Training-Based Descreening

Hasib Siddiqui and Charles A. Bouman, *Fellow, IEEE*

***Abstract*—Conventional halftoning methods employed in electrophotographic printers tend to produce Moiré artifacts when used for printing images scanned from printed material, such as books and magazines. We present a novel approach for descreening color scanned documents aimed at providing an efficient solution to the Moiré problem in practical imaging devices, including copiers and multifunction printers. The algorithm works by combining two nonlinear image-processing techniques, resolution synthesis-based denoising (RSD), and modified smallest univalue segment assimilating nucleus (SUSAN) filtering. The RSD predictor is based on a stochastic image model whose parameters are optimized beforehand in a separate training procedure. Using the optimized parameters, RSD classifies the local window around the current pixel in the scanned image and applies filters optimized for the selected classes. The output of the RSD predictor is treated as a first-order estimate to the descreened image. The modified SUSAN filter uses the output of RSD for performing an edge-preserving smoothing on the raw scanned data and produces the final output of the descreening algorithm. Our method does not require any knowledge of the screening method, such as the screen frequency or dither matrix coefficients, that produced the printed original. The proposed scheme not only suppresses the Moiré artifacts, but, in addition, can be trained with intrinsic sharpening for deblurring scanned documents. Finally, once optimized for a periodic clustered-dot halftoning method, the same algorithm can be used to inverse halftone scanned images containing stochastic error diffusion halftone noise.**

***Index Terms*—Descreening, halftone, Moiré artifacts, resolution synthesis, smallest univalue segment assimilating nucleus (SUSAN) filter.**

I. INTRODUCTION

MOST of the available printing technologies cannot produce continuous tones. Instead the images printed with these devices contain a series of dots arranged in specific patterns to simulate different shades of gray. The process of converting a continuous-tone image into a binary image that can be rendered using a bi-level printing device is called halftoning.

Conventional halftoning methods involve either changing the size of printed dots, called amplitude modulation (AM), or changing the relative density of dots on the page, called frequency modulation (FM). The most common method for performing AM halftoning is clustered dot screening [29] while commonly used FM halftoning algorithms are dispersed dot screening [6], [27], [30] and error diffusion [11], [17], [32].

AM halftoning methods offer the advantage of stable dot formation and, therefore, are widely used in electrophotographic and commercial offset printers where a single isolated dot may not develop and be stable. Essentially all printed material in the form of books, newspapers and magazines employs clustered dot halftoning. However, there is one serious drawback to using AM methods. The regular placement of dots in clustered dot halftoning can lead to Moiré patterns when it is used for reprinting images scanned from printed originals [1]. The term Moiré, in general, is used to describe interference patterns that arise due to the frequency difference or misalignment of two similar overlapping periodic structures. In a scan-to-print process, the halftone screens in the printed original and target printer are the two overlapping frequencies that could interfere and render the resulting print useless due to severe Moiré artifacts. In general, FM halftoning methods do not cause severe Moiré [19], however, the tiny isolated dots produced by these methods cannot be stably printed by electrophotographic printers. Thus, FM halftoning is mostly limited to ink jet printers.

A variety of different approaches have been suggested in the literature to avoid the Moiré patterns when printing scanned originals. The proposed techniques fall into two broad categories. The first approach is to improve upon the image rendering algorithm used in the printer for printing scanned halftones [8], [13], [18], [21]. One example of such a technique is the AM/FM halftoning algorithm developed in [13]. AM/FM works by simultaneously modulating the dot size and dot density to produce the best print quality at each gray level. The formation of larger dot clusters in AM/FM offers improved dot stability over traditional FM methods, while the irregular placement of dots leads to better Moiré suppression than if AM methods were used alone.

The second approach to coping with the Moiré problem is to process the scanned image for removal of screen frequencies before it is actually rehalftoned in the print engine. Perhaps, the simplest way of suppressing halftone noise is to smooth the entire scanned document with a single low-pass filter; however, this approach also softens the image edges [34].

