4$. Maximum is obtained at $\sigma_y = 2.6$.

line height and the maximum value of σ_y. Experimentally, it can be shown that σ_y is a function of the height of the words (which is related to the height of the line).

σ_y may be estimated in a number of different ways and, here, we propose three possible approaches to doing it. They are:

- *Free Search* searches the scale space of σ_y. This is done by sampling for the maximum over a wide range of scales $1 < \sigma_y < 8$ at intervals of 0.3. The range was chosen experimentally so that all maximums lay within it. Searching within such a large range proved to be computationally expensive with each scale adding about 4 seconds per document (on a 500 Mhz Xeon processor). Note that the implementation is not optimized.
- *Estimation by line height* uses the fact that the optimal value of σ_y is closely correlated with the average line height. We estimate a base σ_y as a function of the line height, i.e., $\sigma_y = k \times \text{Line height}$, where $0 < k < 1$. Three values, σ_y, $\sigma_y - 0.3$, and $\sigma_y + 0.3$, are plotted versus extent. The maximum value is chosen for the line.
- *Page Averaging* assumes that the line heights do not vary much within a handwritten page. Furthermore, it assumes that the system may select a suboptimal σ_y for some lines, but the average will be close to optimal. In this method, we obtain an estimate of σ_y for each line using "estimation by line height." We then choose one value, the mean of the σ_ys for the entire document.

Note that the latter two techniques are much faster and the hope is that page averaging will estimate the scale more robustly. Section 9.2 discusses the results for the three different ways of selecting σ_y.

8 Blob Extraction and Postprocessing

After choosing the correct scale and creating blobs, the blobs are then mapped back to the original image to locate the words. The blob is bounded using a rectangular box which can be obtained through connected component analysis. In a blob representation of the word, parts of the words, especially the ascenders and descenders, are lost due to the earlier operations of line segmentation and smoothing (blurring). Therefore, the above bounding box is extended in the vertical direction to include these ascenders and descenders. At this stage, an area/aspect ratio filter is used to remove or combine small structures which result from scale selection errors and noise.

9 Experiments and Results

The technique we present was originally designed to segment degraded, historical documents written in English. The algorithm was developed and tested on 100 images from different sections of the George Washington corpus of 140,000 images (see Section 3 for more on this collection). The aforementioned 100 documents contained images of quality which ranged from good to so degraded that it is difficult for humans to read. Results and discussion from these tests are found in Section 9.2.

Our algorithm has also been tried on the writing of other authors, such as the field notes of biologist Joseph Grinnell. An example of good segmentation results on an image by this author is seen in Fig. 10. We show results on the 10 pages of the Grinnell collection available to us.

Additionally, we tested our algorithm on selected images from the IAM database, a modern multiauthor database of handwritten documents created specifically for handwriting recognition work and described in [19]. This is the same database that Marti and Bunke used to test their algorithm in [20]. We also compared our algorithm with a state-of-the-art segmentation algorithm by implementing the gap-metrics-based algorithm described in [20] by Marti and Bunke. Quantitative results and comparisons to the gap-metrics-based algorithm are shown in Section 9.3. We show there that our algorithm outperforms the gap metrics algorithm on George Washington's data set by a substantial margin.

9.1 An Automatic Bounding Box Evaluator

Our data set consists of 100 pages of George Washington's manuscripts. In addition, we would like to evaluate the different algorithms we have proposed (we explore 12 different combinations later). This would be very difficult and tedious to do manually. Ground truth was created for the George Washington data set and an automatic bounding box evaluator used to quickly access results and to make it possible to test the different combinations of algorithms on the 100 documents. Here, we discuss our automatic evaluator and show that the results obtained correlate well to those obtained by manual evaluation on a small subset of the pages. For the purpose of brevity, we will call the manually corrected bounding boxes "template-boxes" and the automatically generated bounding boxes (those generated by the algorithm presented in this paper) "test-boxes" in the remainder of this section.

