Journal of Electronic Imaging 17(4), 043011 (Oct–Dec 2008)

Document page classification algorithms in low-end copy pipeline

Xiaogang Dong
Sony Electronics Inc.
Media Processing Technology Laboratory
San Jose, California 95112

Kai-Lung Hua
Purdue University
School of Electrical and Computer Engineering
West Lafayette, Indiana 47907

Peter Majewicz
Hewlett-Packard Company
Boise, Idaho 83714

Gordon McNutt
CradlePoint, Inc.
Boise, Idaho 83702

Charles A. Bouman
Jan P. Allebach
Ilya Pollak
Purdue University
School of Electrical and Computer Engineering
West Lafayette, Indiana 47907
E-mail: ipollak@ecn.purdue.edu

Abstract. *We develop real-time, low-complexity image classification algorithms suitable for a copy mode selector embedded in a low-end copier. The algorithms classify scanned images represented in RGB or in an opponent color space. Classes are the eight combinations of mono/color and text/mix/picture/photo. Classification is 30–98% accurate with misclassifications tending to be benign. The algorithms provide for improved copy quality, a simplified user interface, and increased copy rate. © 2008 SPIE and IS&T.*
[DOI: 10.1117/1.3010879]

1 Introduction

The general workflow of a digital copier is to take scanned images from its scanner, process these images, and send them to its printer for the physical reproduction. A copier must be able to process many different kinds of originals. These originals may have different types of content, such as text, line art, graphics, and natural photos; they may be printed on different kinds of media, for example, on papers of various levels of quality and brightness; and they may be created using different rendering techniques, such as halftone or continuous tone. These different kinds of originals may interact in many different ways with various limitations of copy pipeline, such as streaks, stray light, color fringing, drift, gamut, moiré, etc. Fixed settings of the copy pipeline would therefore produce varying levels of reproduction quality, depending on the type of the original. To resolve this issue and generate user-preferred reproductions, different configurations of the copy pipeline are required. We call these different configurations "copy modes."

As illustrated in Fig. 1, a much better reproduction is obtained when the original is copied under the configuration designed specifically for its content. Consider a photograph and a fax. The configuration designed for photo originals (i.e., the photo mode) has smoothing and a wide tone curve range to achieve noise reduction and color accuracy. The configuration designed for fax originals (i.e., the text mode) has sharpening and a nearly bilevel curve to achieve an enhanced reproduction. The left panel of Fig. 1

Paper 07169RR received Aug. 14, 2007; revised manuscript received Sep. 2, 2008; accepted for publication Sep. 4, 2008; published online Nov. 12, 2008. Part of this work was presented at the 2007 IEEE International Conference on Image Processing.

1017-9909/2008/17(4)/043011/17/$25.00 © 2008 SPIE and IS&T.

Fig. 1 Processing a fax in photo and text modes. Left to right: original, photo mode, and text mode.

is the original, and the other two panels are the results of processing in the photo and text modes, respectively. Users prefer the cleaner background, increased contrast, and sharpness of the text mode and do not like the oversmoothed, low-density text, and the dirty background in the photo mode. Therefore, it is essential to match the originals with the most appropriate copy modes.

Three methods exist to match the originals to the copy modes: the "institutional-copier" method, the "defaults-to-a-single-mode" method, and the "submenus" method. A combination of the defaults-to-a-single-mode method and the submenus method is most common among the current copiers. The institutional-copier method prompts the user to describe attributes of the original to determine a matching mode. It requires a trained user. The defaults-to-a-single-mode method yields poor quality for originals that do not match that single mode, in the absence of user intervention. The submenus method provides users the opportunities of selecting modes through optional submenus. Users mostly ignore or are unaware of such options. None of the above three methods nor combinations of them meet the expectations of better qualities and simpler user interactions at the same time.

In this paper, we propose novel document page classification algorithms that can be used as copy mode selectors in a low-end (or entry-level) copy pipeline. Many existing algorithms segment the page and label each segmented region with an appropriate class label.[1–5] These algorithms require simultaneous access to the entire page image and visit each pixel multiple times. On the other hand, any algorithm applicable to our low-end copy pipeline must process image data one strip at a time and never revisit previously processed strips. This makes it impossible to apply the existing approaches.[1–5] Our proposed algorithms operate on strip-based data and make one pass over each strip. The entire page is classified after all its strips have been processed. The computational complexity and memory requirements of our algorithms are both very low. These advantages make our algorithms ideal candidates to be implemented in existing low-end copy pipelines with only slight changes of the hardware.

In the rest of the paper, we describe and illustrate our algorithms. Section 2 provides the general overview of the RGB algorithm, Section 3 describes its specifics, and Section 4 contains the results of applying it to a large database of images. Section 5 presents our algorithms for an opponent color space.

2 Algorithm Overview

Figure 2 illustrates a possible implementation of our classification algorithm. The classifier resides between the front end and the back end of the pipeline, and directs the image data to the most appropriate processing mode. The revised copy pipeline not only improves copy quality and simplifies user interaction, but also significantly increases copy rate in at least one case, as follows. The pipeline reconfigures for each page in a multipage copy job from an automatic document feeder (ADF). When it detects a black-and-white page inside the job, it implements 3-to-1 channel reduction and enters the faster mono pipeline mode.

We work with a specific copy pipeline equipped with eight different copy modes, which are all possible combinations of mono/color and text/mix/picture/photo. Mono mode is a configuration optimized for mono originals, whereas color mode is optimized for color originals. Text mode is optimized for text, line art, simple graphics, handwritten text, and faxes. Simple graphics do not contain gradients into highlights or shadows and typically contain a small number of colors. Picture mode is for high-dynamic-range halftoned originals, such as glossy magazine pages. Photo mode is for continuous-tone natural scenes on photographic paper, such as prints developed using silver halide (AgX) process. The originals from the highest resolution inkjet printers are also considered photos (or near-continuous tone) when they are scanned using our low-end copy pipeline. Mix mode is for originals containing both text and picture content. Mix mode is also preferred for complex graphics with pastels or gradients into highlights or shadows. Our goal is to classify the scanned image of the original into the following eight distinct classes: color-text, color-mix, color-picture, color-photo, mono-text, mono-mix, mono-picture, and mono-photo. For simplicity, we use the names of the copy modes as the names of candidate classes.

Misclassifying an original from one class as any of the other seven classes is an error; however, not all misclassification errors are equally costly. For example, misclassifying color originals as mono is considered harmful, because important color information is lost when color originals are copied under the mono mode. On the other hand, there may exist some colors near the neutral axis in the reproductions of mono originals under the color mode, but still the reproduction would not look much worse than a mono reproduction. Therefore, misclassifying mono originals as color is

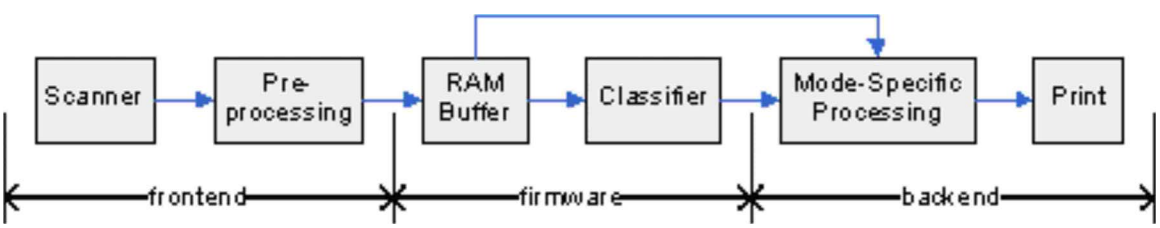


Fig. 2 One possible implementation of our page classification algorithm in an existing copy pipeline.

Table 1 Types of classification results: C represents correct classification, B represents benign errors and H represents harmful errors.

	Classification results							
Ground-truth	Color text	Color mix	Color picture	Color photo	Mono text	Mono mix	Mono picture	Mono photo
Color-text	C	B	H	H	H	H	H	H
Color-mix	H	C	H	H	H	H	H	H
Color-picture	H	B	C	H	H	H	H	H
Color-photo	H	B	H	C	H	H	H	H
Mono-text	B	B	H	H	C	B	H	H
Mono-mix	H	B	H	H	H	C	H	H
Mono-picture	H	B	B	H	H	B	C	H
Mono-photo	H	B	H	B	H	B	H	C

considered a benign error. For text/mix/picture/photo classifications, copying text or mix originals under the picture or photo mode is considered a harmful error because the letters in a text are significantly softened in such reproductions. Copying mix, picture, or photo originals under the text mode is harmful because the reproductions are close to bilevel images, and therefore, pictures and photos are very poorly reproduced. Copying pictures under the photo mode is harmful because of the presence of moiré artifacts. Copying photos under the picture mode is also considered harmful due to color mismatches in the reproductions. Because the mix mode is specially designed for originals with both text, pictures, and/or photos, copying text, picture, or photo originals under the mix mode is considered "benign," although the reproduction of text under the mix mode may not be as sharp as that under the text mode, and the reproductions of pictures and photos under the mix mode may not look as good as those under the picture and photo modes, respectively. Table 1 summarizes our definitions of harmful and benign errors.

