Text Segmentation for MRC Document Compression

Eri Haneda, *Student Member, IEEE,* and Charles A. Bouman, *Fellow, IEEE*

Abstract

The Mixed Raster Content (MRC) standard (ITU-T T.44) specifies a framework for document compression which can dramatically improve the compression/quality tradeoff as compared to traditional lossy image compression algorithms. The key to MRC compression is the separation of the document into foreground and background layers, represented as a binary mask. Therefore, the resulting quality and compression ratio of a MRC document encoder is highly dependent on the segmentation algorithm used to compute the binary mask.

In this paper, we propose a novel multiscale segmentation scheme for MRC document encoding based on the sequential application of two algorithms. The first algorithm, cost optimized segmentation (COS), is a blockwise segmentation algorithm formulated in a global cost optimization framework. The second algorithm, connected component classification (CCC), refines the initial segmentation by classifying feature vectors of connected components using an Markov random field (MRF) model. The combined COS/CCC segmentation algorithms are then incorporated into a multiscale framework in order to improve the segmentation accuracy of text with varying size. In comparisons to state-of-the-art commercial MRC products and selected segmentation algorithms in the literature, we show that the new algorithm achieves greater accuracy of text detection but with a lower false detection rate of non-text features. We also demonstrate that the proposed segmentation algorithm can improve the quality of decoded documents while simultaneously lowering the bit rate.

EDICS Category: ELI-DOC

E. Haneda and C.A. Bouman are with the School of Electrical and Computer Engineering, Purdue University, 465 Northwestern Ave., West Lafayette, IN 47907-2035, USA (E-mail: {ehaneda,bouman}@ecn.purdue.edu)

This work was supported by Samsung Electronics.

Text Segmentation for MRC Document Compression

I. Introduction

With the wide use of networked equipment such as computers, scanners, printers and copiers, it has become more important to efficiently compress, store, and transfer large document files. For example, a typical color document scanned at 300 dpi requires approximately 24M bytes of storage without compression. While JPEG and JPEG2000 are frequently used tools for natural image compression, they are not very effective for the compression of raster scanned compound documents which typically contain a combination of text, graphics, and natural images. This is because the use of a fixed DCT or wavelet transformation for all content typically results in severe ringing distortion near edges and line-art.

The mixed raster content (MRC) standard is a framework for layer-based document compression defined in the ITU-T T.44 [1] that enables the preservation of text detail while reducing the bitrate of encoded raster documents. The most basic MRC approach, MRC mode 1, divides an image into three layers: a binary mask layer, foreground layer, and background layer. The binary mask indicates the assignment of each pixel to the foreground layer or the background layer by a “1” or “0” value, respectively. Typically, text regions are classified as foreground while picture regions are classified as background. Each layer is then encoded independently using an appropriate encoder. For example, foreground and background layers may be encoded using traditional photographic compression such as JPEG or JPEG2000 while the binary mask layer may be encoded using symbol-matching based compression such as JBIG or JBIG2. Moreover, it is often the case that different compression ratios and subsampling rates are used for foreground and background layers due to their different characteristics. Typically, the foreground layer is more aggressively compressed than the background layer because the foreground layer requires lower color and spatial resolution. Figure 1 shows an example of layers in an MRC mode 1 document.

Perhaps the most critical step in MRC encoding is the segmentation step, which creates a binary mask that separates text and line-graphics from natural image and background regions in the document. Segmentation influences both the quality and bitrate of an MRC document. For example, if a text component is not properly detected by the binary mask layer, the text edges will be blurred by the background layer encoder. Alternatively, if non-text is erroneously detected as text, this error can also cause distortion through the introduction of false edge artifacts and the excessive smoothing of regions

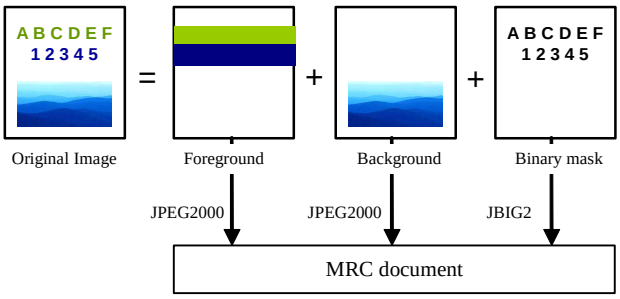


Fig. 1. Illustration of Mixed Raster Content (MRC) document compression standard mode 1 structure. An image is divided into three layers: a binary mask layer, foreground layer, and background layer. The binary mask indicates the assignment of each pixel to the foreground layer or the background layer by a "1" (black) or "0" (white), respectively. Typically, text regions are classified as foreground while picture regions are classified as background. Each layer is then encoded independently using an appropriate encoder.

assigned to the foreground layer. Furthermore, erroneously detected text can also increase the bit rate required for symbol-based compression methods such as JBIG2. This is because erroneously detected and unstructure non-text symbols are not be efficiently represented by JBIG2 symbol dictionaries.

Many segmentation algorithms have been proposed for accurate text extraction, typically with the application of optical character recognition (OCR) in mind. One of the most popular top-down approaches to document segmentation is the X-Y cut algorithm [2] which works by detecting white space using horizontal and vertical projections. The run length smearing algorithm (RLSA) [3], [4] is a bottom-up approach which basically uses region growing of characters to detect text regions, and the Docstrum algorithm proposed in [5] is another bottom-up method which uses k-nearest neighbor clustering of connected components. Chen et. al recently developed a multi-plane based segmentation method by incorporating a thresholding method [6]. A summary of the algorithms for document segmentation can be found in [7], [8], [9], [10], [11].

Perhaps the most traditional approach to text segmentation is Otsu's method [12] which thresholds pixels in an effort to divide the document's histogram into objects and background. There are many modified versions of Otsu's method [13], [6]. While Otsu uses a global thresholding approach, Niblack [14] and Sauvola [15] use a local thresholding approach. Kapur's method [16] uses entropy information for the global thresholding, and Tsai [17] uses a moment preserving approach. A comparison of the algorithms for text segmentation can be found in [18].

In order to improve text extraction accuracy, some text segmentation approaches also use character

properties such as size, stroke width, directions, and run-length histogram [19], [20], [21], [22]. Other binarization approaches for document coding have used rate-distortion minimization as a criteria for document binarization [23], [24].

Many recent approaches to text segmentation have been based on statistical models. One of the best commercial text segmentation algorithms, which is incorporated in the DjVu document encoder, uses a hidden Markov model (HMM) [25], [26]. The DjVu software package is perhaps the most popular MRC-based commercial document encoder. Although there are other MRC-based encoders such as LuraDocument [27], we have found DjVu to be the most accurate and robust algorithm available for document compression. However, as a commercial package, the full details of the DjVu algorithm are not available. Zheng *et al.* [28] used an MRF model to exploit the contextual document information for noise removal. Similarly, Kumar [29] *et al.* used an MRF model to refine the initial segmentation generated by the wavelet analysis. J. G. Kuk *et al.* and Cao *et al.* also developed a MAP-MRF text segmentation framework which incorporates their proposed prior model [30], [31].

Recently, a conditional random field (CRF) model, originally proposed by Lafferty [32], has attracted interest as an improved model for segmentation. The CRF model differs from the traditional MRF models in that it directly models the posterior distribution of labels given observations. For this reason, in the CRF approach the interactions between labels are a function of both labels and observations. The CRF model has been applied to different types of labeling problems including blockwise segmentaion of manmade structures [33], natural image segmentation [34], and pixelwise text segmentation [35].

In this paper, we present a robust multiscale segmentation algorithm for both detecting and segmenting text in complex documents containing background gradations, varying text size, reversed contrast text, and noisy backgrounds. While considerable research has been done in the area of text segmentation, our approach differs in that it integrates a stochastic model of text structure and context into a multiscale framework in order to best meet the requirements of MRC document compression. Accordingly, our method is designed to minimize false detections of unstructured non-text components (which can create artifacts and increase bit-rate) while accurately segmenting true-text components of varying size and with varying backgrounds. Using this approach, our algorithm can achieve higher decoded image quality at a lower bit-rate than generic algorithms for document segmentation. We note that a preliminary version of this approach, without the use of an MRF prior model, was presented in the conference paper of [36], and that the source code for the method described in this paper is publicly available. [1]

[1]https://engineering.purdue.edu/~bouman

Our segmentation method is composed of two algorithms that are applied in sequence: the cost optimized segmentation (COS) algorithm and the connected component classification (CCC) algorithm. The COS algorithm is a blockwise segmentation algorithm based on cost optimization. The COS produces a binary image from a gray level or color document; however, the resulting binary image typically contains many false text detections. The CCC algorithm further processes the resulting binary image to improve the accuracy of the segmentation. It does this by detecting non-text components (i.e. false text detections) in a Bayesian framework which incorporates an Markov random field (MRF) model of the component labels. One important innovation of our method is in the design of the MRF prior model used in the CCC detection of text components. In particular, we design the energy terms in the MRF distribution so that they adapt to attributes of the neighboring components' relative locations and appearance. By doing this, the MRF can enforce stronger dependencies between components which are more likely to have come from related portions of the document.

The organization of this paper is as follows. In Section II and Section III, we describe COS and CCC algorithms. We also describe the multiscale implementation in Section IV. Section V presents experimental results, in both quantitative and qualitative ways.

II. Cost Optimized Segmentation (COS)

The Cost Optimized Segmentation (COS) algorithm is a block-based segmentation algorithm formulated as a global cost optimization problem. The COS algorithm is comprised of two components: blockwise segmentation and global segmentation. The blockwise segmentation divides the input image into overlapping blocks and produces an initial segmentation for each block. The global segmentation is then computed from the initial segmented blocks so as to minimize a global cost function, which is carefully designed to favor segmentations that capture text components. The parameters of the cost function are optimized in an off-line training procedure. A block diagram for COS is shown in Fig. 2.

A. Blockwise Segmentation

Blockwise segmentation is performed by first dividing the image into overlapping blocks, where each block contains $m \times m$ pixels, and adjacent blocks overlap by $m/2$ pixels in both the horizontal and vertical directions. The blocks are denoted by $O_{i,j}$ for $i = 1, .., M$, and $j = 1, .., N$, where M and N are the number of the blocks in the vertical and horizontal directions, respectively. If the height and width of the input image is not divisible by m, the image is padded with zeros. For each block, the color axis

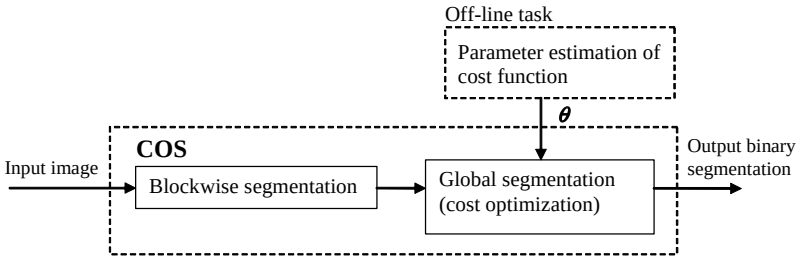


Fig. 2. The COS algorithm comprises two steps: blockwise segmentation and global segmentation. The parameters of the cost function used in the global segmentation are optimized in an off-line training procedure.

having the largest variance over the block is selected and stored in a corresponding gray image block, $\tilde{O}_{i,j}$.