TABLE 1
Results without Postprocessing

Preprocessing method	Scale Selection method	% Words missed	% Over segmented	% Under segmented	% Margin overlap	% Total errors	Time seconds
No Preprocessing	Estimation by line height	0.7	14.9	6.0	2.5	24.0	≈ 23
	Page averaging	0.5	11.7	9.9	3.0	25.0	≈ 26
	Free search	0.8	6.9	18.9	5.5	32.2	≈ 159
Hough Transform	Estimation by line height	0.8	15.3	5.6	0.0	21.6	≈ 29
	Page averaging	0.6	11.9	9.5	0.0	22.1	≈ 35
	Free search	1.0	7.5	18.0	0.0	26.5	≈ 168
Projection Profile	Estimation by line height	0.8	15.1	5.7	0.0	21.6	≈ 20
	Page averaging	0.6	11.7	9.7	0.0	21.9	≈ 26
	Free search	1.0	7.5	17.9	0.0	26.4	≈ 158
LOG Filtering	Estimation by line height	0.8	15.2	5.6	0.0	21.6	≈ 45
	Page averaging	0.6	11.9	9.6	0.0	22.0	≈ 48
	Free search	0.9	7.5	17.9	0.0	26.3	≈ 188

TABLE 2
Segmentation Results with Postprocessing

Preprocessing method	Scale Selection method	% Words missed	% Over segmented	% Under segmented	% Margin overlap	% Total errors	Time secs
No Preprocessing	Estimation by line height	0.8	8.9	7.7	2.4	19.9	≈ 35
	Page averaging	0.7	7.0	11.2	2.9	21.8	≈ 46
	Free search	1.0	4.6	19.7	5.3	30.5	≈ 188
Hough Transform	Estimation by line height	1.0	9.1	7.4	0.0	17.4	≈ 65
	Page averaging	0.8	7.2	10.9	0.0	18.9	≈ 75
	Free search	1.1	4.9	18.8	0.0	24.8	≈ 227
Projection Profile	Estimation by line height	0.9	8.9	7.6	0.0	17.5	≈ 36
	Page averaging	0.7	7.0	11.1	0.0	18.8	≈ 46
	Free search	1.0	4.8	18.9	0.0	24.7	≈ 200
Log Filtering	Estimation by line height	1.0	9.0	7.4	0.0	17.4	≈ 49
	Page averaging	0.7	7.1	11.0	0.0	18.9	≈ 58
	Free search	1.1	4.8	18.7	0.0	24.6	≈ 208

The program works as follows:

1. For each template-box, find the test-boxes which were segmented from the same line.
2. If two boxes overlap by some threshold, say 60 percent, then assign a "hit" to that template-box and a "hit" to the test-box. Several different thresholds were tested and 60 percent produced the best estimation to manual evaluation.
3. After iterating over all template boxes, evaluate the "hits" as follows.
4. All template-boxes with two or more hits have been oversegmented.
5. All test-boxes with two or more hits indicate that the corresponding template-boxes have been undersegmented.
6. All template-boxes with zero hits have been missed.

We evaluated the correctness of this program by manually evaluating 10 randomly chosen documents from the 100 page George Washington data set. The two evaluators mostly concurred. The largest discrepancy between the human and automatic evaluators in each category was under 3 percent, which we found to be reasonable. In almost all the cases, the automatic evaluator assigns higher errors than the human. This leads us to believe that the errors computed later for the 100 page data set in Tables 1 and 2 are overestimated.

9.2 Results on the George Washington Corpus

The technique was tested on 100 page images sampled from different sections of the George Washington corpus of $140,000$ page images. To reduce the runtime, the images have been smoothed and subsampled to a quarter of their original size.

The three different algorithms for estimating scale were tested with the three different techniques for removing the margins on 100 pages from the George Washington data set. The algorithms were also tested without removing the margins. This gives 12 different combinations in all. The automatic evaluator previously described (Section 9.1) was used for evaluation Tables 1 and 2 show, respectively, the results for the case where no postprocessing is done and postprocessing is done. In the tables, missed words are those for which no bounding box was generated. Oversegmentation occurs when two or more bounding boxes are generated for one word. Undersegmentation occurs when two or more words lie within one bounding box. Since the ground truth is generated on a per word basis, three words in a box count as three errors. Margin overlap occurs when a bounding box extends over the black margins on the sides of the images. The total errors column is the sum of errors in the other columns. The last column is the cpu time taken on a 500 Mhz Pentium processor.