We follow a treelike decision structure illustrated in Fig. 3. It divides the classification problem into several smaller, simpler subproblems. A mono versus color classifier is placed at the root level of the decision tree. Then, text versus nontext classifiers, text/mix versus picture/photo classifiers, and mix versus picture versus photo classifiers are deployed at the second, third, and fourth levels of the decision tree. Here, "nontext" refers to all classes other than text (i.e., mix, picture, and photo). As described in the previous paragraph, misclassifying nontext documents as text is harmful while misclassifying text documents as mix is benign. Therefore, text versus nontext classifiers are designed to minimize misclassifications of nontext originals. As a result, there may be a significant number of misclassified text documents that propagate onto the third-level classifiers. In order to prevent these text images from being classified as photo or picture, we design the third-level classifiers to be able to distinguish not only mix documents but also text documents from photos and pictures. The third-level classifiers are biased toward text/mix documents so that the number of harmful misclassifications can be minimized. For a similar reason, we use mix versus picture versus photo classifiers instead of simply picture versus photo classifiers at the fourth level. Note that starting from the second level of the decision tree, the two classifiers at the same level share the same methodology; however, they may use different sets of tunable parameters. Placement of these classifiers in the decision tree reflects the frequency of documents a copier encounters. Text documents are most frequently copied, followed by mix, picture, and photo documents. In Sec. 3, we discuss each of these classifiers. Our use of a tree-structured classification scheme with binary and ternary classifiers at the nodes of the tree is motivated by the desire to reduce a large and complex problem

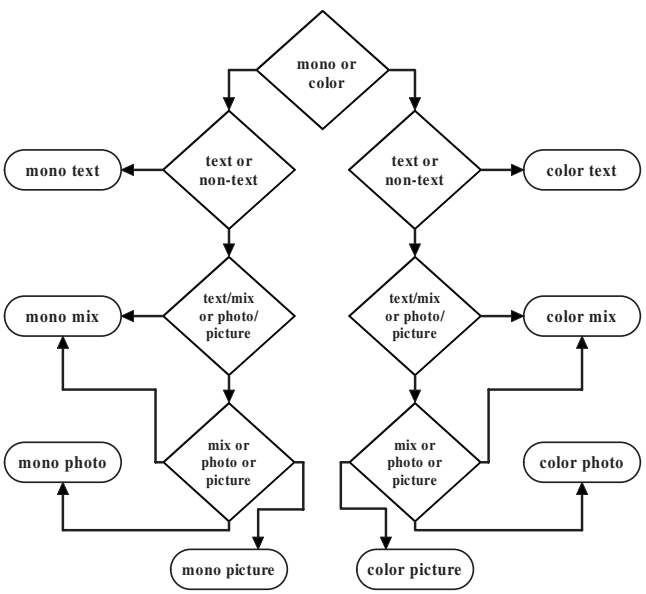
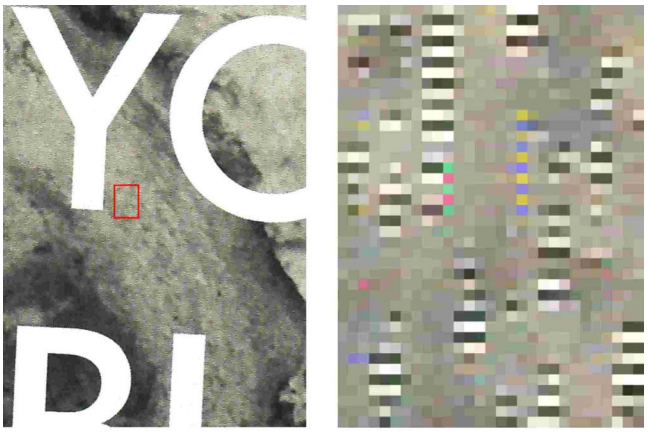


Fig. 3 Classification decision tree.



(a) (b)

Fig. 4 (a) Image patch from the scanned image of a mono original. (b) Enlarged version of the red rectangle shown in the left figure. (Color online only.)

to several simpler parts. In addition, the resulting simple classifiers satisfy the requirements of low time and memory complexity imposed by our low-end copy pipeline. Furthermore, the tree-structured scheme lends itself to early termination and to partial implementation. Both characteristics are important for time-critical and resource-limited applications.

3 RGB Algorithm Details

3.1 *Color Versus Mono Classifier*

All image data are scanned at 300 dots per inch (dpi), gamma-corrected, and represented in the RGB color space with eight bits per channel. Note that the RGB data is not calibrated to some standard color space, such as sRGB. Therefore, our algorithms need parameter retuning when applied to different copy pipelines. We have a data set containing 891 images with at least 100 images for each of eight classes mentioned in the previous section. All images in the data set were carefully selected by HP engineers to include a wide variety of difficult-to-classify scenarios, such as color images with very few color pixels in them, picture-text mixtures with very few small text patches, etc. Each image was classified by hand by at least one of the HP engineers. In many cases to label an image as text, mix, picture, or photo, the original was copied in every mode, and the mode giving the best appearance was selected. When there was no clear preference among the modes, the image was labeled as mix. Note that we did not perform any psychometric testing.

Color noise typically introduced by scanners makes it challenging to construct a reliable classifier for distinguishing between mono and color originals based on their scanned images. This is illustrated in Fig. 4. Figure 4(a) shows a portion of the scanned image of a mono document. A small rectangular patch of this image is outlined in red and is enlarged in Fig. 4(b), clearly showing color noise (i.e., many color pixels). On the other hand, any page that truly has color in it, even a small amount, must be classified as color, because copying a color original in the mono mode would lead to the loss of important color information. In addition to these conflicting requirements of robustness to color noise and color classification accuracy, we are faced with the requirement of low computational complexity stemming from the fact that we are targeting low-end machines.

Among previously proposed methods for the color-mono page-classification problem, most make the classification decisions at the pixel level, then aggregate the decisions over larger regions (e.g., lines or blocks), and then finally make a decision for the entire page.[6–20] (Additional literature addresses the issue of classifying individual pixels.[21–23]) The essence of our algorithm is similar: we divide an image into blocks, classify each block as mono or color, based on the average distance of its pixels to a gray color, and declare the image to be mono only if every block is mono. Among the existing methods, our method most closely resembles the page- classification strategy proposed in Ref 6. Two key differences from the existing methods is that we avoid classifying individual pixels and that we classify the entire page as color if at least one block is classified as color. These features of our algorithm enable us to only have one tunable threshold.

The specifics of our algorithm are as follows. We first define the gray line in RGB color space as the straight line satisfying $R=G=B$. In other words, we say a pixel p is gray if $R(p)=G(p)=B(p)$, where $R(p)$, $G(p)$, and $B(p)$ are the pixel's red, green, and blue intensities, respectively. We then define the colorfulness $C(p)$ of p as the following quantity:

$$C(p) = \max[R(p),G(p),B(p)] - \min[R(p),G(p),B(p)]. \quad (1)$$

Note that $C(p)=0$ if, and only if, the pixel is gray. Note also that the quantity $C(p)$ may be interpreted as the "Manhattan distance" (or the ℓ_1 distance) from the point $[R(p),G(p),B(p)]$ to the gray line, i.e.,

$$C(p) = \min_{0\leqslant\eta\leqslant 255} (|R(p)-\eta| + |G(p)-\eta| + |B(p)-\eta|).$$

The colorfulness $C(b)$ of a block of pixels b is defined as the sum of $C(p)$ over all the pixels p that belong to the block b. We partition the image into blocks and define the colorfulness of the image, C_{image}, as the maximum of $C(b)$ over all blocks b. We classify an image as color if C_{image} is larger than a threshold, and we otherwise classify it as mono. We select the threshold to be just below the minimum colorfulness among all the color images in the training set. This ensures that all the color images in the training set are correctly classified. Recall from Section 2 that misclassifying color as mono is harmful, whereas misclassifying mono as color is benign. Our threshold selection therefore ensures that the number of benign errors on the training data is minimized, subject to no harmful errors on the training data.

Our motivation for summing $C(p)$ over pixels in a block is to achieve robustness to color noise. The motivation for maximizing $C(b)$ over all blocks is the fact that even if there is a single color block in the document, we would like to classify the entire document as color. These two considerations—the robustness to color noise and the desire to have no misclassifications of color images—are in conflict with each other, with the second consideration being

Table 2 Percentage of correctly classified mono images in the training set, under maximum threshold guaranteeing 100% correct classification of color images, for various block sizes, when image data is represented in the RGB color space. 470 color images and 421 mono images are used in these experiments.

	Percentage of correctly classified mono images		
Block size	No filter	3×3 filter	5×5 filter
8×8	67%	69%	69%
16×16	89%	91%	91%
32×32	89%	90%	90%
64×64	72%	75%	76%

more important because we would like to avoid harmful errors. Thus, mono images with a significant enough level of color noise will be misclassified as color. However, as shown in Table 2, the number of such misclassifications on our data set is quite low.

Another important parameter of the algorithm is the block size. Large block sizes improve the robustness of the algorithm to noise. On the other hand, small color regions may be missed if the blocks are too large. One extreme example is a document containing all black-and-white text except for one letter, which is green. In order to select the most appropriate block size, we perform several experiments. We fix a block size, set the decision threshold at C^*-1 where C^* is the minimum colorfulness among all the color images in the training set, and compute the percentage of correctly classified mono images in the training set. We try four different block sizes 8×8, 16×16, 32×32, and 64×64. (Note that our procedure for choosing the decision threshold means that experiments with different block sizes may use different thresholds.) The results for these experiments are shown in Table 2, indicating that 16×16 and 32×32 are the best choices among these four. In both cases, 89% of mono images and 100% of color images are correctly classified. The intuition behind this result is that 64×64 blocks are too large to be effective in detecting small color patches, and therefore we must sacrifice many misclassifications of mono images to keep the correct color image classification rate at 100%. On the other hand, 8×8 blocks are too small to suppress color noise, and therefore, many mono images get misclassified as color due to noise.

Note that there are many strategies described in the existing literature that could potentially enhance the classification performance of our classifier at the expense of a higher computational complexity, and which we therefore are unable to use due to complexity restrictions imposed by our low-end application. For example, low-pass filters have been used to reduce color noise.[21,23] Using a 3×3 or a 5×5 averaging filter in our algorithm improves the classification results by 1–4% (see Table 2) at the expense of three to seven times more computations per pixel. One reason for such a small improvement in performance is perhaps the fact that some low-pass filtering is already built into our algorithm, via summing up the colorfulness values of pixels to determine the colorfulness of a block. Another example of something that we would like to avoid because of computational cost is color space conversions.[14,16–18,21,23,24] We conclude this section by mentioning two existing strategies that are very different from both the rest of the literature and from our method: One compares bit rates of losslessly compressed C, M, Y, and K components,[24] the other uses global color histograms.[25]

3.2 *Text Versus Nontext Classifier*

As previously defined, nontext refers to mix, picture, and photo. We use two properties of text documents in order to distinguish them from documents that contain pictures or photos. First, the histogram for a typical text region has peaks that are more narrow than the peaks in a typical picture or photo histogram. Second, nontext areas of a text document typically contain only a few colors. On the basis of these two properties, we define two scores: the histogram flatness score and the color variability score. We classify images by thresholding these two scores.