The pixels in each block are segmented into foreground ("1") or background ("0") by the clustering method of Cheng and Bouman [24]. The clustering method classifies each pixel in $\tilde{O}_{i,j}$ by comparing it to a threshold t. This threshold is selected to minimize the total sub-class variance. More specifically, the minimum value of the total sub-class variance is given by

$$\gamma_{i,j}^2 = \min_{t\in[0,255]} \frac{N_{0,i,j} * \sigma_{0,i,j}^2 + N_{1,i,j} * \sigma_{1,i,j}^2}{N_{0,i,j} + N_{1,i,j}} \tag{1}$$

where $N_{0,i,j}$ and $N_{1,i,j}$ are number of pixels classified as 0 and 1 in $\tilde{O}_{i,j}$ by the threshold t, and $\sigma_{0,i,j}^2$ and $\sigma_{1,i,j}^2$ are the variances within each sub-class (See Fig. 3). Note that the sub-class variance can be calculated efficiently. First, we create a histogram by counting the number of pixels which fall into each value between 0 and 255. For each threshold $t \in [0, 255]$, we can recursively calculate $\sigma_{0,i,j}^2$ and $\sigma_{1,i,j}^2$ from the values calculated for the previous threshold of $t-1$. The threshold that minimizes the sub-class variance is then used to produce a binary segmentation of the block denoted by $C_{i,j} \in \{0,1\}^{m\times m}$.

B. Global Segmentation

The global segmentation step integrates the individual segmentations of each block into a single consistent segmentation of the page. To do this, we allow each block to be modified using a class

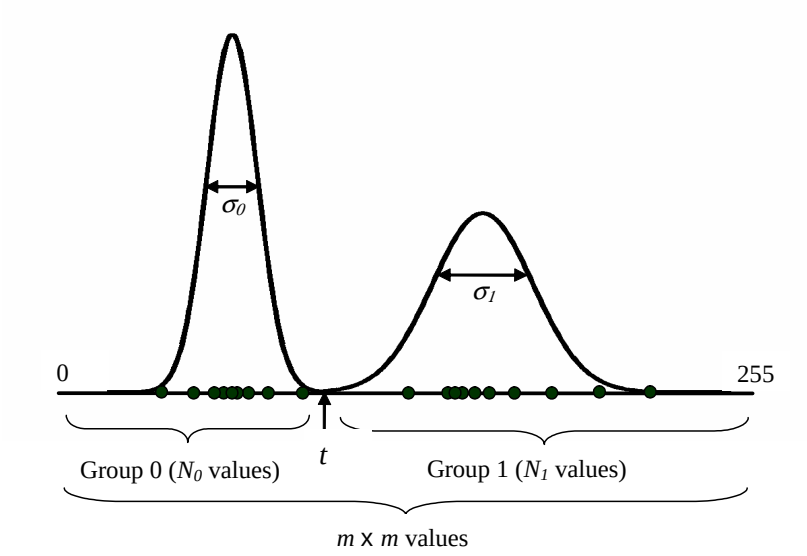


Fig. 3. Illustration of a blockwise segmentation. The pixels in each block are separated into foreground ("1") or background ("0") by comparing each pixel with a threshold t. The threshold t is then selected to minimize the total sub-class variance.

assignment denoted by, $s_{i,j} \in \{0, 1, 2, 3\}$.

$$\begin{aligned} s_{i,j} = 0 \quad &\Rightarrow \quad \tilde{C}_{i,j} = C_{i,j} \qquad \text{(Original)} \\ s_{i,j} = 1 \quad &\Rightarrow \quad \tilde{C}_{i,j} = \neg C_{i,j} \qquad \text{(Reversed)} \\ s_{i,j} = 2 \quad &\Rightarrow \quad \tilde{C}_{i,j} = \{0\}^{m \times m} \quad \text{(All background)} \\ s_{i,j} = 3 \quad &\Rightarrow \quad \tilde{C}_{i,j} = \{1\}^{m \times m} \quad \text{(All foreground)} \end{aligned} \tag{2}$$

Notice that for each block, the four possible values of $s_{i,j}$ correspond to four possible changes in the block's segmentation: original, reversed, all background, or all foreground. If the block class is "original", then the original binary segmentation of the block is retained. If the block class is "reversed", then the assignment of each pixel in the block is reversed (i.e. 1 goes to 0, or 0 goes to 1). If the block class is set to "all background" or "all foreground", then the pixels in the block are set to all 0's or all 1's, respectively. Figure 4 illustrates an example of the four possible classes where black indicates a label of "1" (foreground) and white indicates a label of "0" (background).

Our objective is then to select the class assignments, $s_{i,j} \in \{0, 1, 2, 3\}$, so that the resulting binary masks, $\tilde{C}_{i,j}$, are consistent. We do this by minimizing the following global cost as a function of the class

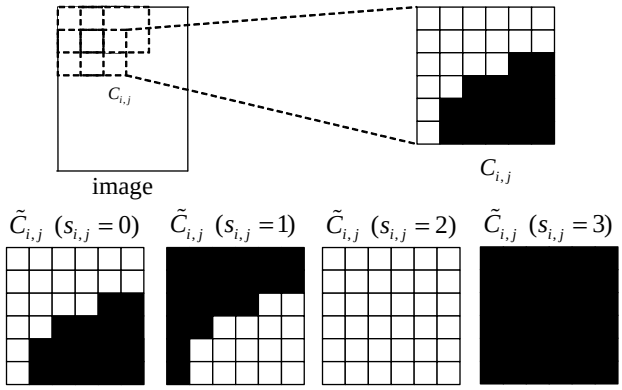


Fig. 4. Illustration of class definition for each block. Black indicates a label of "1" (foreground) and white indicates a label of "0" (background). Four segmentation result candidates are defined; original (class 0), reversed (class 1), all background (class 2), and all foreground (class 3). The final segmentation will be one of these candidates. In this example, block size is $m = 6$.

assignments, $S = [s_{i,j}]$ for all i, j,

$$f_1(S) = \sum_{i=1}^{M} \sum_{j=1}^{N} \{\mathcal{E}(s_{i,j}) + \lambda_1 V_1(s_{i,j}, s_{i,j+1}) + \lambda_2 V_2(s_{i,j}, s_{i+1,j}) + \lambda_3 V_3(s_{i,j})\}. \tag{3}$$

As it is shown, the cost function contains four terms, the first term representing the fit of the segmentation to the image pixels, and the next three terms representing regularizing constraints on the segmentation. The values λ_1, λ_2, and λ_3 are then model parameters which can be adjusted to achieve the best segmentation quality.

The first term $\mathcal{E}$ is the square root of the total sub-class variation within a block given the assumed segmentation. More specifically,

$$\mathcal{E}(s_{i,j}) = \begin{cases} \gamma_{i,j} & \text{if } s_{i,j} = 0 \text{ or } s_{i,j} = 1 \\ \sigma_{i,j} & \text{if } s_{i,j} = 2 \text{ or } s_{i,j} = 3 \end{cases} \tag{4}$$

where $\sigma_{i,j}$ is the standard deviation of all the pixels in the block. Since $\gamma_{i,j}$ must always be less than or equal to $\sigma_{i,j}$, the term $\mathcal{E}$ can always be reduced by choosing a finer segmentation corresponding to $s_{i,j} = 0$ or 1 rather than smoother segmentation corresponding to $s_{i,j} = 2$ or 3.

The terms V_1 and V_2 regularize the segmentation by penalizing excessive spatial variation in the segmentation. To compute the term V_1, the number of segmentation mismatches between pixels in the overlapping region between block $\tilde{C}_{i,j}$ and the horizontally adjacent block $\tilde{C}_{i,j+1}$ is counted. The term V_1 is then calculated as the number of the segmentation mismatches divided by the total number of pixels in the overlapping region. Also V_2 is similarly defined for vertical mismatches. By minimizing these terms,

the segmentation of each block is made consistent with neighboring blocks.

The term V_3 denotes the number of the pixels classified as foreground (i.e. "1") in $\tilde{C}_{i,j}$ divided by the total number of pixels in the block. This cost is used to ensure that most of the area of image is classified as background.

For computational tractability, the cost minimization is iteratively performed on individual rows of blocks, using a dynamic programming approach [37]. Note that row-wise approach does not generally minimize the global cost function in one pass through the image. Therefore, multiple iterations are performed from top to bottom in order to adequately incorporate the vertical consistency term. In the first iteration, the optimization of i^{th} row incorporates the V_2 term containing only the $i-1^{th}$ row. Starting from the second iteration, V_2 terms for both the $i-1^{th}$ row and $i+1^{th}$ row are included. The optimization stops when no changes occur to any of the block classes. Experimentally, the sequence of updates typically converges within 20 iterations.

The cost optimization produces a set of classes for overlapping blocks. Since the output segmentation for each pixel is ambiguous due to the block overlap, the final COS segmentation output is specified by the $\frac{m}{2} \times \frac{m}{2}$ center region of each $m \times m$ overlapping block.

The weighting coefficients λ_1,λ_2, and λ_3 were found by minimizing the weighted error between segmentation results of training images and corresponding ground truth segmentations. A ground truth segmentation was generated manually by creating a mask that indicates the text in the image. The weighted error criteria which we minimized is given by

$$\varepsilon_{weighted} = (1-\omega)N_{missed\_detection} + \omega N_{false\_detection} \tag{5}$$

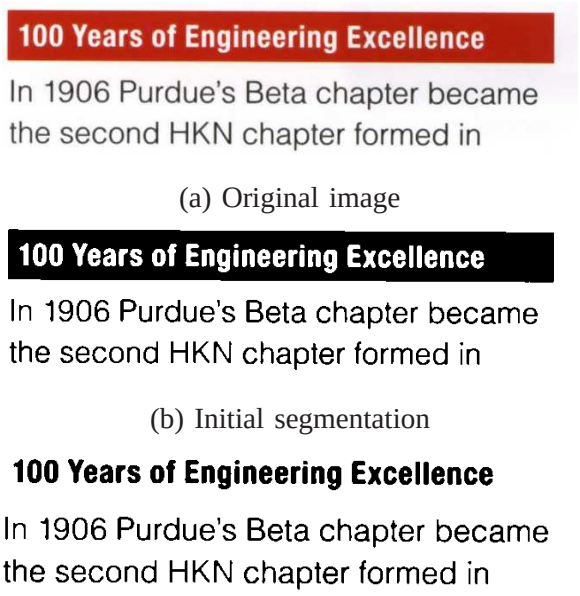
where $\omega \in [0,1]$, and the terms $N_{missed\_detection}$ and $N_{false\_detection}$ are the number of pixels in the missed detection and false detection categories, respectively. For our application, the missed detections are generally more serious than false detections, so we used a value of $\omega = 0.09$ which more heavily weighted miss detections. In the next section, we will show how the resulting false detections can be effectively reduced.