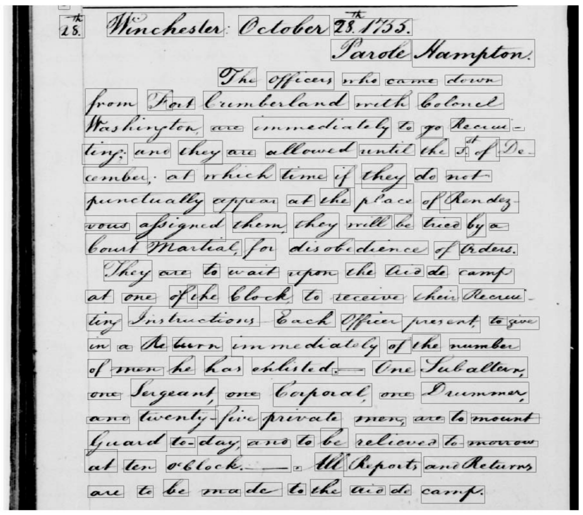

25th Winchester October 28th 1755.
Parole Hampton.
The officers who came down
from Fort Cumberland with Colonel
Washington are immediately to go Recrui-
ting; and they are allowed until the 1st of De-
cember, at which time if they do not
punctually appear at the place of Rendez-
vous assigned them, they will be tried by a
Court Martial, for disobedience of Orders.
They are to wait upon the Aid de camp
at one of the clock to receive their Recrui-
ting Instructions. Each Officer present, to give
in a Return immediately of the number
of men he has enlisted. — One Subaltern,
one Sergeant, one Corporal, one Drummer,
and twenty-five private men, are to mount
Guard to-day, and to be relieved to-morrow
at ten o'clock. — All Reports and Returns
are to be made to the Aid de camp.

Fig. 9. Scale space segmentation on part of an image from the George Washington collection.

Using some kind of preprocessing to remove the margins helped to improve results for all techniques—the most dramatic improvements being for the free search technique. The main improvement resulted from being able to avoid margin errors—that is, where the box includes some portion of the black margin. The three different preprocessing techniques produced more or less similar results.

For all techniques which used preprocessing, the total error rate was between 17.4 percent and 24.9 percent when postprocessing was used. When postprocessing was not used, the error rate was between 21.6 percent and 26.5 percent. The best technique was estimation by line height, closely followed by the page averaging technique. The free search technique was consistently worse by a few percent. It is interesting to note that the free search technique actually produced fewer oversegmentation errors, but produced a lot more undersegmentation errors. We see that, like other document processing problems, using additional constraints improves the results—sometimes, the scale computed may be erroneous due to a variety of reasons. For example, the line may be very short and consist of one or two words, in which case, the scale is difficult to compute. In such cases, the constraint on scales produced by "estimation by line height" or "page averaging" helps. The tables also show that it is not very common for words to be missed (not have a box associated with them) and that postprocessing does help.

The error rate for each document varies from around 6.3 percent for better quality documents and up to 33.7 percent for documents of such poor quality that they are even hard for humans to read. Fig. 9 shows part of a segmented page image with bounding boxes drawn around the extracted words.

On a small data set of 10 pages of Joseph Grinnell's writings, the segmentation error is of the order of 8 percent. The Grinnell data set is of better quality and this is reflected in the numbers, as shown in Table 3. Fig. 10 shows an example on a page from the Grinnell data set. Note that no parameters have been changed from the previous experiment (we use estimation by line height with projection

TABLE 3
Results for the Scale Space Algorithm on 10 Pages from the Grinnell Collection