To motivate the definition of the histogram flatness score, we illustrate two image patches in Figs. 5(a) and 5(b). The first patch is from the scanned image of a text document, and the other patch is from the scanned image of a photo document. The corresponding histograms of the red intensities are shown in Figs. 5(c) and 5(d), respectively. The histogram of the text patch has two peaks, one near white and one near black. The histogram of the photo patch is spread more evenly over all intensities. If we measure the width of the peaks of the two histograms at a fixed height, then we would therefore expect the histogram of the photo patch to have larger widths. This is illustrated in Figs. 5(c) and 5(d) for the height marked by the red lines. Note that Fig. 5 only shows the histograms for the red channel. The green and blue histograms behave similarly.

We extract the histogram flatness features via the following procedure. We partition the image into 64×64 blocks and calculate 64-bin R, G, and B histograms for each block. The histogram bins have equal widths. For every eight-bit pixel value, we simply right-shift the value by two bits to find the corresponding histogram bin. Although, using 64-bin histograms requires slightly more computation than using 256-bin histograms; 64-bin histograms require 75% less memory. In our specific pipeline, image data is in raster order of width 2560 pixels (i.e., each horizontal strip of data covers 40 blocks of width 64 pixels). We thus need to hold $40\times3=120$ histograms in memory at any time (three color histograms for each block) which is a significant amount of memory in our application.

The k-span of a histogram is defined as the largest number of consecutive bins in the histogram whose values are $\geqslant k$. The k-span of an image is defined as the largest k-span over all 64-bin R, G, and B histograms of its blocks. For each image, we form a multidimensional feature vector $\mathbf{x}$ consisting of k-spans of the image for a number of different k values. We use $k=15,30,\ldots,150$. As further discussed below at the end of this subsection, we have selected these values based on several experiments with different settings for the block size, the number of bins per histogram, and k values, as summarized in Tables 3–5.

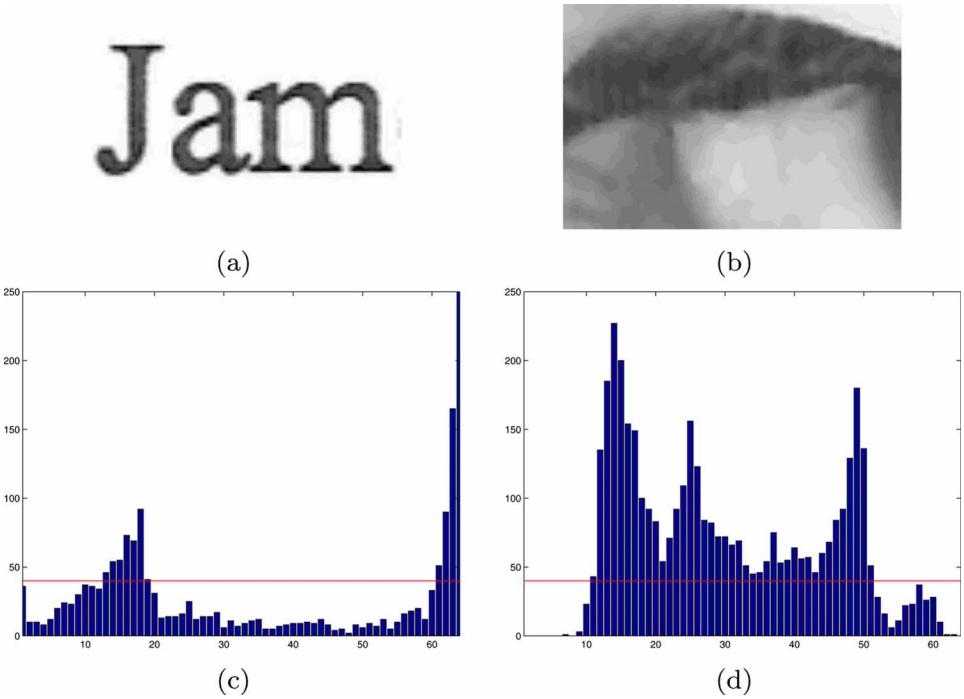


Fig. 5 (a) Image patch from the scanned image of a text document. (b) Image patch from the scanned image of a photo document. (c) The 64-bin *R* histogram of the text image patch. (d) The 64-bin *R* histogram of the photo image patch.

This feature extraction procedure is used to calculate the histogram flatness score for any image as follows. We assume that the text and nontext feature vectors are Gaussian with means $\mathbf{m}_{\text{text}}$ and $\mathbf{m}_{\text{nontext}}$, respectively, and with a common covariance matrix Λ_F. For such binary classification problems, the maximum *a posteriori* probability (MAP) classification of any feature vector $\mathbf{x}$[26] is obtained through a likelihood ratio test (i.e., by comparing to a threshold the logarithm of the ratio of the likelihood of the nontext model to the likelihood of the text model). If this log likelihood ratio exceeds the threshold, then the feature vector is classified as nontext; otherwise, it is classified as text. In our Gaussian case with a common covariance, the log likelihood ratio is, up to an additive constant,

$$
\begin{aligned}
L &= \log\{\text{Prob}(\mathbf{x}|\text{assume } \mathbf{x} \text{ is nontext})\} - \log\{\text{Prob}(\mathbf{x}|\text{assume } \mathbf{x} \text{ is text})\} \\
&= \log\{(2\pi)^{-0.5D} \det(\Lambda_F)^{-0.5} \exp[-0.5(\mathbf{x}-\mathbf{m}_{\text{nontext}})^T \Lambda_F^{-1}(\mathbf{x}-\mathbf{m}_{\text{nontext}})]\} \\
&\quad - \log\{(2\pi)^{-0.5D} \det(\Lambda_F)^{-0.5} \exp[-0.5(\mathbf{x}-\mathbf{m}_{\text{text}})^T \Lambda_F^{-1}(\mathbf{x}-\mathbf{m}_{\text{text}})]\} \\
&= -0.5(\mathbf{x}-\mathbf{m}_{\text{nontext}})^T \Lambda_F^{-1}(\mathbf{x}-\mathbf{m}_{\text{nontext}}) + 0.5(\mathbf{x}-\mathbf{m}_{\text{text}})^T \Lambda_F^{-1}(\mathbf{x}-\mathbf{m}_{\text{text}}) \\
&= \text{const} + (\mathbf{m}_{\text{nontext}} - \mathbf{m}_{\text{text}})^T \Lambda_F^{-1}\mathbf{x},
\end{aligned}
$$

where T denotes the transpose of a vector, D denotes the dimension of a vector, and $\det(\cdot)$ denotes the determinant of a matrix. The likelihood ratio test is therefore equivalent to thresholding the quantity $(\mathbf{m}_{\text{nontext}}-\mathbf{m}_{\text{text}})^T\Lambda_F^{-1}\mathbf{x}$. The means of text and nontext feature vectors as well as the common covariance matrix can be estimated based on labeled training data. We use $\hat{\mathbf{m}}_{\text{nontext}}$ and $\hat{\mathbf{m}}_{\text{text}}$ to denote the estimated means of the nontext and text feature vectors, respectively, and $\hat{\Lambda}_F$ to denote the estimated common covariance matrix of the text and nontext feature vectors. For any image whose feature vector is $\mathbf{x}$, we then define its histogram flatness score as $F \equiv (\hat{\mathbf{m}}_{\text{nontext}}-\hat{\mathbf{m}}_{\text{text}})^T\hat{\Lambda}_F^{-1}\mathbf{x}$. A small histogram flatness score implies peakiness of the histogram and suggests that the image is text, whereas a large histogram flatness score indicates that the image is nontext. Note that $(\hat{\mathbf{m}}_{\text{nontext}}-\hat{\mathbf{m}}_{\text{text}})^T\hat{\Lambda}_F^{-1}$ can be precalculated and stored in the memory to make the number of online calculations required to compute F very small.

Figure 6 motivates the definition of the second score we use in the text-versus-nontext classification, namely, the color variability score. Figure 6(a) shows the scanned image of a text document. The nontext area of the image is simply the white background. On the other hand, if the nontext area of a document contains many colors (or many gray levels in the case of mono originals), then we would want to classify it as nontext. One such example is illustrated in Fig. 6(b). The computation of the color variability score involves the following steps:

1. Partition the image into 8×8 blocks.
2. Identify the nontext blocks.
3. Compute the R, G, and B means of every nontext block.
4. Construct the 256-bin histogram of block means over the entire image, for each of the three color channels. These histograms, for the R channel, are shown in Figs. 6(c) and 6(d) for the images in Figs. 6(a) and 6(b), respectively.
5. Define the color variability score as the largest number of bins whose values are greater than or equal to a threshold η, among the three histograms.

Note that we use 256 bins for block-mean histograms, as

Table 3 Percentages of correctly classified text images for different block sizes, bin sizes for within-block histograms, and sets of thresholds k, for 100% correct classification of nontext images. Other parameters are the same as in the experiments shown in Fig. 7.

Block size	No. Bins	k	Percentage of correctly classified text images: Color	Mono
8×8	32	2,4,…,20	63%	64%
16×16	64	1,2,…,10	64%	80%
32×32	64	3,6,…,30	65%	84%
64×64	64	15,30,…,150	68%	85%

Table 4 Percentages of correctly classified text images for different bin sizes for within-block histograms and sets of thresholds k, for 100% correct classification of nontext images. Other parameters are the same as in the experiments shown in Fig. 7.

No. Bins	k	Percentage of correctly classified text images: Color	Mono
256	2,4,…,20	68%	82%
128	3,6,…,30	67%	86%
64	15,30,…,150	68%	85%
32	24,48,…,240	68%	85%
16	50,100,…,500	70%	83%

opposed to a smaller number of bins, since there are only three such histograms and memory is not a significant concern, unlike in the case of within-block histograms defined above. Experiments also show that 256-bin histograms perform better than histograms with larger bin sizes, as shown in Table 6. Based on some other experiments (see Table 7), we choose $\eta=1$ which means that the color variability score is defined as the largest number of nonzero bins among the three histograms. If the color variability score is low, then every histogram has only a few nonzero entries, as in Fig. 6(c), suggesting that nontext regions in the image tend to be synthetic graphics in few primary colors rendered without much texture. If this score is high, there is at least one histogram with many nonzero bins, as in Fig. 6(d), indicating the presence of many colors (or many gray levels if the originals are mono) in the image. In this case, copying in the text mode would be inappropriate.