III. Connected Component Classification (CCC)

The connected component classification (CCC) algorithm refines the segmentation produced by COS by removing many of the erroneously detected non-text components. The CCC algorithm proceeds in three steps: connected component extraction, component inversion, and component classification. The connected component extraction step identifies all connected components in the COS binary segmentation using a

4-point neighborhood. In this case, connected components less than six pixels were ignored because they are nearly invisible at 300 dpi resolution. The component inversion step corrects text segmentation errors that sometimes occur in COS segmentation when text is locally embedded in a highlighted region (See Fig. 5 (a)). Figure 5 (b) illustrates this type of error where text is initially segmented as background. Notice the text "100 Years of Engineering Excellence" is initially segmented as background due to the red surrounding region. In order to correct these errors, we first detect foreground components that contain more than eight interior background components (holes). In each case, if the total number of interior background pixels is less than half of the surrounding foreground pixels, the foreground and background assignments are inverted. Figure 5 (c) shows the result of this inversion process. Note that this type of error is a rare occurrence in the COS segmentation.

The final step of component classification is performed by extracting a feature vector for each component, and then computing a MAP estimate of the component label. The feature vector, y_i, is calculated for each connected component, CC_i, in the COS segmentation. Each y_i is a 4 dimensional feature vector which describes aspects of the i^{th} connected component including edge depth and color uniformity. Finally, the feature vector y_i is used to determine the class label, x_i, which takes a value of 0 for non-text and 1 for text.



(a) Original image

(b) Initial segmentation

(c) Preprocessed segmentation

Fig. 5. Illustration of how the component inversion step can correct erroneous segmentations of text. (a) Original document before segmentation, (b) Result of COS binary segmentation, (c) Corrected segmentation after component inversion.

The Bayesian segmentation model used for the CCC algorithm is shown in Fig. 6. The conditional distribution of the feature vector y_i given x_i is modeled by a multivariate Gaussian mixture while the underlying true segmentation labels are modeled by a Markov random field (MRF). Using this model, we classify each component by calculating the MAP estimate of the labels, x_i, given the feature vectors, y_i. In order to do this, we first determine which components are neighbors in the MRF. This is done based on the geometric distance between components on the page.

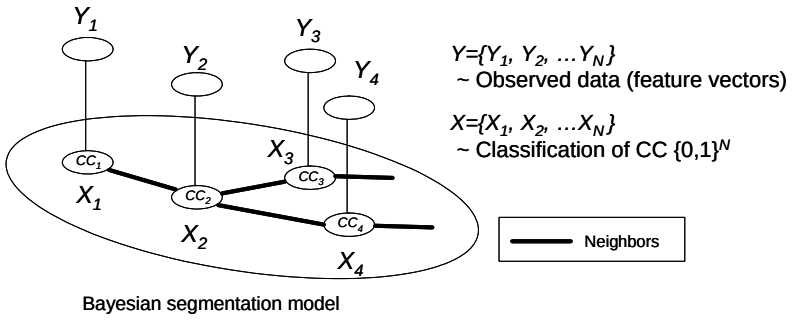


Fig. 6. Illustration of a Bayesian segmentation model. Line segments indicate dependency between random variables. Each component CC_i has an observed feature vector, y_i, and a class label, $x_i \in \{0, 1\}$. Neighboring pairs are indicated by thick line segments.

A. Statistical model

Here, we describe more details of the statistical model used for the CCC algorithm. The feature vectors for "text" and "non-text" groups are modeled as D-dimensional multivariate Gaussian mixture distributions,

$$p(y_i|x_i) = \sum_{m=0}^{M_{x_i}-1} \frac{a_{x_i,m}}{(2\pi)^{D/2}|R_{x_i,m}|^{-1/2}} \exp\left\{-\frac{1}{2}(y_i - \mu_{x_i,m})^t R_{x_i,m}^{-1}(y_i - \mu_{x_i,m})\right\}, \tag{6}$$

where $x_i \in \{0, 1\}$ is a class label of non-text or text for the i^{th} connected component. The M_0 and M_1 are the number of clusters in each Gaussian mixture distribution, and the $\mu_{x_i,m}$, $R_{x_i,m}$, and $a_{x_i,m}$ are the mean, covariance matrix, and weighting coefficient of the m^{th} cluster in each distribution. In order to simplify the data model, we also assume that the values Y_i are conditionally independent given the associated values X_i.

$$p(y|x) = \prod_{i=1}^{N} p(y_i|x_i) \tag{7}$$

The components of the feature vectors y_i include measurements of edge depth and external color uniformity of the i^{th} connected component. The edge depth is defined as the Euclidean distance between RGB values of neighboring pixels across the component boundary (defined in the initial COS segmentation). The color uniformity is associated with the variation of the pixels outside the boundary. In this experiment, we defined a feature vector with four components, $y_i = [y_{1i}\ y_{2i}\ y_{3i}\ y_{4i}]^T$, where the first two are mean and variance of the edge depth and the last two are the variance and range of external pixel values. More details are provided in the Appendix.

To use an Markov random field model (MRF), we must define a neighborhood system. To do this, we first find the pixel location at the center of mass for each connected component. Then for each component we search outward in a spiral pattern until the k nearest neighbors are found. The number k is determined in an off-line training process along with other model parameters. We will use the symbol ∂s to denote the set of neighbors of connected component s. To ensure all neighbors are mutual (which is required for an MRF), if component s is a neighbor of component r (i.e. $s \in \partial r$), we add component r to the neighbor list of component s (i.e. $r \in \partial s$) if this is not already the case.

In order to specify the distribution of the MRF, we first define augmented feature vectors. The augmented feature vector, z_i, for the i^{th} connected component consists of the feature vector y_i concatenated with the horizontal and vertical pixel location of the connected component's center. We found the location of connected components to be extremely valuable contextual information for text detection. For more details of the augmented feature vector, see Appendix.

Next, we define a measure of dissimilarity between connected components in terms of the Mahalanobis distance of the augmented feature vectors given by

$$d_{i,j} = \sqrt{(z_i - z_j)^T \Sigma^{-1} (z_i - z_j)} \tag{8}$$

where Σ is the covariance matrix of the augmented feature vectors on training data. Next, the Mahalanobis distance, $d_{i,j}$, is normalized using the equations,

$$\begin{aligned} D_{i,j} &= \frac{d_{i,j}}{\frac{1}{2}(\bar{d}_{i,\partial i} + \bar{d}_{j,\partial j})} \\ \bar{d}_{i,\partial i} &= \frac{1}{|\partial i|} \sum_{k \in \partial i} d_{i,k} \\ \bar{d}_{j,\partial j} &= \frac{1}{|\partial j|} \sum_{k \in \partial j} d_{j,k} \end{aligned} \tag{9}$$

where $\bar{d}_{i,\partial i}$ is averaged distance between the i^{th} connected component and all of its neighbors, and $\bar{d}_{j,\partial j}$ is similarly defined. This normalized distance satisfies the symmetry property, that is $D_{i,j} = D_{j,i}$.

Using the defined neighborhood system, we adopted an MRF model with pair-wise cliques. Let P be the set of all (i, j) where i and j denote neighboring connected components. Then, the X_i are assumed to be distributed as

$$p(x) = \frac{1}{Z} \exp\left\{ - \sum_{\{i,j\} \in P} w_{i,j} \delta(x_i \neq x_j) \right\} \tag{10}$$

$$w_{i,j} = \frac{b}{D_{i,j}^p + a} \tag{11}$$

where $\delta(\cdot)$ is an indicator function taking the value 0 or 1, and a, b, and p are scalar parameters of the MRF model. As we can see, the classification probability is penalized by the number of neighboring pairs which have different classes. This number is also weighted by the term $w_{i,j}$. If there exists a similar neighbor close to a given component, the term $w_{i,j}$ becomes large since $D_{i,j}$ is small. This favors increasing the probability that the two similar neighbors have the same class.

Figure 7 illustrates the classification probability of x_i given a single neighbor x_j as a function of the distance between the two components $D_{i,j}$. Here, we assume the classification of x_j is given. The solid line shows a graph of the probability $p(x_i \neq x_j | x_j)$ while the dashed line shows a graph of the probability $p(x_i = x_j | x_j)$. Note that the parameter p controls the roll-off of the function, and a and b control the minimum and transition point for the function. The actual parameters, $\phi = [p,\ a,\ b]^T$, are optimized in an off-line training procedure (See sec. III-B).

With the MRF model defined above, we can compute a maximum a posteriori (MAP) estimate to find the optimal set of classification labels $x = [x_1\ x_2\ \ldots\ x_N]^T$. The MAP estimate is given by,

$$\hat{x}_{MAP} = \underset{x \in \{0,1\}^N}{\operatorname{argmin}} \left\{ - \sum_{i \in S} \log p(y_i | x_i) + \sum_{\{i,j\} \in P} w_{i,j} \delta(x_i \neq x_j) - c_{text} \delta(x_i = 1) \right\}. \tag{12}$$

We introduced a term c_{text} to control the trade-off between missed and false detections. The term c_{text} may take a positive or negative value. If c_{text} is positive, both text detections and false detections increase. If it is negative, false and missed detections are reduced.

To find an approximate solution of (12), we use iterative conditional modes (ICM) which sequentially minimizes the local posterior probabilities [38], [39]. The classification labels, $\{x_i |\ i \in S\}$, are initialized with their maximum likelihood (ML) estimates, and then the ICM procedure iterates through the set of classification labels until a stable solution is reached.

B. Parameter estimation

In our statistical model, there are two sets of parameters to be estimated by a training process. The first set of parameters comes from the D-dimensional multivariate Gaussian mixture distributions given

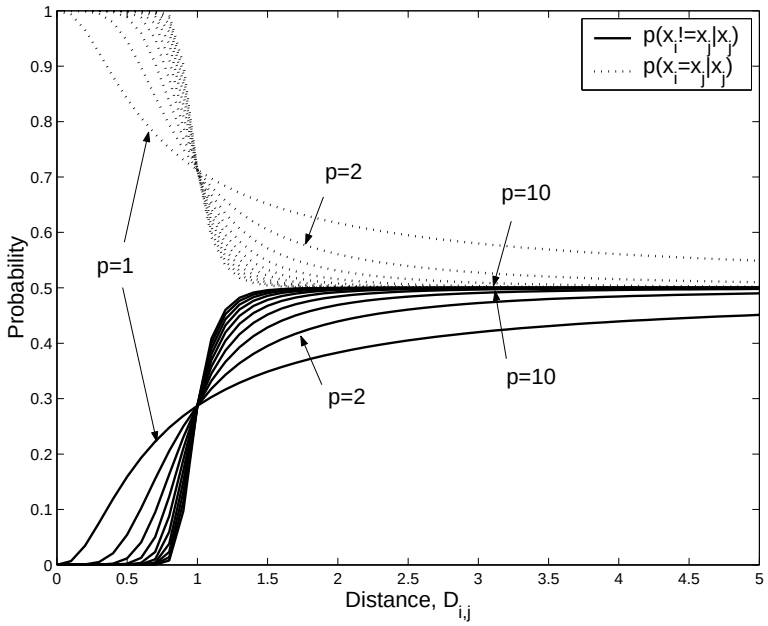


Fig. 7. Illustration of classification probability of x_i given a single neighbor x_j as a function of the distance between the two components. The solid line shows a graph of $p(x_i \neq x_j|x_j)$ while the dashed line shows a graph of $p(x_i = x_j|x_j)$. The parameters are set to $a = 0.1$ and $b = 1.0$.

by Eq. (6) which models the feature vectors from text and non-text classes. The second set of parameters, $\phi = [p,\ a,\ b]^T$, controls the MRF model of Eq. (11). All parameters are estimated in an off-line procedure using a training image set and their ground truth segmentations. The training images were first segmented using the COS algorithm. All foreground connected components were extracted from the resulting segmentations, and then each connected component was labeled as text or non-text by matching to the components on the ground truth segmentations. The corresponding feature vectors were also calculated from the original training images.