Missed	Over Segmented	Under Segmented	Total Errors
0.5	0.1	7.2	7.8

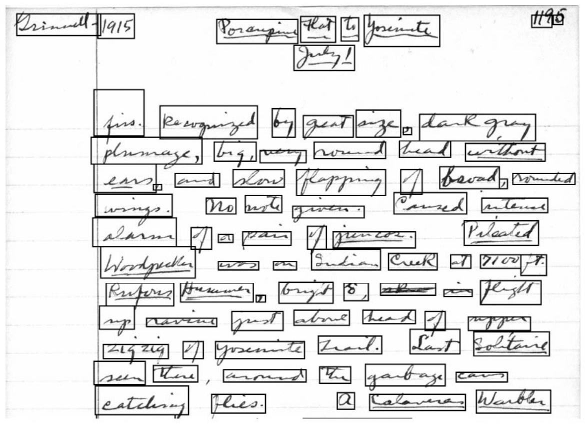

Grinnell - 1915 Porcupine Flat to Yosemite 1195
July 1
fis. Recognized by great size, dark gray
plumage, big, very round head without
ears, and slow flapping of broad, rounded
wings. No note given. Caused intense
alarm of a pair of juncos. Pileated
Woodpecker was on Indian Creek at 7100 ft.
Rufous Hummer, bright ♂, ~~also~~ in flight
up ravine just above head of upper
zigzag of Yosemite Trail. Last Solitaire
seen there, around the garbage cans
catching flies. a Calaveras Warbler

Fig. 10. Scale space segmentation result on one of the field notes of Joseph Grinnell.

profiles for preprocessing and postprocessing is also used). Our focus here is on segmentation of English text. However, the same approach can, in principle, be tried in other languages. In the next section, we compare our algorithm with a standard gap metrics-based approach.

9.3 Testing on Documents by Different Authors and Comparison to a Gap Metrics-Based Segmentation Algorithm

In order to compare our algorithm against the Gap Metrics approach, we implemented a modified version of the segmentation algorithm presented by Marti and Bunke [20]. The procedure for the original algorithm is as follows:

1. Segment the document into lines.
2. Binarize each line using Otsu's algorithm [23] and extract the connected components.
3. Compute the convex hull and centroid for each of these components.
4. For each pair of connected components, c_1 and c_2, consider the straight line s that connects the two centroids.
5. The distance d, between the two points where s intersects the convex hull of c_1 and c_2 is computed and assigned as a weight to s.
6. A completely connected and weighted graph is obtained. Given this graph, its minimum spanning tree is computed.
7. The following function is constructed from properties of the current line and the threshold t_{seg} is determined from this function:

$$t_{seg} = \alpha \frac{w_l - w_s}{d_s}, \tag{9}$$

where w_l and w_s are the line width and median stroke width, respectively, d_s is the median distance

TABLE 4
Comparison of Segmentation Results on the IAM-C Data Set

Technique	Over Seg.	Under Seg.	Total Errors
Gap metrics	2.7	2.9	5.6
Scale Space	1.2	8.3	9.5

between any two vertical strokes, and α is a constant that needs to be experimentally determined.

8. Each edge in the minimum spanning tree whose weight is greater than the the t_{seg} are removed, creating a series of subtrees.
9. Each one of these subtrees denotes a word and we extract the minimum and maximum row and column coordinates to create our bounding boxes. We then segment the bounding boxes.

We replaced Marti and Bunke's thresholding algorithm (Step 7) with one based on taking a histogram of the distances between the connected components and thresholded this histogram at the first valley going from left to right. The assumption is that there is a peak gap for characters followed by a valley and then the gap lengths for words. This produced a much more stable algorithm, avoided the need for tuning the α parameter, and, in general, produced much better results over different data sets. To make the comparison fair, we used our approach for segmenting a page into lines. For the George Washington manuscripts, we also used the same preprocessing technique for removing margins.

Marti and Bunke [20] tested their algorithm on a set of 59 documents from the IAM [19] database. To the best of our knowledge, this is the largest test of the gap metrics algorithm on full page handwritten material. The IAM database is a multiwriter collection of handwritten documents which is split up into several sections. The writers were given a form or forms to copy. They were also instructed to use a ruler to write on straight horizontal lines. The quality of these documents, while somewhat variable between sections, is, therefore, much better than those in the George Washington database. In [20], 59 images (c03-000[a-f]) are used to test their gap-metrics segmentation system. The handwriting in most of these 59 images is of some of the best quality in the database. On this data set, Marti and Bunke's [20] best results obtained by tuning parameters for this data set were around 4.4 percent. Our implementation produces an error of about 5.6 percent which is reasonably close to their value and a good verification of the implementation of the gap metrics algorithm.[3] For comparison, the scalespace algorithm produced an error of 9.5 percent (detailed numbers are provided in Table 4). We used the same parameters for the scalespace algorithm as for the George Washington data set (although tuning could have improved numbers substantially) since our aim is to use the same algorithm over a range of historical data sets. Neither algorithm suffered from missed-word errors or margin overlap (since margins did not exist), so these were not counted.