To complete the specification of the color variability score, we must specify how the identification of the nontext blocks works in step 2. A text edge in the red channel is defined as three consecutive pixels p_1, p_2, and p_3, in either horizontal or vertical direction, satisfying the following two conditions:

- $R(p_1)<R(p_2)<R(p_3)$ or $R(p_1)>R(p_2)>R(p_3)$,
- $|R(p_1)-R(p_3)|>T$,

where $R(p_i)$ represents the red intensity of p_i for $i=1,2,3$ and T is a large threshold (i.e., we define text edges as very significant color changes to reflect that the color of the text is usually very different from the color of the background on which the text is printed). In other words, (p_1,p_2,p_3) is a text edge in the red channel if $R(p_1)$, $R(p_2)$, and $R(p_3)$ are monotonically increasing or decreasing and the absolute value of the difference between the first and the third exceeds the threshold T. Text edges in the green and blue channels are defined similarly. A nontext block is any image block that does not contain any text edges. An important parameter in the definition of the text edges is the threshold T. We choose $T=100$ based on some experiments summarized in Tables 8 and 9.

In order to classify an image, we set thresholds for both the histogram flatness score and the color variability score, based on the training data. We classify an image as text if both scores are smaller than their corresponding thresholds. Otherwise, we classify it as nontext. Note that copying nontext originals in text mode is harmful, whereas copying text originals in the mix mode is benign. Therefore, we must choose the thresholds conservatively so that as few nontext documents as possible are misclassified as text.

In our experiments, we train two different sets of $\hat{\mathbf{m}}_{\text{text}}$, $\hat{\mathbf{m}}_{\text{nontext}}$, and $\hat{\Lambda}_F$ for color originals and mono originals. As a result, we have two different sets of thresholds for color originals and mono originals. We build 64-bin histograms for each block of size 64×64 and let $k=15,30,\ldots,150$. We also build 256-bin block-mean histograms and let the threshold for the color variability scores $\eta=1$, the threshold for the text edges $T=100$. Figure 7(a) shows the scatter plot of histogram flatness and color variability scores for the color originals in the training suite. Blue O's represent text documents, and red X's represent nontext documents. The selection of the decision boundaries for this and other decision rules in our system is application dependent and involves subjective evaluation of the trade-offs between various classification errors. We typically choose to set

Table 5 Percentages of correctly classified text images for different sets of thresholds k, for 100% correct classification of nontext images. Other parameters are the same as in the experiments shown in Fig. 7.

k	Percentage of correctly classified text images: Color	Mono
9,18,…,90	64%	84%
12,24,…,120	69%	84%
15,30,…,150	68%	85%
18,36,…,180	68%	84%
21,42,…,210	68%	80%

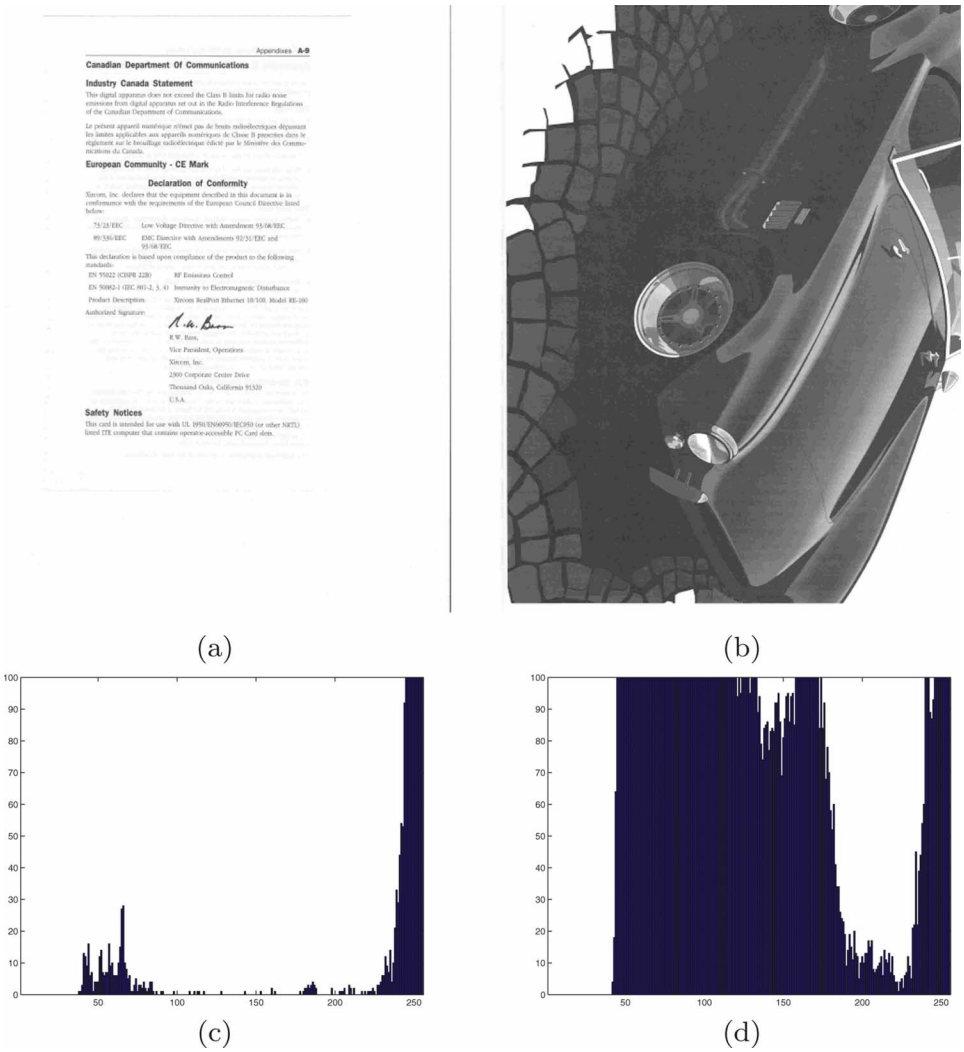


Fig. 6 (a) The scanned image of a text document. (b) The scanned image of a picture document. (c) The *R* histogram of block means for the text document. (d) The *R* histogram of block means for the picture document.

thresholds in such a way that we get zero "harmful" misclassifications, as shown in Fig. 7(a). The solid cyan line is a decision boundary that results in all color nontext documents being correctly classified. With this boundary, 68% of color text documents are correctly classified. With the dot-dashed magenta line similarly chosen in Fig. 7(b) for mono text-versus-nontext classification, 77% of mono text documents are correctly classified. We in addition provide the solid cyan boundary in Fig. 7(b), which misclassifies one mono nontext document but improves the correct classification rate for mono text documents to 85%. In our particular application, we deem it acceptable to misclassify this nontext document as text in exchange for such a significant improvement in the correct classification rate for mono text documents. In addition, in Section 4, we propose an automated method to select decision boundaries, which globally minimizes the following criterion:

$$(\text{number of benign errors}) + \alpha(\text{number of harmful errors}),$$

where α is a number indicating how many benign errors we are willing to trade for a single harmful error.

We have also tested a linear classifier to separate text and nontext documents. The dashed black lines in Figs. 7(a) and 7(b) illustrate the decision boundaries for the linear classifier. The correct classification rate for color text images is 62% with no misclassification error of color nontext images, 6% lower than the rate obtained by using the cyan decision boundary in Fig. 7(a). The correct classification rate for mono text images is 79% with only one mono nontext image misclassified as text. It is also 6% lower than the rate obtained by applying the solid cyan decision line in Fig. 7(b). Therefore, in this case, linear classifiers do not perform as well as the classifiers that threshold two scores individually.

The following parameters are used in this subsection: block size, bin size for within-block histograms, thresholds k for histogram spans, bin size for block-mean histograms, threshold η used in the definition of the color variability score, and threshold T used in the definition of a text edge. We have conducted experiments with various combinations of these parameters and summarize the results in Tables 3–8. We consider the following block sizes: 8×8, 16×16,

Table 6 Percentages of correctly classified text images for different bin sizes of block-mean histograms, for 100% correct classification of nontext images. Other parameters are the same as in the experiments shown in Fig. 7.

	Percentage of correctly classified text images	
# Bins	Color	Mono
256	68%	85%
128	68%	82%
64	65%	76%

32×32, and 64×64. We do not consider larger block sizes due to memory limitations of our low-end copy pipeline. We vary the histogram bin size and the set of thresholds k for histogram spans. The best results for each block size are listed in Table 3. In these experiments, $\eta=1$, $T=100$, and the bin size for block-mean histograms is 256. For a 64×64 block size, we consider different bin sizes of within-block histograms as well as different sets of thresholds k and summarize the best results in Table 4. Tables 5–8 show the results for varying the set of thresholds k, the threshold η, the threshold T, and the bin size of block-mean histograms, respectively. On the basis of these experiments, we select 64×64 block size, 64-bin within-block histograms, 256-bin block-mean histograms, $k=15,30,\ldots,150$, $\eta=1$, and $T=100$. Note that our algorithm is quite robust to parameter changes, i.e., similar sets of parameters result in similar misclassification rates, as evidenced by Tables 3–8.

As with our color-mono classifier, the text-nontext classifier can be outperformed by using various ideas from the existing literature, at the expense of added computational complexity, which is prohibitive in our low-end application. For example, the mixed raster content (MRC) model[27] is a multilayered and multiresolution image model for image compression which separates foreground from background and could be used to construct a better approach to text-nontext classification. However, it cannot be implemented in our low-end copy pipeline due to our requirement of strip-based data structures, limited computational capability, and limited memory in the pipeline.

Table 7 Percentages of correctly classified text images for different values of the threshold η used in the definition of the color variability score, for 100% correct classification of nontext images. Other parameters are the same as in the experiments shown in Fig. 7.

	Percentage of correctly classified text images	
η	Color	Mono
1	68%	85%
3	69%	82%
5	68%	81%
10	69%	82%

Table 8 Percentages of correctly classified text images for different values of the thresholds T used in the definition of a text edge, for 100% correct classification of nontext images. Other parameters are the same as in the experiments shown in Fig. 7.

	Percentage of correctly classified text images	
T	Color	Mono
90	71%	85%
95	69%	85%
100	68%	85%
105	68%	85%
110	68%	84%

3.3 *Text/Mix Versus Picture/Photo Classifier*

Pictures and photos contain only natural scenes and no text, as illustrated in Fig. 8(a). Figure 8(b) shows a mix image with some text and non-natural scenes, for example, the cartoon of a snowman in the image. Our strategy for distinguishing picture and photo images from mix and text documents is to detect text and other regions that do not look like natural scenes.