Currently, there exist many sophisticated inverse halftoning algorithms aimed at reconstructing continuous-tone images from their halftone versions while protecting the detail and edge information in the image. Methods developed in [2], [10], [14], [28], and [31], including the projection scheme in [34], have been shown to produce high-quality results with binary halftones. Most of these procedures are iterative, and all assume full knowledge of the halftoning kernel, which is not reasonable when descreening scanned images printed through unknown printing processes. We note that while there do exist techniques for estimating the dither matrix thresholds [2], [10] and error diffusion weights [34] from binary halftones, these methods do not work effectively with grayscale (or color) scanned halftones. Lookup table [23] and hybrid LMS-MMSE [7]

Manuscript received April 3, 2006; revised October 30, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Reiner Eschbach.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: hsiddiqu@purdue.edu; bouman@ecn.purdue.edu).

Digital Object Identifier 10.1109/TIP.2006.888356

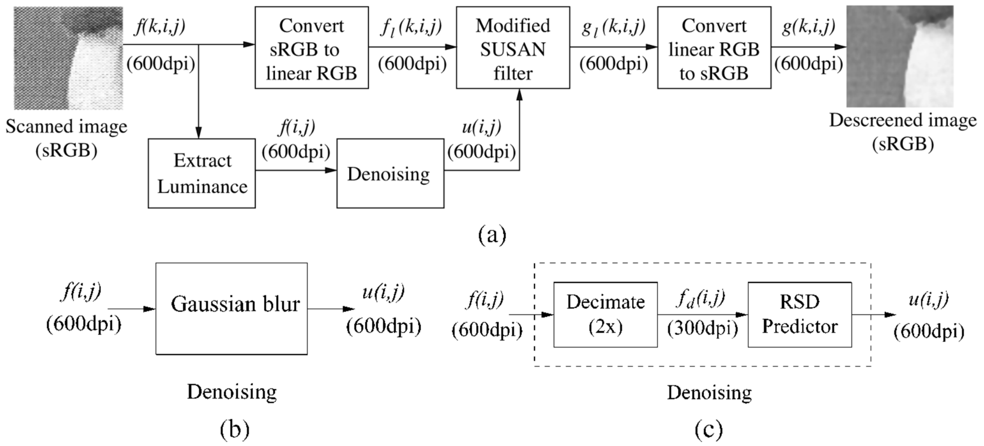


Fig. 1. Architecture of the descreening algorithm for processing color scanned halftones. (a) Block diagram. (b) Denoising accomplished through Gaussian blur. (c) Denoising accomplished through training-based RSD.

inverse halftoning algorithms are fast, and they do not assume explicit knowledge of the halftoning kernel; however, these methods are also intended for processing only binary halftones. Among the wavelet-based approaches, important contributions include [22], [24], and [38]. Some of the existing algorithms that can be used to provide efficient descreening solutions in low-cost implementations are gradient-based adaptive filtering [16], blind inverse halftoning [33], and segmentation-based descreening [15].

In this paper, we present a new descreening algorithm targeted for scan-to-print pipeline in a multifunction printer (MFP). The algorithm effectively removes a wide range of Moiré causing screen frequencies, as well as error diffusion halftone noise, from a scanned document while maintaining the overall image quality in terms of edge sharpness and detail preservation. In fact, the proposed technique can be trained with built-in sharpening to enhance the text quality and edge details.

The algorithm is based on two major components: resolution synthesis-based denoising (RSD), similar to the RS method developed by Atkins *et al.* [4], [5] for image scaling, and a modified version of a noise removal technique, the smallest univalue segment assimilating nucleus (SUSAN) filter [26]. RSD is based on a stochastic image model whose parameters are optimized in an offline training process. The training data for optimizing the stochastic model parameters is generated using print-to-scan methods from real scanners and printers, and is thus representative of the nonidealities found in a practical imaging device. The training method provides a framework where the training data obtained from an imaging platform is used to automatically optimize the descreening parameters with respect to that specific hardware.

In principle, RSD can be compared with the Kite's method [16] that works by adaptively constructing smoothing filters based on the local gradient information extracted using four different masks. In RSD, we use a larger (8-D) local feature vector, which includes higher order information in addition to the first order gradients, based on which optimal filters are chosen to process the local window.

The modified SUSAN filter is a nonlinear filter that reduces the halftone noise in smooth regions of the scanned image while preserving the edges and nonhalftoned image detail.

The remainder of this paper is organized as follows. In Section II, we explain in detail our proposed scheme for descreening color scanned halftones. Section III contains the experimental results, while the concluding remarks are presented in Section IV.