We also tested both algorithms on the first section (IAM-A) of 24 pages of the IAM database. On this section, both algorithms performed worse and their performance is about the same (13.3 percent for gap-metrics versus 13.5 percent for the scalespace algorithm). IAM-A is somewhat more challenging than IAM-C and this is confirmed by the fact that both algorithms perform worse, as shown in Table 5.

TABLE 5
Comparison of Segmentation Results on the IAM-A Data Set

Technique	Over Seg.	Under Seg.	Total Errors
Gap metrics	8.0	5.3	13.3
Scale Space	4.4	9.1	13.5

TABLE 6
Results for a Gap-Metrics Algorithm on the George Washington Data Set

Missed	Over Seg.	Under Seg.	Margin	Total Errors
0.4	18.1	13.3	0.1	31.9

We ran the gap-metrics algorithm on the same set of 100 documents from the George Washington database that were tested with the scale-space algorithm. In this test, the scale-space algorithm performed substantially better (see Table 6) than gap-metrics (compare the 32 percent total error of gap metrics with the 17 percent total error obtained by the scalespace algorithm). We see that gap-metrics had a lower rate of undersegmentation, but the oversegmentation was quite severe. In examining the individual documents, we found that gap-metrics performed especially poorly on documents where the ink was faded and letters did not connect clearly or there were other artifacts like bleed-through and noise. Our intuition for this poor performance is that gap-metrics suffers from the weakness that it requires good binarization. Since they are degraded and noisy and the ink may be faded, it is difficult to binarize historical documents consistently. What results is an incorrect binarization that causes many light character-connecting ink pixels to be turned off and a large number of small connected components are extracted. Fig. 11 shows an example of this problem. The connections between some of the characters in the second line are faint and are eliminated by the binarization step, resulting in multiple connected components for a given word. This kind of problem does not usually arise in IAM or other modern databases. On the first line, the faint flourish before the word "Each" connects the previous word and is darker than some of the character connections in the second line. If the binarization were changed to correct for the character problem in the second line, it is likely to end up connecting the flourish in the first line. Thus, one kind of error is traded off for another kind of error. The problem becomes even more serious when bleed-through and other artifacts are present. The end result is that the threshold t_{seg} is not correctly estimated. It is theoretically possible that more expensive binarization algorithms than Otsu's algorithm may improve performance, but, as Nagy [22] points out, ". . . it may be time to concede that many document images cannot be binarized without extensive gray-level analysis (that undermines the economies of binarization)." The real answer is that one needs to deal with such documents using gray-level

3. The small difference in numbers could also possibly be due to minor variations in how errors are counted.

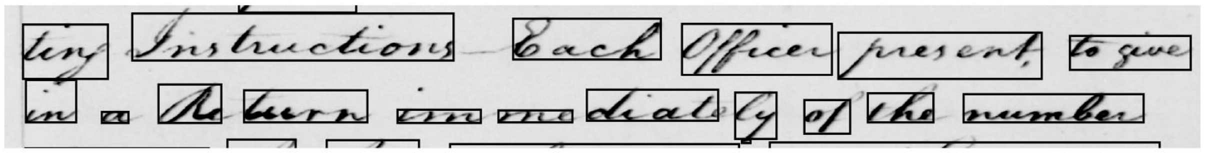


Fig. 11. Gap-metrics segmentation result on a portion of a Washington image (same image as in Fig. 9). Note the faint flourish just before "Each" in the top line connecting the previous word. This has roughly the same intensity as some of the character connections in the next line.

techniques and our algorithm is one approach to using gray-level segmentation techniques.

The results show that gap-metrics algorithms work well for good quality pages while scalespace outperforms gap-metrics on noisy historical documents.

10 Conclusion

We have presented a novel and efficient technique for segmenting words out of historical documents. We have shown that this algorithm outperforms a state of the art gap-metrics algorithm when the scanned page images are noisy. The algorithm takes advantage of the gray-level information in images and does not require any binarization—unlike gap-metrics. Binarization is difficult on historic documents which are noisy and have bleed-through, faded ink, or other artifacts.