As in Section 3.2, we define a text edge to be a vertical or a horizontal sequence of three pixels with monotone values, such that the absolute value of the difference between the first and third values is larger than a threshold. In addition, we would like to avoid counting a single-edge multiple times. (This was not an issue in Section 3.2, since in that section we only wanted to determine whether at least one text edge was present in a block.) For example, the

Table 9 Percentage of correctly classified picture/photo images for different T. Other parameters are the same as in the experiments of Fig. 9

	Percentage of correctly classified picture/photo images	
T	Color	Mono
90	62%	53%
95	66%	54%
100	69%	57%
105	65%	57%
110	66%	59%

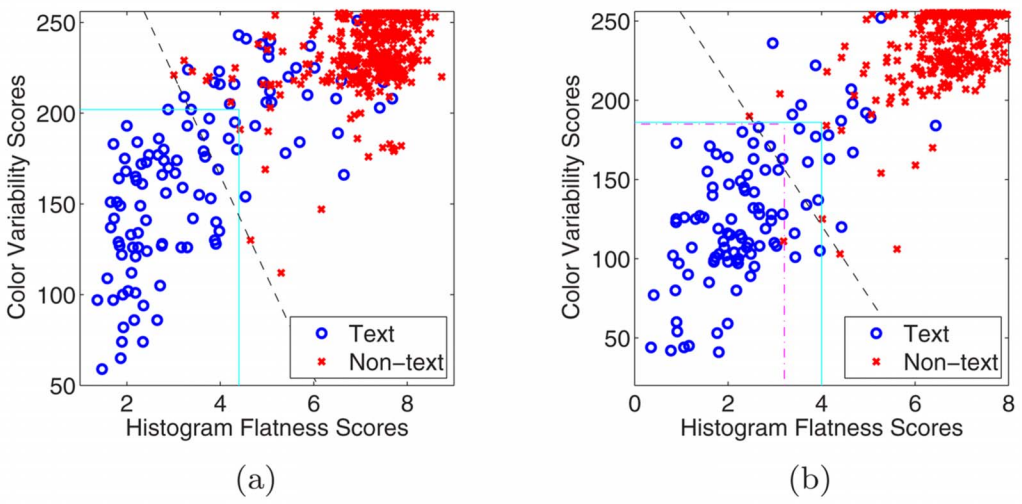


Fig. 7 Histogram flatness and color variability scores for (a) color images and (b) mono images. 111 color text, 359 color nontext, 102 mono text, and 319 mono nontext images are used in these experiments. (Color online only.)

following sequence of horizontally consecutive pixels should correspond to a single edge: $p_1=50$, $p_2=60$, $p_3=160$, and $p_4=170$. Yet, according to our definition, both (p_1,p_2,p_3) and (p_2,p_3,p_4) will get counted as text edges. According to our experiments, such multiple counting causes the classification performance to deteriorate. To avoid this, we use the following strategy. Let $p_1,p_2,\ldots,p_n$ be n consecutive pixels in either the horizontal or the vertical direction. We start from $i=1$ to check whether (p_i,p_{i+1},p_{i+2}) is a text edge. If it is a text edge, we skip $(p_{i+1},p_{i+2},p_{i+3})$ and $(p_{i+2},p_{i+3},p_{i+4})$, and check whether $(p_{i+3},p_{i+4},p_{i+5})$ is a text edge next. Otherwise, if (p_i,p_{i+1},p_{i+2}) is not a text edge, we increment i by one and check $(p_{i+1},p_{i+2},p_{i+3})$.

Let $E_R(b)$, $E_G(b)$, and $E_B(b)$ denote the numbers of the text edges inside a block b, for the red, green, and blue channels, respectively. The text edge count for block b is defined as the maximum of $E_R(b)$, $E_G(b)$, and $E_B(b)$. The text edge count for an image is then defined as the maximum text edge count among all the blocks. A large text edge count suggests that the image may be in the mix or text category. On the basis of some experiments (Table 10), we use 64×64 blocks.

In addition to the text edge counts, we reuse the 256-bin R, G, and B histograms of the means of 8×8 blocks which, as described in Section 3.2, have been computed for the text versus nontext classification task. As in Section 3.2, we extract the number of nonzero bins for each of the three histograms. In addition, we extract the k-spans for each of the three histograms, for three values of k: $k=M/8,M/4,M/2$, where M is the maximum of the histogram over its first ζ bin. (see Section 3.1 for the definition of a k-span). Note that the top bins of a block-mean histogram often have very large values due to large white background areas in many documents. In order to get useful color information, we clip these top bins when extracting k-spans. We use $\zeta=230$, based on experiments summarized in Table 11. These twelve features (the number of nonzero bins and the three k-spans for each of the three histograms)

Table 10 Percentage of correctly classified picture/photo images for different block sizes. Other parameters are the same as in the experiments of Fig. 9

	Percentage of correctly classified picture/photo images	
Block size	Color	Mono
8×8	33%	48%
16×16	45%	58%
32×32	61%	56%
64×64	69%	57%

Table 11 Percentage of correctly classified picture/photo images for different ζ. Other parameters are the same as in the experiments of Fig. 9

	Percentage of correctly Classified picture/photo images	
ζ	Color	Mono
220	67%	55%
225	68%	56%
230	69%	57%
235	67%	57%
240	64%	57%

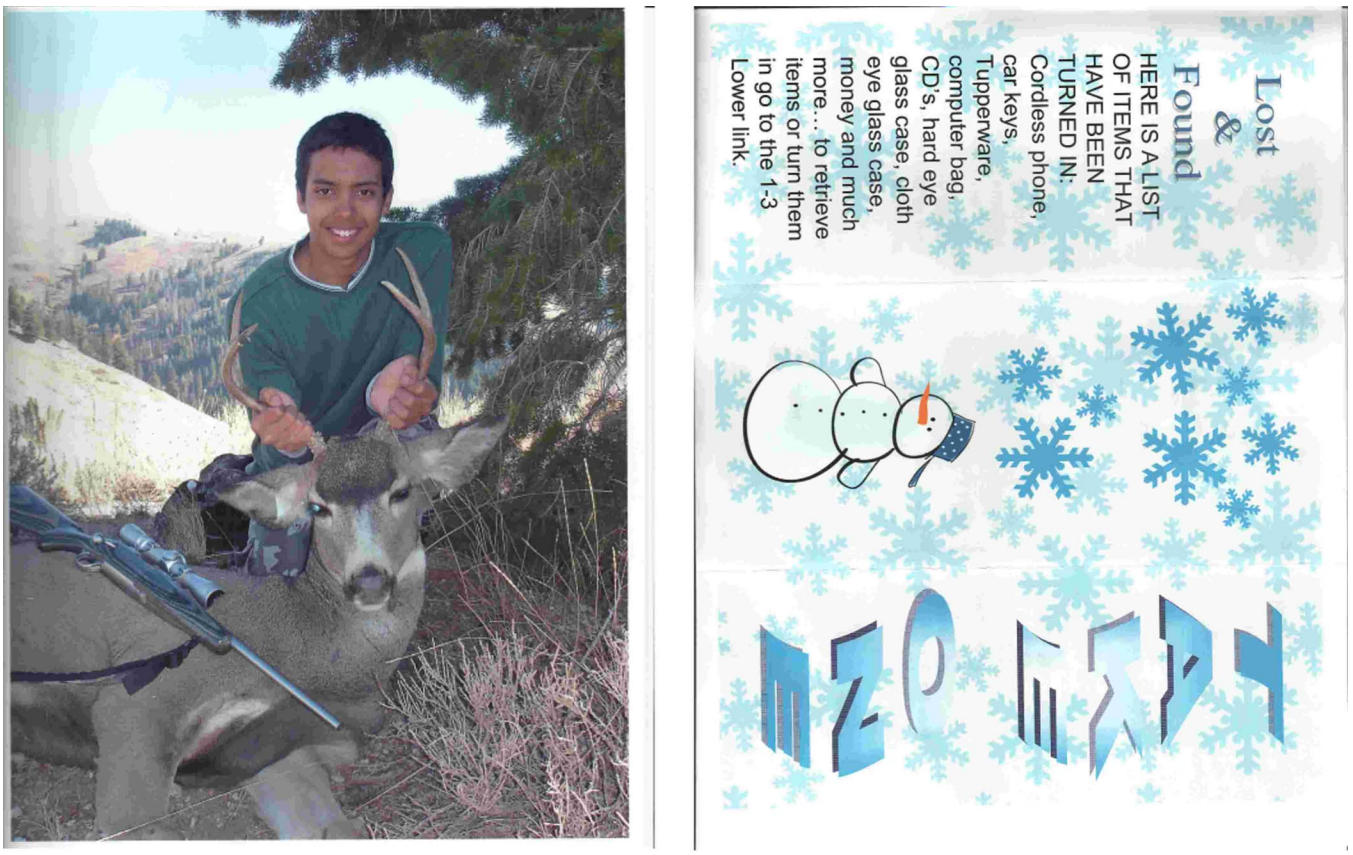


Fig. 8 (a) A picture image containing only a natural scene. (b) A mix image containing both text and non-natural scenes. (Color online only.)

form a feature vector for the image. We estimate the means of the feature vectors for the two classes from training data, and we also estimate a common covariance matrix. For any image whose feature vector is $\mathbf{y}$, we then define the unnaturalness score

$$U = (\hat{\mathbf{m}}_{\text{text/mix}} - \hat{\mathbf{m}}_{\text{pic/photo}})^T \hat{\Lambda}_U^{-1} \mathbf{y},$$

where $\hat{\mathbf{m}}_{\text{pic/photo}}$ and $\hat{\mathbf{m}}_{\text{text/mix}}$ are the estimated mean vectors for the two classes, and $\hat{\Lambda}_U$ is the estimated common covariance matrix. The larger the score U, the more likely it is that the image is not a photo or a picture. As in Section 3.2, this score may be obtained from a Gaussian likelihood ratio test.