The parameters of the Gaussian mixture distribution were estimated using the EM algorithm [40]. The number of clusters in each Gaussian mixture for text and non-text were also determined using the minimum description length (MDL) estimator [41].

The prior model parameters $\phi = [p,\ a,\ b]^T$ were independently estimated using pseudolikelihood maximization [42], [43], [44]. In order to apply pseudolikelihood estimation, we must first calculate the conditional probability of a classification label given its neighboring components' labels,

$$p(x_i|x_{\partial i}) = \frac{1}{Z_i} \exp\left\{ - \sum_{j \in \partial i} w_{i,j} \delta(x_i \neq x_j) \right\} \tag{13}$$

where the normalization factor Z_i is defined by

$$Z_i = \sum_{x_i \in \{0,1\}} \exp\left\{ - \sum_{j \in \partial i} w_{i,j} \delta(x_i \neq x_j) \right\}. \tag{14}$$

Therefore, the pseudolikelihood parameter estimation for our case is,

$$\begin{aligned} \hat{\phi} &= \underset{\phi}{\operatorname{argmax}} \prod_{i \in S} p(x_i | x_{\partial i}) \\ &= \underset{\phi}{\operatorname{argmax}} \prod_{i \in S} \frac{1}{Z_i} \exp\left\{ - \sum_{j \in \partial i} w_{i,j} \delta(x_i \neq x_j) \right\} \\ &= \underset{\phi}{\operatorname{argmin}} \sum_{i \in S} \left\{ \log Z_i + \sum_{j \in \partial i} w_{i,j} \delta(x_i \neq x_j) \right\}. \end{aligned} \tag{15}$$

IV. Multiscale-COS/CCC segmentation scheme

In order to improve accuracy in the detection of text with varying size, we incorporated a multiscale framework into the COS/CCC segmentation algorithm. The multiscale framework allows us to detect both large and small components by combining results from different resolutions [45], [46], [47]. Since the COS algorithm uses a single block size (i.e. single scale), we found that large blocks are typically better suited for detecting large text, and small blocks are better suited for small text. In order to improve the detection of both large and small text, we use a multiscale segmentation scheme which uses the results of coarse-scale segmentations to guide segmentation on finer scales. Note that both COS and CCC segmentations are performed on each scale, however, only COS is adapted to the multiscale scheme.

Figure 8 shows the overview of our multiscale-COS/CCC scheme. In the multiscale scheme, segmentation progresses from coarse to fine scales, where the coarser scales use larger block sizes, and finer scales use smaller block sizes. Each scale is numbered from $L-1$ to 0, where $L-1$ is the coarsest scale and 0 is the finest scale. The COS algorithm is modified to use different block sizes for each scale (denoted as $m^{(n)}$), and incorporates the previous coarser segmentation result by adding a new term to the cost function.

The new cost function for the multiscale scheme is shown in Eq. (16). It is a function of the class assignments on both the n^{th} and $n+1^{th}$ scale.

$$f_2^{(n)}(S^{(n)}) = \sum_{i=1}^{M} \sum_{j=1}^{N} \left\{ f_1^{(n)}(S^{(n)}) + \lambda_4^{(n)} V_4(s_{i,j}^{(n)}, x_{B_{i,j}}^{(n+1)}) \right\} \tag{16}$$

where $(\cdot)^{(n)}$ denotes the term for the n^{th} scale, and $S^{(n)}$ is the set of class assignments for the scale, that is $S = [s_{i,j}^{(n)}]$ for all i, j. The term $B_{i,j}$ is a set of pixels in a block at the position (i, j), and the term $x_{B_{i,j}}$ is the segmentation result for the block.

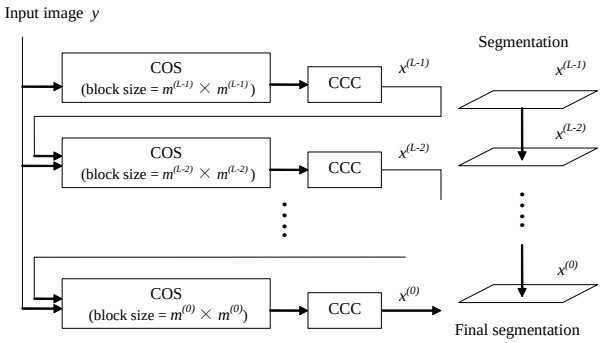


Fig. 8. Illustration of a multiscale-COS/CCC algorithm. Segmentation progresses from coarse to fine scales, incorporating the segmentation result from the previous coarser scale. Both COS and CCC are performed on each scale, however only COS was adapted to the multiscale scheme.

As it is shown in the equation, this modified cost function incorporates a new term V_4 that makes the segmentation consistent with the previous coarser scale. The term V_4, is defined as the number of mismatched pixels within the block $B_{i,j}$ between the current scale segmentation $x^{(n)}_{B_{i,j}}$ and the previous coarser scale segmentation $x^{(n+1)}_{B_{i,j}}$. The exception is that only the pixels that switch from "1" (foreground) to "0" (background) are counted when $s^{(n)}_{i,j} = 0$ or $s^{(n)}_{i,j} = 1$. This term encourages a more detailed segmentation as we proceed to finer scales. The V_4 term is normalized by dividing by the block size on the current scale. Note that the V_4 term is ignored for the coarsest scale. Using the new cost function, we find the class assignments, $s_{i,j} \in \{0, 1, 2, 3\}$, for each scale.

The parameter estimation of the cost function $f_2^{(n)}$ for $n \in \{0, \dots L-1\}$ is performed in an off-line task. The goal is to find the optimal parameter set $\Theta^{(n)} = \{\lambda_1^{(n)} \dots \lambda_4^{(n)}\}$ for $n \in \{0, \dots L-1\}$. To simplify the optimization process, we first performed single scale optimization to find $\Theta'^{(n)} = \{\lambda_1^{(n)} \dots \lambda_3^{(n)}\}$ for each scale. Then, we found the optimal set of $\Theta'' = \{\lambda_4^{(0)} \dots \lambda_4^{(L-2)}\}$ given the $\{\Theta'^{(0)} \dots \Theta'^{(L-1)}\}$. The error to be minimized was the number of mismatched pixels compared to ground truth segmentations, as shown in (5). The ω, the weighting factor of the error for false detection, was fixed to 0.5 for the multiscale-COS/CCC training process.

V. RESULTS

In this section, we compare the multiscale-COS/CCC, COS/CCC, and COS segmentation results with the results of two popular thresholding methods, an MRF-based segmentation method, and two existing commercial software packages which implement MRC document compression. The thresholding methods

used for the comparison in this study are Otsu [12] and Tsai [17] methods. These algorithms showed the best segmentation results among the thresholding methods in [12], [14], [15], [16], and [17]. In the actual comparison, the sRGB color image was first converted to a luma grayscale image, then each thresholding method was applied. For 'Otsu/CCC" and "Tsai/CCC", the CCC algorithm was combined with the Otsu and Tsai binarization algorithms to remove false detections. In this way, we can compare the end result of the COS algorithm to alternative thresholding approaches.

The MRF-based binary segmentation used for the comparison is based on the MRF statistical model developed by Zheng and Doermann [28]. The purpose of their algorithm is to classify each component as either noise, hand-written text, or machine printed text from binary image inputs. Due to the complexity of implementation, we used a modified version of the CCC algorithm incorporating their MRF model by simply replacing our MRF classification model by their MRF noise classification model. The multiscale COS algorithm was applied without any change. The clique frequencies of their model were calculated through off-line training using a training data set. Other parameters were set as proposed in the paper.

We also used two commercial software packages for the comparison. The first package is the DjVu implementation contained in Document Express Enterprise version 5.1 [48]. DjVu is commonly used software for MRC compression and produces excellent segmentation results with efficient computation. By our observation, version 5.1 produces the best segmentation quality among the currently available DjVu packages. The second package is LuraDocument PDF Compressor, Desktop Version [27]. Both software packages extract text to create a binary mask for layered document compression.

The performance comparisons are based primarily on two aspects: the segmentation accuracy and the bitrate resulting from JBIG2 compression of the binary segmentation mask. We show samples of segmentation output and MRC decoded images using each method for a complex test image. Finally, we list the computational run times for each method.

A. Preprocessing

For consistency all scanner outputs were converted to sRGB color coordinates [49] and descreened [50] before segmentation. The scanned RGB values were first converted to an intermediate device-independent color space, CIE XYZ, then transformed to sRGB [51]. Then, Resolution Synthesis-based Denoising (RSD) [50] was applied. This descreening procedure was applied to all of the training and test images. For a fair comparison, the test images which were fed to other commercial segmentation software packages were also descreened by the same procedure.

B. Segmentation accuracy and bitrate

To measure the segmentation accuracy of each algorithm, we used a set of scanned documents along with corresponding "ground truth" segmentations. First, 38 documents were chosen from different document types, including flyers, newspapers, and magazines. The documents were separated into 17 training images and 21 test images, and then each document was scanned at 300 dots per inch (dpi) resolution on the EPSON STYLUS PHOTO RX700 scanner. After manually segmenting each of the scanned documents into text and non-text to create ground truth segmentations, we used the training images to train the algorithms, as described in the previous sections. The remaining test images were used to verify the segmentation quality. We also scanned the test documents on two additional scanners: the HP Photosmart 3300 All-in-One series and Samsung SCX-5530FN. These test images were used to examine the robustness of the algorithms to scanner variations.

The parameter values used in our results are as follows. The optimal parameter values for the multiscale-COS/CCC were shown in the Table I. Three layers were used for multiscale-COS/CCC algorithm, and the block sizes were 36×36, 72×72, and 144×144. The parameters for the CCC algorithm were $p = 7.806$, $a = 0.609$, $b = 0.692$. The number of neighbors in the k-NN search was 6. In DjVu, the segmentation threshold was set to the default value while LuraDocument had no adjustable segmentation parameters.

TABLE I

PARAMETER SETTINGS FOR THE COS ALGORITHM IN MULTISCALE-COS/CCC.