The approach takes advantage of scalespace ideas to smooth images to produce blobs which correspond to character-like entities at small scales and word-like entities at larger scales. Three different techniques were suggested for scale selection and their performance compared. It was shown that removing margins helped improve performance, although there was little difference between the three different techniques used.

A number of different data sets were used to test the scalespace approach. These included a 100 page set from the George Washington data set and a 10 page data set from Joseph Grinnell's field notes. In addition, modern multi-writer data sets from the IAM database were all used. It was shown that, while, on one IAM data set, the gap-metrics algorithm works slightly better than the scalespace algorithm and, on another IAM data set, their performance is comparable, on the noisy historical George Washington data set, the scalespace algorithm works much better. The scalespace approach to word segmentation has been used to build a system for retrieving 1,000 pages of George Washington's pages and we believe this approach will allow for other historical documents to be made easily accessible. Since no assumptions are made about the nature of handwriting or fonts, this approach may be extendable to documents in other languages.

Acknowledgments

This work is supported in part by the Center for Intelligent Information Retrieval and in part by the US National Science Foundation under grant number IIS-9909073. Any opinions, findings, and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors. The authors thank the Library of Congress for providing scans of Washington's manuscripts and the Museum of Vertebrate Zoology for the scans of Grinnell's writings. They thank Nitin Srimal for his work on an early version of this algorithm and Toni Rath for his comments.

References

[1] H.S. Baird and K. Thompson, "Reading Chess," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 12, pp. 552-559, 1990.

[2] R.M. Bozinovic and S.N. Srihari, "Off-Line Cursive Script Word Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 11, no. 1, pp. 68-83, Jan. 1989.

[3] R.G. Casey and E. Lecolinet, "A Survey of Methods and Strategies in Character Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 7, pp. 690-706, July 1996.

[4] R.O. Duda and P.E. Hart, "Experiments in Recognition of Hand-Printed Text," *Am. Federation of Information Processing Soc. Conf. Proc.,* pp. 1139-1149, 1968.

[5] M. Feldbach and K.D. Tonnies, "Segmentation of the Date in Entries of Historical Church Registers," *Proc. Deutsche Arbeitsgemeinschaft für Mustererkennung Symp.,* pp. 403-410, 2002.

[6] L.M.J. Florack, *The Syntactic Structure of Scalar Images.* Kluwer Academic, 1997.

[7] V. Govindaraju and H. Xue, "Fast Handwriting Recognition for Indexing Historical Documents," *Proc. Workshop Document Image Analysis for Libraries,* pp. 314-320, 2004.

[8] J. Ha, R.M. Haralick, and I.T. Phillips, "Document Page Decomposition by the Bounding-Box Projection Technique," *Proc. Int'l Conf. Document Analysis and Recognition,* pp. 1119-1122, 1995.

[9] P.V.C. Hough, *Method and Means for Recognizing Complex Patterns,* US patent 3069654, Washington, D.C.: Patent and Trademark Office, 1962.

[10] G. Kim, V. Govindaraju, and S.N. Srihari, "An Architecture for Handwritten Text Recognition Systems," *Int'l J. Document Analysis and Recognition,* pp. 37-44, 1999.

[11] J.J. Koenderink and A.J. van Doorn, "Surface Shapes and Curvatures Scales," *Image and Vision Computing,* vol. 10, no. 8, 1992.

[12] V. Lavrenko, T.M. Rath, and R. Manmatha, "Holistic Word Recognition for Handwritten Historical Documents," *Proc. Workshop Document Image Analysis for Libraries,* pp. 278-287, 2004.

[13] T. Lindeberg, *Scale-Space Theory in Computer Vision.* Kluwer Academic, 1994.

[14] U. Mahadevan and R.C. Nagabushnam, "Gap Metrics for Word Separation in Handwritten Lines," *Proc. Int'l Conf. Document Analysis and Recognition,* pp. 124-127, 1995.

[15] R. Manmatha and W.B. Croft, "Word Spotting: Indexing Handwritten Manuscripts," *Intelligent Multimedia Information Retrieval,* Mark Maybury, ed., AAAI/MIT Press, April 1998.