Similar to Section 3.2, we train two different sets of $\hat{\mathbf{m}}_{\text{pic/photo}}$, $\hat{\mathbf{m}}_{\text{text/mix}}$, and $\hat{\Lambda}_U$ for color originals and mono originals. Figure 9(a) shows the scatter plot of the text edge counts and unnaturalness scores for all color originals in the data set. Blue O's represent picture/photo documents, and red X's represent text/mix documents. In order to classify an image, we train a linear classifier. Note that misclassifying most text and mix documents as picture/photo is harmful because copying mix and text originals in picture or photo mode results in poor quality, especially in reproducing text. On the other hand, mislabeling picture and photo documents as mix is acceptable. Therefore, we must select the classifiers conservatively so that as few as possible mix and text documents are misclassified as picture/photo. The dashed black line in Fig. 9(a) illustrates one of such linear classifiers. It correctly classifies 69% of color picture/photo documents without any misclassification of color text/mix documents. The scatter plot for the mono originals is shown in Fig. 9(b). The linear classifier corresponding to the dashed black line leads to 57% correct classification rate of mono picture/photo documents as well as one misclassification of mono text/mix documents.

We also try to use separate thresholds for the text edge count and the unnaturalness score to classify an image. The image is classified as photo or picture if both scores are less than their corresponding thresholds. Otherwise, it is classified as mix or text. The solid cyan line in Fig. 9(a) indicates the decision boundaries for color originals. Given no misclassification of color text/mix documents, the correct classification rate is 59% for color picture/photo documents, which is 10% lower than the result of the linear classifier. Similarly, the solid cyan line in Fig. 9(b) only gives 42% correctly classified mono picture/photo documents. The percentage is 15% lower than the rate obtained by the linear classifier, whereas in both cases there is only one misclassified mono text/mix documents.

We use three parameters in this subsection: block size for text edge counts, threshold T used in the definition of a

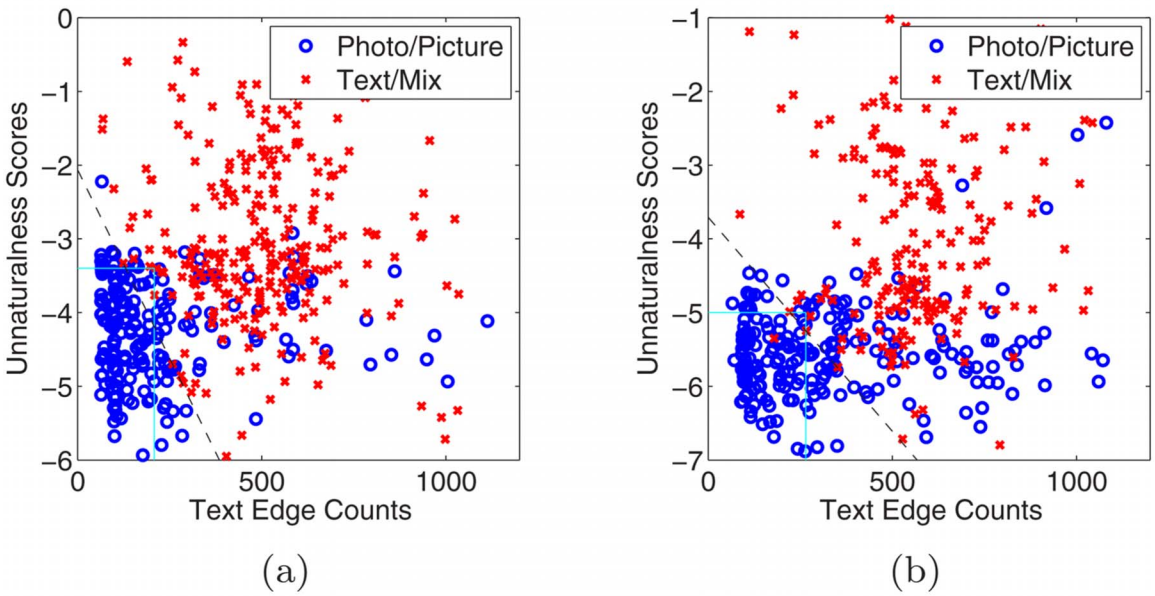


Fig. 9 Text edge counts and unnaturalness scores for (a) color images and (b) mono images; 200 color picture/photo images, 270 color text/mix images, 215 mono picture/photo images, 206 mono text/mix images are used in these experiments. (Color online only.)

text edge, and the clip point ζ for the block-mean histograms. On the basis of the experiments summarized in Tables 9–11, we use 64×64 blocks, $T=100$, and $\zeta=230$. These experiments also indicate that the algorithm is not very sensitive to these parameters: similar parameter sets lead to similar misclassification rates.

We compare our text edge detector to traditional edge detectors such as Sobel[28] and Laplacian[29] operators. Again, we divide an image into blocks and use Sobel or Laplacian operators to count the number of text edges inside each block. The text edge count of the entire image is the maximum count of text edges over all blocks. For the Sobel operator, a 3×3 horizontal filter F_H and a 3×3 vertical filter F_V are used where

$$F_H=\begin{bmatrix}+1 & 0 & -1\\ +2 & 0 & -2\\ +1 & 0 & -1\end{bmatrix} \text{ and } F_V=\begin{bmatrix}+1 & +2 & +1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}.$$

The filter responses of F_H and F_V are denoted by G_H and G_V, respectively. At each pixel, if $|G_H|$ is larger than a threshold T_s, we increment the text edge count of the corresponding block by one. Similarly, we increment the text edge count by one if $|G_V|>T_s$. For Laplacian operator, a 3×3 filter F_L is used and

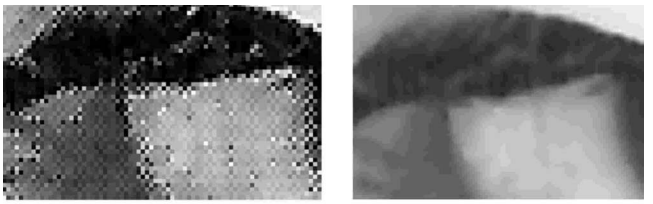
$$F_L=\begin{bmatrix}-1 & -1 & -1\\ -1 & +8 & -1\\ -1 & -1 & -1\end{bmatrix}.$$

The filter response of F_L is denoted by G_L. At each pixel, we increment the text edge count of the corresponding block by one if G_L at the current pixel has an sign opposite to G_L at any neighboring pixels in a eight-point neighborhood, and the range of pixel values in the eight-point neighborhood is larger than a threshold T_L. After getting the text edge count of an image, we combine it with the unnaturalness score discussed previously to make a classification decision. We try various block sizes, T_s, and T_L. We also try both rectanglar classifier and linear classifier mentioned previously. The best classification results for Sobel operator are obtained when block size is 64×64 and $T_s=500$ while the best results for Laplacian operator are obtained when block size is 64×64 and $T_L=100$. The results are listed in Table 12. Our text edge detector not only results in better percentages of correct classifications, but also requires less computation (eight additions/comparisons per pixel for our detector, 14 additions/comparisons per pixel for Sobel operator, and 31 additions/comparisons per pixel for Laplacian operator).

As with color-mono and text-nontext classification algorithms, there are existing methods that could improve upon our classification accuracy if we did not have severe com-

Table 12 Percentage of correctly classified picture/photo images for different methods. 200 color picture/photo images, 270 color text/mix images, 215 mono picture/photo images, 206 mono text/mix images are used in these experiments.

	Percentage of correctly classified images	
Methods	Color	Mono
Ours	69%	57%
Sobel	41%	53%
Laplacian	53%	52%



(a) (b)

Fig. 10 (a) Image patch from the scanned image of a picture document. (b) Image patch from the scanned image of a photo document.

putational complexity and memory limitations. In Ref. 30, certain features are calculated using 2-D histograms constructed from pixel values. These features are used to classify an image as either a natural picture or a synthetic graphic. This technique is more sophisticated than our unnaturalness score. However, 2-D histograms require a large amount of memory, and the features require significant computation power. As a result, this method cannot be implemented in our low-end copy pipeline.

3.4 *Picture Versus Photo Classifier*

In order to illustrate the difference between picture documents and photo documents, we generate two documents of the same content: one is developed using the continuous tone technology. and the other is developed using the halftone technology. Figure 10 shows image patches from the scanned images of these two documents. As shown in Fig. 10, pictures typically contain halftone noise while photos do not. We also observe that smooth regions that are near midtone are most affected by the halftone noise. Therefore, we use these regions to distinguish between a picture and a photo.

We partition an image into 8×8 blocks and measure each block b's noise level in the red, green, and blue channels. We define a block b's roughness $\gamma_R(b)$ in the red channel as follows:

$$\gamma_R(b)=\begin{cases}\displaystyle\sum_{\substack{i,j:\ i \text{ and } j\\ \text{are neighbors}}} |R(i)-R(j)| & \text{if } |\bar{R}(b)-128|<\phi\\ \infty & \text{otherwise}\end{cases},$$

where the summation is performed over all the pairs (i,j) of neighboring pixels inside block b in both horizontal and vertical directions, $R(i)$ and $R(j)$ are the red intensities of pixels i and j, $\bar{R}(b)$ is the average red intensity of all pixels inside the block b, and ϕ is a positive threshold. We define the block b's roughness $\gamma_G(b)$ in the green channel and $\gamma_B(b)$ in the blue channel in a similar way. The basic idea here is to first identify whether a block is near midtone by calculating the distance between its mean value and 128, then calculate the block's roughness if it is actually near midtone. The roughness $\gamma(b)$ of block b is the minimum of $\gamma_R(b)$, $\gamma_G(b)$, and $\gamma_B(b)$. We define the roughness of the image γ_{image} as the minimum $\gamma(b)$ over all its blocks b. We use $\phi=48$; however, our experiments indicate that γ_{image} is very robust with respect to different choices of ϕ (see Table 13).

Table 13 Percentage of correctly classified picture and photo images for different ϕ. There is always only one picture image misclassified as photo and no photos misclassified as pictures in these experiments. These experiments use 221 pictures and 200 photos.

	Percentage of correctly classified images	
ϕ	Photo	Picture
40	89%	88%
44	89%	88%
48	89%	88%
52	89%	88%
56	89%	87%

Note that copying photo originals in the picture mode or copying picture originals in the photo mode are both considered harmful, but copying both types of originals in the mix mode is acceptable. Therefore, we use a scheme involving two thresholds and let the hard-to-classify photos and pictures be labeled as "mix." Given two thresholds T_1 and T_2 with $T_1<T_2$, an image is classified as photo if its roughness is below T_1, classified as picture if its roughness is larger than T_2, and classified as mix if its roughness is between T_1 and T_2. Both T_1 and T_2 are determined from training data. When $T_1=105$ and $T_2=161$, 88% of picture images in the training suite are correctly classified as "picture," 11.5% are mislabeled as mix and 0.5% (only one image) is misclassified as photo. Among all photo images, 89% are correctly classified as photo and the rest are misclassified as mix.