	λ_1	λ_2	λ_3	λ_4
2^{nd} layer	20.484	8.9107	17.778	1.0000
1^{st} layer	53.107	28.722	39.359	17.200
0^{th} layer	30.681	21.939	36.659	56.000

To evaluate the segmentation accuracy, we measured the percent of missed detections and false detections of segmented *components*, denoted as p_{MC} and p_{FC}. More specifically, p_{MC} and p_{FC} were computed in the following manner. The text in ground truth images were manually segmented to produce N_{gt} components. For each of these ground truth components, the corresponding location in the test segmentation was searched for a symbol of the same shape. If more than 70% of the pixels matched in the binary pattern, then the symbol was considered detected. If the total number of correctly detected

components is N_d, then we define that the fraction of missed components as

$$p_{MC} = \frac{N_{gt} - N_d}{N_{gt}}. \tag{17}$$

Next, each correctly detected component was removed from the segmentation, and the number of remaining false components, N_{fa}, in the segmentation was counted. The fraction of false components is defined as

$$p_{FC} = \frac{N_{fa}}{N_{gt}}. \tag{18}$$

We also measured the percent of missed detections and false detections of individual *pixels*, denoted as p_{MP} and p_{FP}. They were computed similarly to the p_{MC} and p_{FC} defined above, except the number of pixels in the missed detections (X_{MP}) and false detections (X_{FP}) were counted, and these numbers were then divided by the total number of pixels in the ground truth document (X_{total}).

$$p_{MP} = \frac{X_{MP}}{X_{total}}. \tag{19}$$

$$p_{FP} = \frac{X_{FP}}{X_{total}}. \tag{20}$$

Table II shows the segmentation accuracy of our algorithms (multiscale-COS/CCC, COS/CCC, and COS), the thresholding methods (Otsu and Tsai), an MRF-based algorithm (multiscale-COS/CCC/Zheng), and two commercial MRC document compression packages (DjVu and LuraDocument). The values in the Table II were calculated from all of the available test images from each scanner. Notice that multiscale-COS/CCC exhibits quite low error rate in all categories, and shows the lowest error rate in the missed component detection error p_{MC}. For the missed pixel detection p_{MP}, both multiscale-COS/CCC/Zheng and the multiscale-COS/CCC show the first and second lowest error rates. For the false detections p_{FC} and p_{FP}, the thresholding methods such as Otsu and Tsai exhibit the low error rates, however those thresholding methods show a high missed detection error rate. This is because the thresholding methods cannot separate text from background when there are multiple colors represented in the text.

For the internal comparisons among our algorithms, we observed that CCC substantially reduces the p_{FC} of the COS algorithm without increasing the p_{MC}. The multiscale-COS/CCC segmentation achieves further improvements and yields the smallest p_{MC} among our methods. Note that the missed pixel detection error rate p_{MP} in the multiscale-COS/CCC is particularly reduced compared to the other methods. This is due to the successful detection of large text components along with small text detection in the multiscale-COS/CCC. Large text influences p_{MP} more than small text since each symbol has a large number of pixels.

In the comparison of multiscale-COS/CCC to commercial products DjVu and LuraDocument, the multiscale-COS/CCC exhibits a smaller missed detection error rate in all categories. The difference is most prominent in the false detection error rates (p_{FC}, p_{FP}).

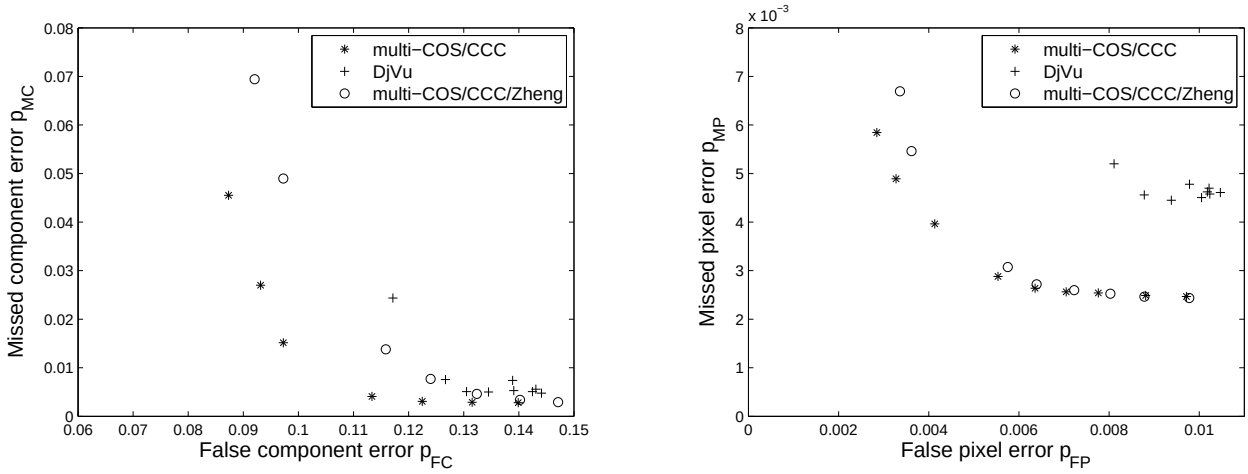
TABLE II

SEGMENTATION ACCURACY COMPARISON BETWEEN OUR ALGORITHMS (MULTI-COS/CCC, COS/CCC, AND COS), THRESHOLDING ALGORITHMS (OTSU AND TSAI), AN MRF-BASED ALGORITHM (MULTISCALE-COS/CCC/ZHENG), AND TWO COMMERCIAL MRC DOCUMENT COMPRESSION PACKAGES (DJVU AND LURADOCUMENT). MISSED COMPONENT ERROR, p_{MC}, THE CORRESPONDING MISSED PIXEL ERROR, p_{MP}, FALSE COMPONENT ERROR, p_{FC}, AND THE CORRESPONDING FALSE PIXEL ERROR, p_{FP}, ARE CALCULATED FOR EPSON, HP, AND SAMSUNG SCANNER OUTPUT.

EPSON	Multi-COS/CCC	Multi-COS/CCC/Zheng	DjVu	LuraDoc	COS/CCC	Otsu/CCC	Tsai/CCC	COS	Otsu	Tsai
p_{MC}	0.41%	0.95%	0.49%	4.64%	0.60%	2.71%	3.29%	0.53 %	4.47%	4.84%
p_{MP}	0.33%	0.27%	0.47%	0.75%	0.57%	0.55%	0.61%	0.48 %	0.64%	0.65%
p_{FC}	9.14%	9.79%	12.1%	19.5%	9.10%	9.66%	8.70%	20.1 %	25.3%	40.7%
p_{FP}	0.45%	0.54%	1.05%	6.64%	0.44%	1.60%	1.20%	3.28 %	20.4%	19.9%
HP	Multi-COS/CCC	Multi-COS/CCC/Zheng	DjVu	LuraDoc	COS/CCC	Otsu/CCC	Tsai/CCC	COS	Otsu	Tsai
p_{MC}	0.35%	1.67%	0.56%	4.84%	0.44%	3.29%	4.17%	0.47%	4.94%	5.07%
p_{MP}	0.20%	0.28%	0.48%	0.68%	0.51%	0.57%	0.61%	0.43%	0.59%	0.60%
p_{FC}	16.9%	16.8%	19.4%	41.7%	16.9%	21.4%	19.5%	45.2%	91.6%	141.2%
p_{FP}	0.70%	0.66%	1.19%	6.33%	0.62%	1.35%	1.18%	3.04%	16.4%	15.2 %
Samsung	Multi-COS/CCC	Multi-COS/CCC/Zheng	DjVu	LuraDoc	COS/CCC	Otsu/CCC	Tsai/CCC	COS	Otsu	Tsai
p_{MC}	0.44%	1.51%	0.61%	4.50%	0.48%	3.05%	8.59%	0.51%	5.01%	8.15 %
p_{MP}	0.32%	0.35%	0.44%	0.68%	0.53%	0.63%	0.78%	0.48%	0.66%	0.73 %
p_{FC}	7.95%	8.10%	11.4%	19.4%	7.95%	7.31%	6.67%	17.5%	20.2%	36.1 %
p_{FP}	0.50%	0.51%	0.81%	5.75%	0.33%	1.16%	0.67%	2.76%	19.0%	17.6 %

Figure 9 shows the trade-off between missed detection and false detection, p_{MC} vs. p_{FC} and p_{MP} vs. p_{FP}, for the multiscale-COS/CCC, multiscale-COS/CCC/Zheng, and DjVu. All three methods employ a statistical model such as an MRF or HMM for text detection. In DjVu, the trade-off between missed detection and false detection was controlled by adjustment of sensitivity levels. In multiscale-COS/CCC and multiscale-COS/CCC/Zheng method, the trade-off was controlled by the value of c_{text} in (12), and the c_{text} was adjusted over the interval $[-2, 5]$ for the finest layer. The results of Fig. 9 indicate that the MRF model used by CCC results in more accurate classification of text. This is perhaps not surprising since the CCC model incorporates additional information by using component features to determine the MRF clique weights of Eq. (11).

We also compared the bitrate after compression of the binary mask layer generated by multiscale-COS/CCC, multiscale-COS/CCC/Zheng, DjVu, and LuraDocument in Table III. For the binary compression, we used JBIG2 [52], [53] encoding as implemented in the SnowBatch JBIG2 encoder, developed



(a) Trade-off between p_{MC} and p_{FC}

(b) Trade-off between p_{MP} and p_{FP}

Fig. 9. Comparison of multiscale-COS/CCC, multiscale-COS/CCC/Zheng, and DjVu in trade-off between missed detection error vs. false detection error. (a) component-wise (b) pixel-wise

by Snowbound Software[2], using the default settings. JBIG2 is a symbol-matching based compression algorithm that works particularly well for documents containing repeated symbols such as text. Moreover, JBIG2 binary image coder generally produces the best results when used in MRC Document compression. Typically, if more components are detected in a binary mask, the bitrate after compression increases. However, in the case of JBIG2, if only text components are detected in a binary mask, then the bitrate does not increase significantly because JBIG2 can store similar symbols efficiently.

Table III lists the sample mean and standard deviation (STD) of the bitrates (in bits per pixel) of multiscale-COS/CCC, multiscale-COS/CCC/Zheng, DjVu, LuraDocument, Otsu/CCC, and Tsai/CCC after compression. Notice that the bitrates of our proposed multiscale-COS/CCC method are similar or lower than DjVu, and substantially lower than LuraDocument, even though the multiscale-COS/CCC algorithm detects more text. This is likely due to the fact that the multiscale-COS/CCC segmentation has fewer false components than the other algorithms, thereby reducing the number of symbols to be encoded. The bitrates of the multiscale-COS/CCC and multiscale-COS/CCC/Zheng methods are very similar while The bitrates of the Otsu/CCC and Tsai/CCC are low because many text components are missing in the binary mask.

[2]http://www.snowbound.com/

TABLE III

COMPARISON OF BITRATE BETWEEN MULTISCALE-COS/CCC, MULTISCALE-COS/CCC/ZHENG, DJVU, LURADOCUMENT, OTSU/CCC, AND TSAI/CCC FOR JBIG2 COMPRESSED BINARY MASK LAYER FOR IMAGES SCANNED ON EPSON, HP, AND SAMSUNG SCANNERS.

	Multi-COS/CCC		Multi-COS/CCC/Zheng		DjVu		LuraDoc		Otsu/CCC		Tsai/CCC		ground truth	
	average	STD	average	STD	average	STD	average	STD	average	STD	average	STD	average	STD
EPSON (bits/pxl)	0.037	0.014	0.037	0.014	0.040	0.014	0.046	0.016	0.037	0.016	0.036	0.016	0.037	0.014
HP (bits/pxl)	0.040	0.015	0.040	0.015	0.041	0.015	0.052	0.019	0.040	0.016	0.040	0.016	0.039	0.016
Samsung (bits/pxl)	0.035	0.015	0.035	0.015	0.036	0.015	0.041	0.016	0.035	0.016	0.034	0.017	0.036	0.016

C. Computation time

Table IV shows the computation time in seconds for multiscale-COS/CCC with 3 layers, multiscale-COS/CCC with 2 layers, COS/CCC, COS, and multiscale-COS/CCC/Zheng. We evaluated the computation time using an Intel Xeon CPU (3.20GHz), and the numbers are averaged on 21 test images. The block size on the finest resolution layer is set to 32. Notice that the computation time of multiscale segmentation grows almost linearly as the number of layers increases. The computation time of our multiscale-COS/CCC and multiscale-COS/CCC/Zheng are almost same. We also found that the computation time for Otsu and Tsai thresholding methods are 0.02 seconds for all of the test images.