[16] R. Manmatha, C. Han, and E.M. Riseman, "Word Spotting: A New Approach to Indexing Handwritting," *Proc. Computer Vision and Pattern Recognition Conf.,* pp. 53-62, 1996.

[17] R. Manmatha and N. Srimal, "Scale Space Technique for Word Segmentation in Handwritten Documents," *Scale-Space Theories in Computer Vision,* pp. 22-33, 1999.

[18] D. Marr and E. Hildreth, "Theory of Edge Detection," *Proc. Royal Soc. London,* vol. 207-B, pp. 187-217, 1980.

[19] U.V. Marti and H. Bunke, "A Full English Sentence Database for Off-Line Handwriting Recognition," *Proc. Fifth Int'l Conf. Document Analysis and Recognition,* pp. 705-708, 1999.

[20] U.V. Marti and H. Bunke, "Text Line Segmentation and Word Recognition in a System for General Writer Independent Handwriting Recognition," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 159-163, 2001.

[21] U.V. Marti and H. Bunke, "Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System," *Int'l J. Pattern Recognition and Artificial Intelligence,* vol. 15, no. 1, pp. 65-90, 2001.
[22] G. Nagy, "Twenty Years of Document Image Analysis in PAMI," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 1, pp. 38-62, Jan. 2000.
[23] N. Otsu, "A Threshold Selection Method from Gray-Level Histogram," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 9, no. 1, pp. 62-66, Jan. 1979.
[24] R. Plamondon and S.N. Srihari, "Online and Offline Handwriting Recognition: A Comprehensive Survey," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 1, pp. 63-84, Jan. 2000.
[25] T.M. Rath, V. Lavrenko, and R. Manmatha, "A Search Engine for Historical Manuscript Images," *Proc. ACM Int'l Conf. Research and Development in Information Retrieval,* pp. 369-376, 2004.
[26] G. Seni and E. Cohen, "External Word Segmentation of Off-Line Handwritten Text Lines," *Pattern Recognition,* vol. 27, pp. 41-52, 1994.
[27] A.W. Senior and A.J. Robinson, "An Off-Line Cursive Handwriting Recognition System," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 20, pp. 309-321, 1998.
[28] C. Tomai, B. Zhang, and V. Govindaraju, "Transcript Mapping for Historic Handwritten Document Images," *Proc. Eighth Int'l Workshop Frontiers in Handwriting Recognition,* pp. 413-418, 2002.
[29] O. Trier, A. Jain, and T. Taxt, "Feature Extraction Methods for Character Recognition—A Survey," *Pattern Recognition,* vol. 29, pp. 641-662, 1996.
[30] A. Vinciarelli and J. Luettin, "Off-Line Cursive Script Recognition Based on Continuous Density HMM," *Proc. Seventh Int'l Workshop Frontiers in Handwriting Recognition,* pp. 493-498, Sept. 2000.
[31] J.A. Weickert, S. Ishikawa, and A. Imiya, "On the History of Gaussian Scale-Space Axiomatics," *Gaussian Scale-Space Theory,* J. Sporring, M. Nielsen, L.M.J. Florack, and P. Johansen, eds., pp. 45-59. Kluwer Academic, 1997.

R. Manmatha received the PhD degree from the University of Massachusetts, Amherst, and the BTech degree from the Indian Institute of Technology, Kanpur, India. He is a research assistant professor in the Computer Science Department at the University of Massachusetts. He leads the multimedia indexing and retrieval group at the Center for Intelligent Information Retrieval. His current research interests are in applying statistical models to the recognition and retrieval of historical handwritten manuscripts and to image and video retrieval. He has served on a number of program committees of conferences. He was an associate editor of the *ACM Transactions on Information Systems* (2000-2002) and is currently an associate editor of *Pattern Recognition Letters*. He is a member of the IEEE Computer Society.

Jamie L. Rothfeder received the BS degree in computer science from the University of Massachusetts and is currently working toward the MS degree in computer science. He is part of the multimedia indexing and retrieval group at the Center of Intelligent Information Retrieval and a graduate research assistant in the Computer Science Department at the University of Massachusetts. He is interested in combining techniques from machine learning, computer vision, and information retrieval to perform object recognition and retrieval.