We compare our method with the peak-detection-based halftone detection techniques.[31–35] Specifically, we compare to a method found in Refs. 31 and 33–35, which considers a pixel to be a peak if it is the local maximum in the eight-point neighborhood, and if the ranges of these pixel values in the vertical, horizontal, and diagonal directions are all greater than a threshold T_p. To perform image classification using this method, we divide the image into blocks and count the number of peaks for each block. We call the maximal count over all the blocks the peak count for the image. We then classify the image as mix, photo, or picture by thresholding the peak count. We have tried various T_p: 10, 20, 30, 40, and 50. We have also tried various block sizes: 8×8, 16×16, 32×32, and 64×64. The best percentages of correctly classified pictures and photos are obtained when $T_p=40$ and the block size is 64×64, and these numbers are listed in Table 14. Our method results in slightly higher classification accuracy and also requires less computation (four additions/comparisons per pixel for our method and 28 additions/comparisons per pixel for the other method). Autocorrelation-based halftone detection methods[36,37] are generally good for detecting periodic halftone, but may not be appropriate for stochastic halftone.

Table 14 Percentage of correctly classified picture and photo images for different methods. A block size of 8×8 is used in our method and a block size of 64×64 is used in the peak count method. These experiments use 221 pictures and 200 photos.

	Percentage of correctly classified images	
Methods	Photo	Picture
Ours	89%	88%
Peak count	79%	88%

Moreover, depending on window sizes used, autocorrelation-based methods may require much more computation than our method.

4 Experimental Results for the Overall RGB Algorithm

As mentioned in Section 3.1, we have a data set containing 891 images with at least 100 images for each class. The ground-truth labels of all images in the data set are selected by hand. We do the following cross-validation test. The images in each class are randomly divided into two groups of equal sizes: Group I and Group II. All images in group I of all eight classes form the training set; the remaining images form the testing set. All decision thresholds and coefficients for calculating histogram flatness scores and unnaturalness scores are obtained from the training set.

In particular, we use an automated method to select decision boundaries for the text versus nontext and text/mix versus picture/photo classifiers. We select the decision boundary that minimizes the following criterion:

$$(\text{number of benign errors}) + \alpha(\text{number of harmful errors}),$$

where α is a number indicating how many benign errors we are willing to trade for a single harmful error.

We follow the treelike decision structure described in Section 2 and let all the images in the testing set run through the classifiers discussed in Section 3. Finally, we swap the training and testing sets and repeat the previous experiment. Note that we perform the cross-validation procedure once (i.e., for a fixed group I and a fixed group II). The running time for each image including feature extraction and classification is approximately 4.28 s on a Pentium IV 3.2 GHz desktop computer, and the running time in the actual hardware implementation will be much faster.

Table 15 summarizes the combined classification results for all eight modes. In our experiments, $\alpha=10$. It contains the empirical conditional probabilities $P(\text{classification result}\,|\,\text{ground truth})$ for the whole data set. Depending on the ground truth, accuracy ranges from 38% to 96%. These numbers are shown in the main diagonal of Table 16. We consider the accuracy to be satisfactory because very few images fall into harmful modes. These harmful misclassifications are indicated in boldface numbers in Table 15. All other misclassifications are considered benign.

5 NIQ Algorithm

5.1 *NIQ Color Space*

When image data are represented in the *RGB* color space, some feature extraction procedures are repeated for the red, green, and blue channels. The repeated procedures include building local histograms and computing the histogram spans, counting the number of text edges, building the histograms of the block means, and calculating the roughness for the near-midtone blocks. This fact suggests that an al-

Table 15 Combined testing classification rates for the cross-validation when the image data are in the RGB space. Harmful misclassifications are indicated in boldface. Here we choose $\alpha=10$.

	Classification results							
Ground truth	Color text	Color mix	Color picture	Color photo	Mono text	Mono mix	Mono picture	Mono photo
Color text	61%	37%	**1%**	**1%**	—	—	—	—
Color mix	—	96%	**3%**	**1%**	—	—	—	—
Color picture	—	61%	38%	**1%**	—	—	—	—
Color photo	—	24%	—	75%	—	**1%**	—	—
Mono text	14%	4%	—	—	63%	18%	—	—
Mono mix	—	8%	—	—	**2%**	85%	**4%**	—
Mono picture	—	4%	—	—	—	58%	38%	—
Mono photo	—	—	—	—	—	31%	**1%**	68%

Table 16 Percentage of correctly classified mono images in the training set, under maximum threshold guaranteeing 100% correct classification of color images, for various block sizes when image data are represented in the NIQ color space. 470 color images and 421 mono images are used in these experiments.

Block size	Percentage of correctly classified mono images
8×8	66%
16×16	89%
32×32	88%
64×64	75%

gorithm requiring even less computational complexity may be achieved if image data are represented in an opponent color space.

The specific opponent color space we use is called the *NIQ* space, where *N* is the luminance channel, *I* is the red-green color channel, and *Q* is the yellow-blue color channel. The transformation from the *RGB* color space to the *NIQ* color space is as follows:

$$\begin{bmatrix} N \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/2 & -1/2 & 0 \\ 1/4 & 1/4 & -1/2 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}. \quad (2)$$

Note that *N*, *I*, and *Q* are obtained through a linear transformation of *R*, *G*, and *B*. We add a constant 128 when calculating *I* and *Q* values so that *I* and *Q* range from 0 to 255.

5.2 *Mono Versus Color Classifier*

Our mono versus color classifier in the *RGB* space measures each block's distance to gray. We construct an analogous classifier in the *NIQ* space. A pixel is gray if and only if both its *I* and *Q* values are equal to 128. This can be easily verified using Eq. (2). For a pixel *p* with the intensities $N(p)$, $I(p)$, and $Q(p)$, we define its colorfulness (or its distance to gray) $C_{NIQ}(p)$ as follows:

$$C_{NIQ}(p) = |I(p) - 128| + |Q(p) - 128|.$$

Note that this equation is not equivalent to Eq. (1). We follow a procedure similar to that described in Section 3.1. We divide the image into blocks, calculate a block's colorfulness as the sum of $C_{NIQ}(p)$ of all the pixels *p* inside the block and let the colorfulness $C_{NIQ,\text{image}}$ of the image be the maximum colorfulness among all its blocks. We classify the image as color if $C_{NIQ,\text{image}}$ is larger than a threshold, and as mono otherwise. As mentioned in Section 3.1, we choose the thresholds conservatively so that all color images in the training suite are correctly identified as color. Among several block sizes that we have tried, 16×16 blocks produce the best result, as shown in Table 16.

5.3 *Other Classifiers*

We use the luminance channel *N* to do the three remaining classification tasks (i.e., text versus nontext, text/mix versus picture/photo, and picture versus photo. We apply the algorithms described in Sections 3.2–3.4, to the *N* channel. For the text versus nontext classifier, 64-bin local histograms are built for the *N* channel only, and the color variability score is the number of nonzero bins in the histogram of block means for the *N* channel. We choose 64×64 as the block size for the local histograms and $T=100$. For the text/mix versus picture/photo classifier, we count the text edges for the *N* channel only. The unnaturalness score is

Table 17 Combined testing classification rates for the cross-validation when the image data is in the NIQ space. Harmful misclassifications are indicated in boldface. Here we choose $\alpha=10$.

	Classification results							
Ground truth	Color text	Color mix	Color picture	Color photo	Mono text	Mono mix	Mono picture	Mono photo
Color text	60%	40%	—	—	—	—	—	—
Color mix	**1%**	98%	**1%**	—	—	—	—	—
Color picture	—	58%	38%	**3%**	—	—	**1%**	—
Color photo	—	36%	—	64%	—	—	—	—
Mono text	14%	6%	—	—	65%	15%	—	—
Mono mix	—	5%	—	—	**1%**	89%	**5%**	—
Mono picture	—	6%	1%	—	—	63%	30%	—
Mono photo	—	5%	—	1%	—	26%	**2%**	66%

calculated based on a four-dimensional feature vector which consists of the number of nonzero bins and the histogram spans for three different values k, extracted from the histogram of block means, as described in Section 3.3. We use 64×64 blocks for the text edge count. For the picture versus photo classifier, we use the average N value of a block to determine whether or not the block is near midtone. The roughness is measured in the N channel only. We choose $\phi=48$. Experiments similar to those in Section 3 have been done to guide our selection of parameter values for the *NIQ* algorithms.

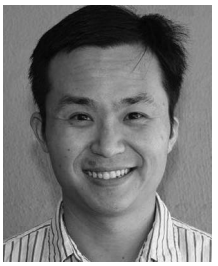
As in Section 4, we randomly divide all images into two subsets of equal size. We first use one subset as the training set and the other as the testing set; then swap the training and testing sets and repeat the experiment. Again, we are willing to trade 10 benign errors for a single harmful error for the text versus nontext and text/mix versus picture/photo classifiers. The combined testing results are displayed in Table 17. The numbers in Table 17 are the average empirical conditional probabilities $P(\text{classification result}|\text{ground truth})$ over two testing sets. Comparing these results to those in Table 15, we observe that performance drops in the classifications of color originals. This reduction in performance is predictable since we only use the N channel information to decide among the text, mix, photo, and picture classes. However, the running time of the *NIQ* algorithm is approximately 1.60 s per image on the same Pentium IV 3.2 GHz desktop. It is slightly higher than one-third of the running time of the *RGB* algorithm.

6 Conclusions

We have introduced real-time, low-complexity document page classification algorithms to be used for mode selection in a low-end copy pipeline. The revised pipeline improves copy quality, simplifies user interactions, and increases the copy rate. The classification algorithms analyze the scanned image and classify it into one of eight modes in the copy pipeline. Modes are the combinations of mono/color and text/mix/picture/photo. When the data are represented in the *RGB* color space, the mode classification is 38% to 96% accurate with misclassifications tending toward benign modes. We have also introduced an alternative algorithm to classify the images represented in the *NIQ* color space. Although its classification performance for color originals is slightly worse than that of the *RGB* algorithm, the *NIQ* algorithm is about three times less computationally complex than the *RGB* algorithm.