TABLE IV

COMPUTATION TIME OF MULTISCALE-COS/CCC ALGORITHMS WITH 3 LAYERS, 2 LAYERS, COS-CCC, COS, AND MULTISCALE-COS/CCC/ZHENG.

	Multi-COS/CCC		COS/CCC	COS	Multi-COS/CCC/Zheng
	3 layers	2 layers			3 layers
Average	23.89 sec	16.32 sec	8.73 sec	5.39 sec	23.91 sec
STD	3.16 sec	2.12 sec	1.15 sec	0.39 sec	3.19 sec

D. Qualitative Results

Figure 10 illustrates segmentations generated by Otsu/CCC, multiscale-COS/CCC/Zheng, DjVu, LuraDocument, COS, COS/CCC, and multiscale-COS/CCC for a 300 dpi test image. The ground truth segmentation is also shown. This test image contains many complex features such as different color text, light-color text on a dark background, and various sizes of text. As it is shown, COS accurately detects most text components but the number of false detections is quite large. However, COS/CCC eliminates

most of these false detections without significantly sacrificing text detection. In addition, multiscale-COS/CCC generally detects both large and small text with minimal false component detection. Otsu/CCC method misses many text detections. LuraDocument is very sensitive to sharp edges embedded in picture regions and detects a large number of false components. DjVu also detects some false components but the error is less severe than LuraDocument. Multiscale-COS/CCC/Zheng's result is similar to our multiscale-COS/CCC result but our text detection error is slightly less.

Figure 11 and 12 show a close up of text regions and picture regions from the same test image. In the text regions, our algorithms (COS, COS/CCC, and multiscale-COS/CCC), multiscale-COS/CCC/Zheng, and Otsu/CCC provided detailed text detection while DjVu and LuraDocument missed sections of these text components. In the picture regions, while our COS algorithm contains many false detections, COS/CCC and multiscale-COS/CCC algorithms are much less susceptible to these false detections. The false detections by COS/CCC and multiscale-COS/CCC are also less than DjVu, LuraDocument, and multiscale-COS/CCC/Zheng.

Figure 13 and Fig. 14 show MRC decoded images when the encodings relied on segmentations from Ground truth, Otsu/CCC, multiscale-COS/CCC/Zheng, DjVu, LuraDocument, COS, COS/CCC, and multiscale-COS/CCC. The examples from text and picture regions illustrate how segmentation accuracy affects the decoded image quality. Note that the MRC encoding method used after segmentation is different for each package, and MRC encoders used in DjVu and LuraDocument are not open source, therefore we developed our own MRC encoding scheme. This comparison is not strictly limited to segmentation effects, but it provides an illustration of how missed components and false component detection affects the decoded images.

As shown in Fig. 13, all of the text from COS, COS/CCC, multiscale-COS/CCC, and multiscale-COS/CCC/Zheng is clearly represented. Some text in the decoded images from Otsu/CCC, DjVu, and LuraDocument are blurred because missed detection placed these components in the background. In the picture region, our methods classify most of the parts as background so there is little visible distortion due to mis-segmentation. On the other hand, the falsely detected components in DjVu and LuraDocument generate artifacts in the decoded images. This is because the text-detected regions are represented in the foreground layer, therefore the image in those locations is encoded at a much lower spatial resolution.

E. Prior Model Evaluation

In this section, we will evaluate our selected prior model. We used the initial segmentation result generated by COS with a single block size 32×32. Then we performed the CCC segmentation with the

same parameter set described in the previous section. Figure 15 shows the local conditional probability of each connected component given its neighbors' classes for two test images. The colored components indicate the foreground regions segmented by the COS algorithm. The yellowish or redish components were classified as text by the CCC algorithm, whereas the bluish components were classified as non-text. The brightness of each connected component indicates the intensity of the conditional probability which is described as $P(x_i|x_{\partial i})$. As shown, the conditional probability of assigned classification are close to 1 for most components. We observed that the components on boundaries between text and non-text regions take slightly smaller values but overall this local conditional probability map shows that the contextual model fits the test data well, and that the prior term contributes to an accurate classification.

VI. Conclusion

We presented a novel segmentation algorithm for the compression of raster documents. While the COS algorithm generates consistent initial segmentations, the CCC algorithm substantially reduces false detections through the use of a component-wise MRF context model. The MRF model uses a pair-wise Gibbs distribution which more heavily weights nearby components with similar features. We showed that the multiscale-COS/CCC algorithm achieves greater text detection accuracy with a lower false detection rate, as compared to state-of-the-art commercial MRC products. Such text-only segmentations are also potentially useful for document processing applications such as OCR.

Appendix

Feature vector for CCC

The feature vector for the connected components (CC) extracted in the CCC algorithm is a 4-dimensional vector and denoted as $y = [y_1\ y_2\ y_3\ y_4]^T$. Two of the components describe edge depth information, while the other two describe pixel value uniformity.

More specifically, an inner pixel and an outer pixel are first defined to be the two neighboring pixels across the boundary for each boundary segment $k \in \{1, \ldots, N\}$, where N is the length of the boundary. Note that the inner pixel is a foreground pixel and the outer pixel is a background pixel. The inner pixel values are defined as $X_{in}(k) = [R_{in}(k),\ G_{in}(k),\ B_{in}(k)]$, whereas the outer pixel values are defined as $X_{out}(k) = [R_{out}(k),\ G_{out}(k),\ B_{out}(k)]$. Using these definitions, the edge depth is defined as

$$edge(k) = \sqrt{||X_{in}(k) - X_{out}(k)||^2}.$$

Then, the terms y_1 and y_2 are defined as,

$$y_1 \stackrel{\text{def}}{=} \text{Sample mean of } edge(k),\ k = 1, 2, \ldots, N$$

$y_2 \stackrel{\text{def}}{=}$ Standard deviation of $edge(k)$, $k = 1, 2, \ldots, N$.

The terms y_3 and y_4 describe uniformity of the outer pixels. The uniformity is defined by the range and standard deviation of the outer pixel values, that is

$y_3 \stackrel{\text{def}}{=} \max\{O(k)\} - \min\{O(k)\}$, $k = 1, 2, \ldots, N$

$y_4 \stackrel{\text{def}}{=}$ Standard deviation of $O(k)$, $k = 1, 2, \ldots, N$

where

$O(k) = \sqrt{R_{out}(k)^2 + G_{out}(k)^2 + B_{out}(k)^2}$.

In an actual calculation, the 95^{th} percentile and the 5^{th} percentile are used instead of the maximum and minimum values to eliminate outliers. Note that only outer pixel values were examined for the uniformness because we found that inner pixel values of the connected components extracted by COS are mostly uniform even for non-text components.

The augmented feature vector of CCC algorithm contains the four components described above concatenated with two additional components corresponding the horizontal and vertical position of the connected component's center in 300 dpi, that is $z = [y_1\ y_2\ y_3\ y_4\ a_1\ a_2]^T$.

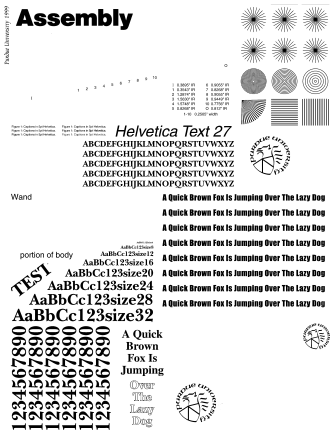
$a_1 \stackrel{\text{def}}{=}$ horizontal pixel location of a connected component's center

$a_2 \stackrel{\text{def}}{=}$ vertical pixel location of a connected component's center

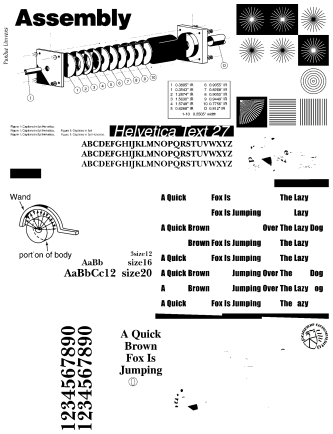
ACKNOWLEDGMENT

We would like to thank Sang Ho Kim and Jonghyon Yi at Samsung Electronics for their support of this research.