References

1. H. Cheng and C. A. Bouman, "Document compression using rate-distortion optimized segmentation," *J. Electron. Imaging*, **10**(2), 460–474 (2001).
2. R. de Queiroz, "Compression of compound documents," in *Proc. ICIP*, Vol., 1, pp. 209–213, Kobe, Japan (1999).
3. S. Revankar and Z. Fan, "Picture, graphics and text classification of document image regions," *Proc. SPIE*, 4300, 224–228 (2001).
4. S. Simske and S. Baggs, "Digital capture for automated scanner workflows," *Proc. 2004 ACM Symposium on Document Engineering*, pp. 171–177, Milwaukee (2004).
5. W. Wang, I. Pollak, T.-S. Wong, C. A. Bouman, M. P. Harper, and J. M. Siskind, "Hierarchical stochastic image grammars for classification and segmentation," *IEEE Trans. Image Process.*, **15**(10), 3033–3052 (2006).
6. Z. Fan, Y. Zhang, and M. E. Banton, "Multi-resolution neutral color detection," U.S. Patent No. 6249592 (2001).
7. K. Hara, "Image processing apparatus and method, and image processing system," U.S. Patent No. 6804033 (2004).
8. Y. Hirota, K. Toyama, and T. Nabeshima, "Image forming apparatus for distinguishing between types of color and monochromatic documents," U.S. Patent No. 6118895 (2000).
9. Y. Hirota, K. Toyama, S. Imaizumi, H. Hashimoto, and K. Ishiguro, "Image processing apparatus, image forming apparatus and color image determination method thereof," U.S. Patent No. 7177462 (2007).
10. Y. Hirota, K. Toyama, S. Imaizumi, H. Hashimoto, and K. Ishiguro, "Image processing apparatus, image forming apparatus and color image determination method thereof," U.S. Patent No. 7319786 (2008).
11. W.-H. Hsieh, C.-H. Shih, and I.-C. Teng, "Method of determining color composition of an image," U.S. Patent No. 7308137 (2007).
12. K. Kanamori, "Image processing method, Image processing apparatus and image forming apparatus," U.S. Patent No. 6643397 (2003).
13. H. Kawano, "Color type determining device," U.S. Patent No. 6256112 (2001).
14. H. Kawano and H. Yamamoto, "Image discriminating apparatus," U.S. Patent No. 6240203 (2001).
15. H. Koizumi, K. Kouno, Y. Sorimachi, Y. Suzuki, and Y. Awata, "Image recognition apparatus for judging between monochromic and color originals," U.S. Patent No. 5287204 (1994).
16. K. Murai, N. Kasahara, and K. Hashimoto, "Color image processing apparatus," U.S. Patent No. 5032904 (1991).
17. K. Nagata, "Device for judging the type of color of a document," U.S. Patent No. 5282026 (1994).
18. J. Shishizuka, "Method and apparatus for processing image," U.S. Patent No. 5786906 (1998).
19. H. Tanaka, "Image determining apparatus capable of properly determining image and image forming apparatus utilizing the same," U.S. Patent No. 6900902 (2005).
20. K. Yamamoto, Y. Matsui, and H. Matsuo, "Image processing apparatus," U.S. Patent No. 5734758 (1998).
21. J. Bares, and T. W. Jacobs, "Detecting process neutral colors," U.S. Patent No. 6972866 (2005).
22. J. C. Handley and Y.-W. Lin, "Neutral pixel detection using color space feature vectors wherein one color space coordinate represents lightness," U.S. Patent No. 7116443 (2006).
23. D. Horie, and N. Okisu, "Color discrimination apparatus and method," U.S. Patent No. 6480624 (2002).
24. M. R. Grosso and C. D. Woodward, "Automatic detection of black and white pages in a color document stream," U.S. Patent No. 6718878 (2004).
25. H. Kanno and T. Sawada. "Color image-forming apparatus capable of discriminating the colors of the original image," U.S. Patent No. 6504628 (2003).
26. H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part I.*, Wiley, Hoboker, NJ (2001).
27. R. de Queiroz, R. R. Buckley, and M. Xu, "Mixed raster content (MRC) model for compound image compression," *Proc. SPIE* 3653, p. 1106–1117 (1999).
28. I. Sobel and G. Feldman, "A 3×3 isotropic gradient operator for image processing," presented at the Stanford Artificial Project (1968).
29. R. Jain, R. Kasturi, and B. G. Schuck, *Machine Vision*, McGraw Hill, New York (1995).
30. Z. Fan, "Image type classification using color discreteness features," U.S. Patent No. 6996277 (2006).
31. R. J. Clark, L. C. Williams, S. A. Schweid, and J.-N. Shiau, "Method and system for classifying and processing of pixels of image data," U.S. Patent No. 6181829 (2001).
32. X. Li, M. E. Meyers, and F. K. Tse, "Image segmentation apparatus and method," U.S. Patent No. 6389164 (2002).
33. S. A. Schweid and J.-N. Shiau, "Method and system for classifying and processing of pixels of image data," U.S. Patent No. 6549658 (2003).
34. J.-N. Shiau, "Method and system for classifying and processing of pixels of image data," U.S. Patent No. 6185328 (2001).
35. L. C. Williams, R. J. Clark, and J.-N. Shiau, "Method and system for classifying and processing of pixels of image data," U.S. Patent No. 6229923 (2001).
36. Y.-W. Lin and A. F. Calarco, "Image processing apparatus using approximate auto correlation function to detect the frequency of halftone image data," U.S. Patent No. 4811115 (1989).
37. J.-N. Shiau and Y.-W. Lin, "Binary halftone detection," U.S. Patent No. 7239430 (2007).



Xiaogang Dong received his BS in electronic engineering from Shanghai Jiao Tong University, China, in 2000, MS in applied mathematics, and PhD in electrical and computer engineering from Purdue University, West Lafayette, IN, in 2002 and 2007, respectively. He is a senior research engineer with Media Processing Technology Laboratory, Sony Electronics Inc. His research interests are in image and signal processing, specifically image noise reduction, segmentation, and document imaging.

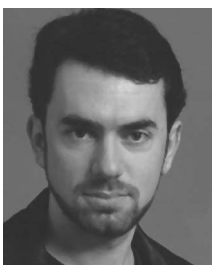
Kai-Lung Hua received his BS in electrical engineering from National Tsing Hua University, Taiwan, in 2000, and MS in communication engineering from National Chiao Tung University, Taiwan, in 2002. He is currently pursuing a PhD at the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN. His research interests include signal, image, and video processing.

Peter Majewicz has a BS in computer science mathematics from the State University of New York-Geneseo, and a MS in imaging science from Rochester Institute of Technology. He currently works as a color and imaging scientist for Hewlett-Packard.

Gordon McNutt has a BS and MS in computer science from Boise State University. He currently works as an embedded firmware developer for Cradlepoint, Inc., headquartered in Boise, Idaho.

Charles A. Bouman received a BSEE from the University of Pennsylvania in 1981 and MS 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 his PhD in electrical engineering from Princeton University. In 1989, he joined the faculty of Purdue University, where he holds the rank of professor with a primary appointment in the School of Electrical and Computer Engineering and a secondary appointment in the School of Biomedical Engineering. Currently, he is codirector of Purdues Magnetic Resonance Imaging Facility located in Purdues Research Park. 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. He is a Fellow of the IEEE, the American Institute for Medical and Biological Engineering (AIMBE). The society for Imaging Science and Technology (IS&T), and the SPIE professional society, and a recipient of IS&Ts Raymond C. Bowman Award for outstanding contributions to digital imaging education and research, and a University Faculty Scholar of Purdue University. He is currently the Editor-in-Chief Elect for the *IEEE Transactions on Image Processing*, a member of the IEEE Biomedical Image and Signal Processing Technical Committee, and a member of the Steering Committee for the *IEEE Transactions on Medical Imaging*. 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 cochair of the 2006 SPIE/IS&T Symposium on Electronic Imaging, Cochair 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; he is the founder and cochair of the SPIE/IS&T conference on computational imaging.

Jan P. Allebach received his BSEE from the University of Delaware in 1972 and his PhD from Princeton University in 1976. He was on the faculty at the University of Delaware from 1976 to 1983. Since 1983, he has been at Purdue University where he is Hewlett-Packard Professor of Electrical and Computer Engineering. His current research interests include image rendering, image quality, color imaging and color measurement, printer and sensor forensics, and digital publishing. Professor Allebach is a fellow of the IEEE, the Society for Imaging Science and Technology (IS&T), and SPIE. He has been especially active with the IEEE Signal Processing Society and IS&T. He has served as Distinguished/Visiting Lecturer for both societies, and has served as an officer and on the Board of Directors of both societies. He is a past associate editor of the *IEEE Transactions on Signal Processing* and *IEEE Transactions on Image Processing* and is presently editor of the *IS&T/SPIE Journal of Electronic Imaging*. He received the Senior (best paper) Award from the IEEE Signal Processing Society. In 2004, he was named Electronic Imaging Scientist of the Year by IS&T and SPIE. In 2007, he was named Honorary Member of IS&T, the highest award that IS&T bestows. He is the recipient of four teaching awards from Purdue University.



Ilya Pollak received his BS and MEng in 1995 and PhD in 1999, all in electrical engineering from the Massachusetts Institute of Technology, Cambridge. In 1999 to 2000, he was a postdoctoral researcher at the Division of Applied Mathematics, Brown University, Providence, Rhode Island since 2000, he has been with Purdue University, West Lafayette, Indiana, where he is currently associate professor of electrical and computer engineering. He has held visiting positions at the Institut National de Recherche en Informatique et Automatique in Sophia Antipolis, France, at Tampere University of Technology, Finland, and at Jefferies, Inc., New York. His research interests are in image and signal processing, specifically, hierarchical statistical models, fast estimation algorithms, nonlinear scale-spaces, adaptive representations, with applications to image and video compression, segmentation, classification, and financial time series analysis. Prof. Pollak received a CAREER award from the National Science Foundation in 2001, the Eta Kappa Nu Outstanding Faculty Award in 2002 and 2007, and the Chicago-Area Alumni Young Faculty Award in 2003. He is an associate editor of the *IEEE Transactions on Image Processing* and a member of the IEEE Signal Processing Society's Technical Committee on Signal Processing Theory and Methods. He is a Cochair of the SPIE/IS&T Conference on Computational Imaging.