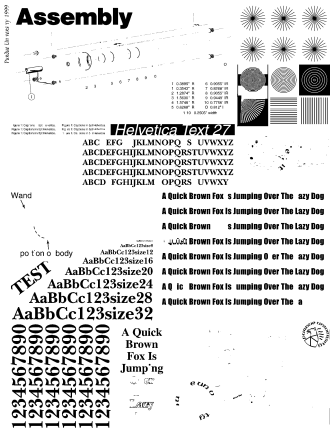
(a) Original



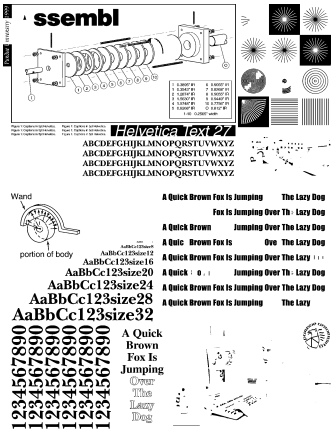
(b) Ground Truth



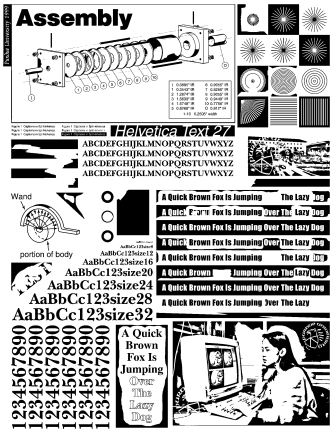
(c) Otsu/CCC



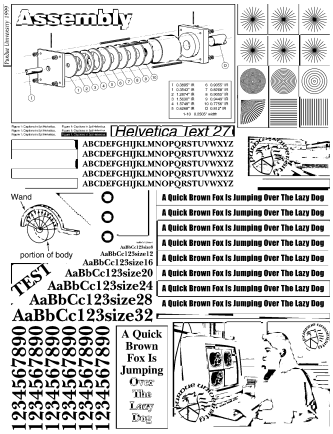
(d) Multi-COS/CCC/Zheng



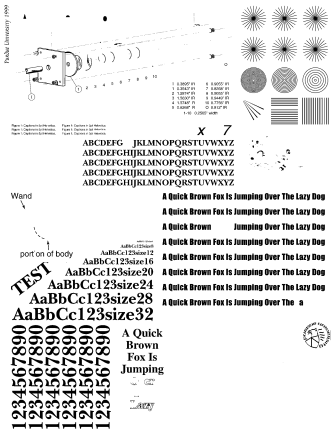
(e) DjVu



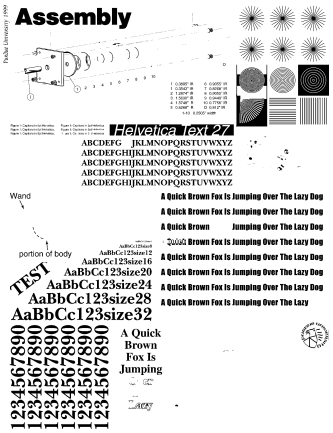
(f) LuraDocument



(g) COS



(h) COS/CCC



(i) multiscale-COS/CCC

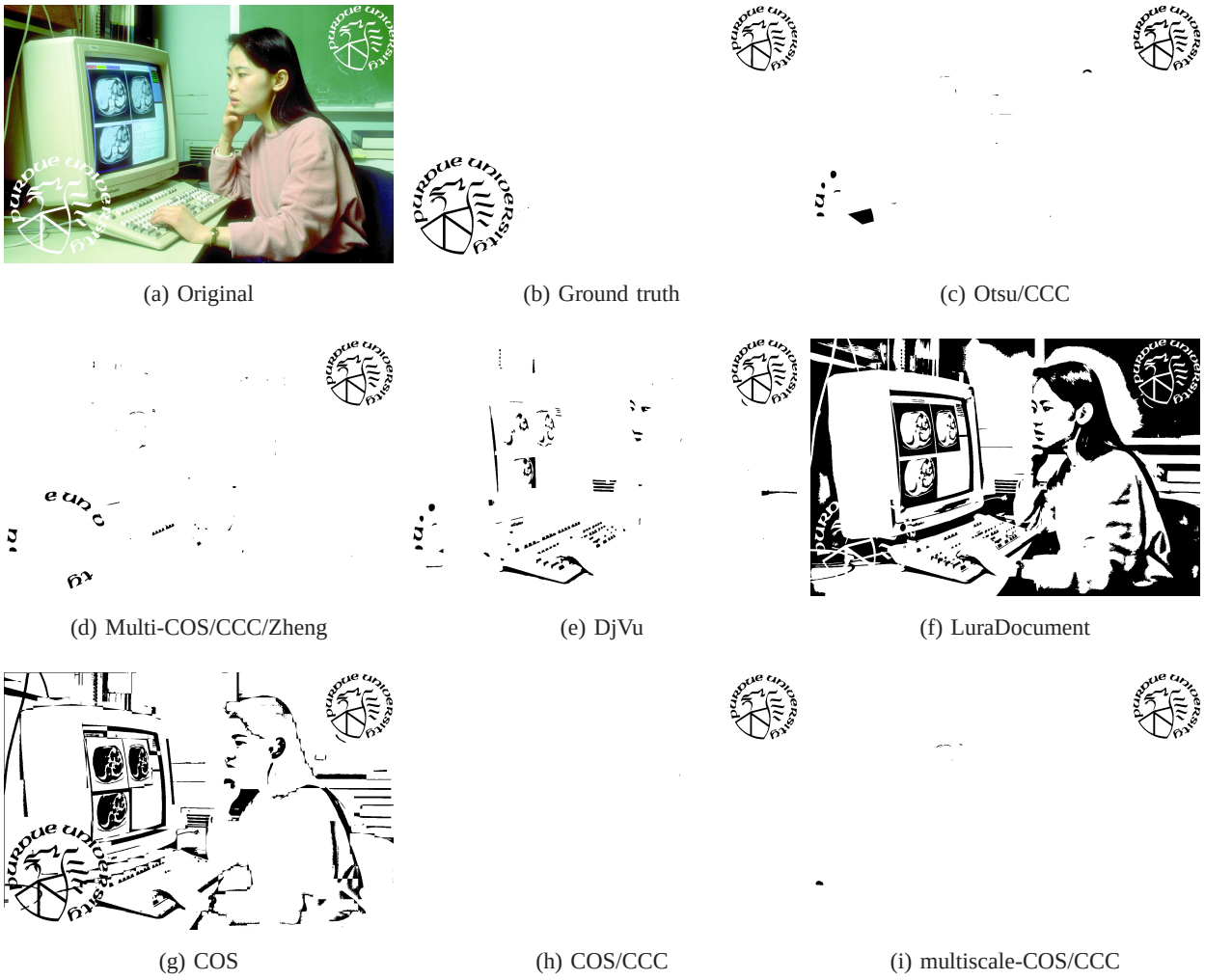
Fig. 10. Binary masks generated from Otsu/CCC, multiscale-COS/CCC/Zheng, DjVu, LuraDocument, COS, COS/CCC, and multiscale-COS/CCC. (a) Original test image (b) Ground truth segmentation (c) Otsu/CCC (d) Multiscale-COS/CCC/Zheng (e) DjVu (f) LuraDocument (g) COS (h) COS/CCC (i) Multiscale-COS/CCC

(a) Original (b) Ground truth (c) Otsu/CCC

(d) Multi-COS/CCC/Zheng (e) DjVu (f) LuraDocument

(g) COS (h) COS/CCC (i) multiscale-COS/CCC

Fig. 11. Text regions in the binary mask. The region is 165 × 370 pixels at 400 dpi, which corresponds to 1.04 cm × 2.34 cm. (a) Original test image (b) Ground truth segmentation (c) Otsu/CCC (d) Multiscale-COS/CCC/Zheng (e) DjVu (f) LuraDocument (g) COS (h) COS/CCC (i) Multiscale-COS/CCC



(a) Original (b) Ground truth (c) Otsu/CCC

(d) Multi-COS/CCC/Zheng (e) DjVu (f) LuraDocument

(g) COS (h) COS/CCC (i) multiscale-COS/CCC

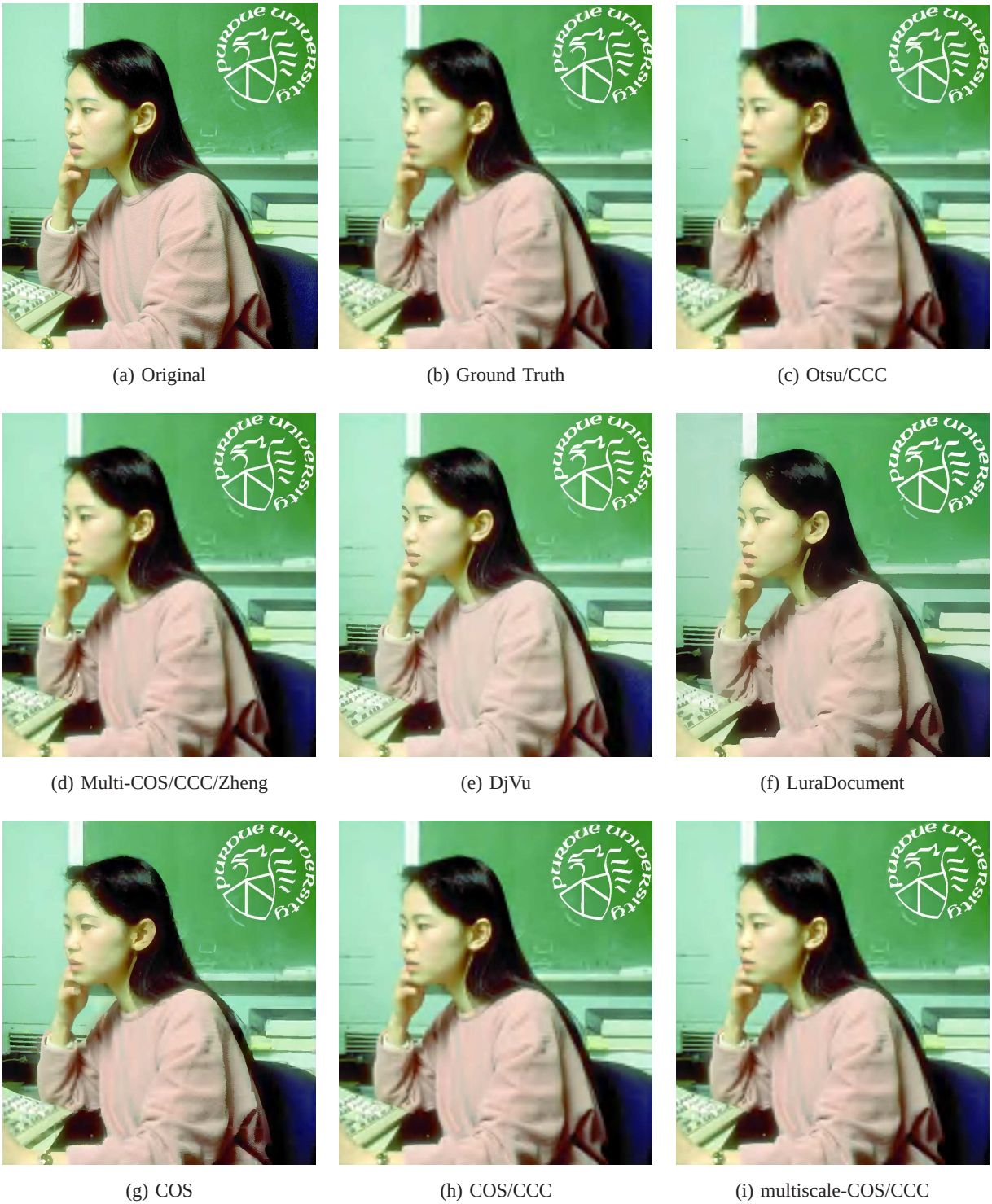
Fig. 12. Picture regions in the binary mask. Picture region is 1516 $\times$ 1003 pixels at 400 dpi, which corresponds to 9.63 cm $\times$ 6.35 cm. (a) Original test image (b) Ground truth segmentation (c) Otsu/CCC (d) Multiscale-COS/CCC/Zheng (e) DjVu (f) LuraDocument (g) COS (h) COS/CCC (i) Multiscale-COS/CCC

(a) Original (b) Ground Truth (c) Otsu/CCC

(d) Multiscale-COS/CCC/Zheng (e) DjVu (f) LuraDocument

(g) COS (h) COS/CCC (i) multiscale-COS/CCC

Fig. 13. Decoded MRC image of text regions (400 dpi). (a) Original test image (b) Ground truth (300:1 compression) (c) Otsu/CCC (311:1 compression) (d) Multiscale-COS/CCC/Zheng (295:1) (e) DjVu (281:1) (f) LuraDocument (242:1) (g) COS (244:1) (h) COS/CCC (300:1) (i) Multiscale-COS/CCC (289:1).



(a) Original (b) Ground Truth (c) Otsu/CCC

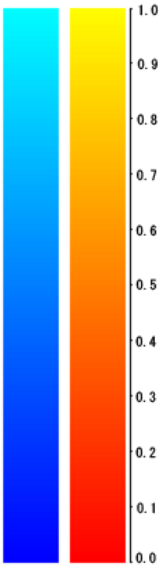
(d) Multi-COS/CCC/Zheng (e) DjVu (f) LuraDocument

(g) COS (h) COS/CCC (i) multiscale-COS/CCC

Fig. 14. Decoded MRC image of picture regions (400 dpi). (a) Original test image (b) Ground truth (300:1 compression) (c) Otsu/CCC (311:1 compression) (d) Multiscale-COS/CCC/Zheng (295:1) (e) DjVu (281:1) (f) LuraDocument (242:1) (g) COS (244:1) (h) COS/CCC (300:1) (i) Multiscale-COS/CCC (289:1).

(a) An original training image

(b) Local probabilities



(d) An original test image

(e) Local probabilities

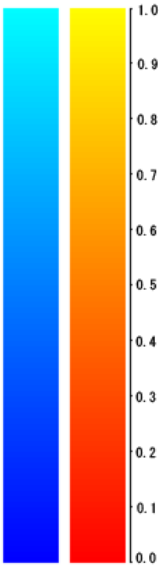


Fig. 15. The yellowish or redish components were classified as text by the CCC algorithm, whereas the bluish components were classified as non-text. The brightness of each connected component indicates the intensity of the conditional probability which is described as $P(x_i|x_{\partial i})$.

REFERENCES

[1] International Telecommunication Union, *ITU-T recommendation T.44 Mixed raster content (MRC)*, April 1999.

[2] G. Nagy, S. Seth, and M. Viswanathan, "A prototype document image analysis system for technical journals," *Computer*, vol. 25, no. 7, pp. 10–22, 1992.

[3] K. Y. Wong and F. M. Wahl, "Document analysis system," *IBM Journal of Research and Development*, vol. 26, pp. 647–656, 1982.

[4] J. Fisher, "A rule-based system for document image segmentation," in *Pattern Recognition, 10th international conference*, 1990, pp. 567–572.

[5] L. O'Gorman, "The document spectrum for page layout analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 11, pp. 1162–1173, 1993.

[6] Y. Chen and B. Wu, "A multi-plane approach for text segmentation of complex document images," *Pattern Recognition*, vol. 42, no. 7, pp. 1419–1444, 2009.

[7] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Pearson Education, 2008.

[8] F. Shafait, D. Keysers, and T. Breuel, "Performance evaluation and benchmarking of six-page segmentation algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, pp. 941–954, 2008.

[9] K. Jung, K. Kim, and A. K. Jain, "Text information extraction in images and video: a survey," *Pattern Recognition*, vol. 37, no. 5, pp. 977–997, 2004.

[10] G. Nagy, "Twenty years of document image analysis in PAMI," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 38–62, 2000.

[11] A. K. Jain and B. Yu, "Document representation and its application to page decomposition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 294–308, 1998.

[12] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE trans. on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[13] H. Oh, K. Lim, and S. Chien, "An improved binarization algorithm based on a water flow model for document image with inhomogeneous backgrounds," *Pattern Recognition*, vol. 38, no. 12, pp. 2612–2625, 2005.

[14] W. Niblack, *An introduction to Digital Image Processing*. Prentice-Hall, 1986.

[15] J. Sauvola and M. Pietaksinen, "Adaptive document image binarization," *Pattern Recognition*, vol. 33, no. 2, pp. 335–236, 2000.

[16] J. Kapur, P. Sahoo, and A. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Graphical Models and Image Processing*, vol. 29, no. 3, pp. 273–285, 1985.

[17] W. Tsai, "Moment-preserving thresholding: A new approach," *Graphical Models and Image Processing*, vol. 29, no. 3, pp. 377–393, 1985.

[18] P. Stathis, E. Kavallieratou, and N. Papamarkos, "An evaluation survey of binarization algorithms on historical documents," in *19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[19] J. M. White and G. D. Rohrer, "Image thresholding for optical character recognition and other applications requiring character image extraction," *IBM J. Res. Dev*, vol. 27, pp. 400–411, 1983.

[20] M. Kamel and A. Zhao, "Binary character/graphics image extraction: a new technique and six evaluation aspects," *IAPR International conference*, vol. 3, no. C, pp. 113–116, 1992.

[21] Y. Liu and S. Srihari, "Document image binarization based on texture features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 5, pp. 540–544, 1997.

[22] C. Jung and Q. Liu, "A new approach for text segmentation using a stroke filter," *Signal Processing*, vol. 88, no. 7, pp. 1907–1916, 2008.

[23] R. de Queiroz, Z. Fan, and T. Tran, "Optimizing block-thresholding segmentation for multilayer compression of compound images," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1461–1471, 2000.

[24] H. Cheng and C. A. Bouman, "Document compression using rate-distortion optimized segmentation," *Journal of Electronic Imaging*, vol. 10, no. 2, pp. 460–474, 2001.

[25] L. Bottou, P. Haffner, P. G. Howard, P. Simard, Y. Bengio, and Y. LeCun, "High quality document image compression with DjVu," *Journal of Electronic Imaging*, vol. 7, no. 3, pp. 410–425, 1998.

[26] P. Haffner, L. Bottou, and Y. Lecun, "A general segmentation scheme for DjVu document compression," in *Proc. of ISMM 2002*, Sydney, Australia, April 2002.

[27] "Luradocument pdf compressor," available from https://www.luratech.com.

[28] Y. Zheng and D. Doermann, "Machine printed text and handwriting identification in noisy document images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 3, pp. 337–353, 2004.

[29] S. Kumar, R. Gupta, N. Khanna, S. Chaundhury, and S. D. Joshi, "Text extraction and document image segmentation using matched wavelets and MRF model," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2117–2128, 2007.

[30] J. Kuk, N. Cho, and K. Lee, "MAP-MRF approach for binarization of degraded document image," in *Proc. of IEEE Int'l Conf. on Image Proc.*, 2008, pp. 2612–2615.

[31] H. Cao and V. Govindaraju, "Processing of low-quality handwritten documents using Markov Random Field," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1184–1194, 2009.

[32] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th International Conf. on Machine Learning*. Morgan Kaufmann, 2001, pp. 282–289.

[33] S. Kumar and M. Hebert, "Discriminative random fields: A discriminative framework for contextual interaction in classification," in *Proc. of Int'l Conference on Computer Vision*, vol. 2, 2003, pp. 1150–1157.

[34] M. C.-P. a. X. He, R.S. Zemel, "Multiscale conditional random fields for image labeling," *Proc. of IEEE Computer Soc. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 695–702, 2004.

[35] M. Li, M. Bai, C. Wang, and B. Xiao, "Conditional random field for text segmentation from images with complex background," *Pattern Recognition Letters*, vol. 31, no. 14, pp. 2295–2308, 2010.

[36] E. Haneda, J. Yi, and C. A. Bouman, "Segmentation for MRC compression," in *Color Imaging XII: Processing, Hardcopy, and Applications*, vol. 6493, San Jose, CA, 29th January 2007.

[37] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. McGraw-Hill, 2001.

[38] J. Besag, "On the statistical analysis of dirty pictures," *J. Roy. Statist. Soc. B*, vol. 48, no. 3, pp. 259–302, 1986.

[39] C. A. Bouman, "Digital Image Processing Laboratory: Markov Random Fields and MAP Image Segmentation," January 2007, available from http://www.ece.purdue.edu/~bouman.

[40] ——, "Cluster: An unsupervised algorithm for modeling Gaussian mixtures," April 1997, available from http://www.ece.purdue.edu/~bouman.

[41] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *The annals of Statistics*, vol. 11, no. 2, pp. 417–431, 1983.

[42] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *Journal of the Royal Statistical society B*, vol. 36, no. 2, pp. 192–236, 1974.

[43] ——, "Statistical analysis of non-lattice data," *The Statistician*, vol. 24, no. 3, pp. 179–195, 1975.

[44] ——, "Efficiency of pseudolikelihood estimation for simple Gaussian fields," *Biometrika*, vol. 64, no. 3, pp. 616–618, 1977.

[45] C. A. Bouman and B. Liu, "Multiple Resolution Segmentation of Textured Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 99–113, 1991.

[46] C. A. Bouman and M. Shapiro, "A multiscale random field model for Bayesian image segmentation," *IEEE Trans. on Image Processing*, vol. 3, no. 2, pp. 162–177, 1994.

[47] H. Cheng and C. A. Bouman, "Multiscale Bayesian Segmentation Using a Trainable Context Model," *IEEE Trans. on Image Processing*, vol. 10, no. 4, pp. 511–525, 2001.

[48] "Document express with djvu," available from https://www.celartem.com.

[49] International Electrotechinical Commission, *IEC 61966-2-1*, 1999.

[50] H. Siddiqui and C. A. Bouman, "Training-Based Descreening," *IEEE Trans. on Image Processing*, vol. 16, no. 3, pp. 789–802, 2007.

[51] A. Osman, Z. Pizlo, and J. Allebach, "CRT calibration techniques for better accuracy including low luminance colors," in *Proc. of SPIE Conf. on Color Imaging: Processing, Hardcopy and Applications*, vol. 5293, San Jose, January 2004, pp. 286–297.

[52] International Telecommunication Union, *ITU-T recommendation T.88 Lossy/lossless coding of bi-level images*, February 2000.

[53] P. G. Howard, F. Kossentini, B. Martins, S. Forchhammer, W. J. Rucklidge, and F. Ono, "The emerging JBIG2 standard." [Online]. Available: citeseer.ist.psu.edu/howard98emerging.html

Eri Haneda received the B.S. degree in electrical engineering from Tokyo Institute of Technology, Japan, in 1997, and the M.S. degrees from Purdue University in 2003. She is currently pursuing the Ph.D. degree at the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. From 1997 to 2000, she worked for Switching & Mobile Communication Dept. , NEC Co. Ltd., Japan. Her research interests are in statistical image processing such as image segmentation and image restoration.

Charles A. Bouman (S'86-M'89-SM'97-F'01) received a B.S.E.E. degree from the University of Pennsylvania in 1981 and a MS degree from the University of California at Berkeley in 1982. From 1982 to 1985, he was a full staff member at MIT Lincoln Laboratory and in 1989 he received a Ph.D. in electrical engineering from Princeton University. He joined the faculty of Purdue University in 1989 where he is currently the Michael J. and Katherine R. Birck Professor of Electrical and Computer Engineering. He also holds a courtesy appointment in the School of Biomedical Engineering and is co-director of Purdue's Magnetic Resonance Imaging Facility located in Purdue's Research Park.

Professor Bouman's research focuses on the use of statistical image models, multiscale techniques, and fast algorithms in applications including tomographic reconstruction, medical imaging, and document rendering and acquisition. Professor Bouman is a Fellow of the IEEE, a Fellow of the American Institute for Medical and Biological Engineering (AIMBE), a Fellow of the society for Imaging Science and Technology (IS&T), a Fellow of the SPIE professional society. He is also a recipient of IS&T's Raymond C. Bowman Award for outstanding contributions to digital imaging education and research, has been a Purdue University Faculty Scholar, and received the College of Engineering Engagement/Service Award, and Team Award. He was previously the Editor-in-Chief for the IEEE Transactions on Image Processing, and is currently a member of the Board of Governors and a Distinguished Lecturer for the IEEE Signal Processing Society. He has been an associate editor for the IEEE Transactions on Image Processing and the IEEE Transactions on Pattern Analysis and Machine Intelligence. He has also been Co-Chair of the 2006 SPIE/IS&T Symposium on Electronic Imaging, Co-Chair of the SPIE/IS&T conferences on Visual Communications and Image Processing 2000 (VCIP), a Vice President of Publications and a member of the Board of Directors for the IS&T Society, and he is the founder and Co-Chair of the SPIE/IS&T conference on Computational Imaging.