II. DESCREENING COLOR SCANNED HALFTONES

Fig. 1(a) shows the basic structure of our scheme. The notational convention uses two indices (i,j) to index gray scale images and three indices (k,i,j) to index RGB color images. We shall use $k = 0,1$, and 2 to represent the R, G, and B color channels, respectively. The input image $f(k,i,j)$ is an sRGB color image [3] that is obtained by scanning a printed color halftone image using a calibrated scanner. The scanner resolution is considered to be 600 dpi. The first step in the descreening process is to obtain the grayscale image $f(i,j)$ from the color scanned image. The grayscale image is computed using the relation: $f(i,j) = 0.30f(0,i,j) + 0.59f(1,i,j) + 0.11f(2,i,j)$. The grayscale image is then processed through the denoising module in order to reduce the effects of the periodic halftone noise. The output of the denoising module $u(i,j)$ is treated as a first order estimate to the reconstructed contone image.

The denoising could be accomplished with as simple a technique as a conventional Gaussian blur as shown in Fig. 1(b); however, in this paper, we develop a sophisticated statistical procedure, RSD, that significantly improves the descreening quality by preserving high-frequency image details. As shown in Fig. 1(c), the grayscale image $f(i,j)$ is decimated to $f_d(i,j)$ before it is input to the RSD predictor. Decimation blends the large halftone dots in the 600-dpi image resulting in a reduced screen period. The value of this is explained in Section II-A3. The task of RSD is two fold. First, it must suppress the residual screen patterns in the decimated image without further deteriorating the edge details. Second, it should scale up the 300-dpi image to the original scanner resolution of 600 dpi. RSD performs spatial processing that depends on the local content in the input image. The strength of RSD comes from the fact that, even in the presence of strong halftone noise, it can accurately distinguish whether the local window forms part of a smooth region or belongs to an edge of a particular orientation. If the local feature is identified as a smooth texture, the optimal RSD prediction filters perform blurring to dilute the effect of the halftone noise, whereas for an edge region the filters tend to preserve the local image structure.

The output of the RSD module $u(i,j)$ is a grayscale image at 600 dpi in which the screen patterns have been substantially removed while the nonhalftone details are essentially preserved. However, the overall visual quality of the RSD output is not sufficiently high to be used as the final descreened result. The image $u(i,j)$ is input to the modified SUSAN filter where it is used to perform an edge preserving smoothing of the color scanned data that results in the descreened output.

In order to prevent color shifts during the descreening process, it is important that the modified SUSAN filter be applied to the scanned image data in a linear RGB color space. The sRGB color image $f(k,i,j)$ is transformed to linear color coordinates $f_l(k,i,j)$ using the following relation:

$$f_l(k,i,j) = \omega\left(\frac{f(k,i,j)}{255}\right) \qquad (1)$$

where

$$\omega(x) = \begin{cases} x/12.92, & \text{if } 0 \le x \le 0.03928 \\ \left(\frac{x+0.055}{1.055}\right)^{2.4}, & \text{if } 0.03928 < x \le 1. \end{cases} \qquad (2)$$

For each k, the modified SUSAN filter then processes $f_l(k,i,j)$ to produce the linear RGB image $g_l(k,i,j)$ which is then gamma corrected to produce the sRGB output $g(k,i,j)$. The transformation from linear to gamma corrected color coordinates is given by the relation

$$f(k,i,j) = 255\omega^{-1}\left(f_l(k,i,j)\right) \qquad (3)$$

where

$$\omega^{-1}(x) = \begin{cases} 12.92x, & \text{if } 0 \le x \le 0.00304 \\ 1.055x^{\frac{1}{2.4}} - 0.055, & \text{if } 0.00304 < x \le 1. \end{cases} \qquad (4)$$

The training-based RSD algorithm is discussed in Section II-A. In Section II-B, we introduce the basic SUSAN filtering algorithm and develop the modified SUSAN filter.

A. Resolution Synthesis-Based Denoising (RSD)

An overview of the training-based RSD procedure is presented in Fig. 2, while a detailed illustration of the predictor is provided in Fig. 3. The RSD predictor discussed in this section is similar to the RS predictor developed by Atkins *et al.* [4], [5] for optimal image scaling. The prediction parameters comprise the classification parameters $\boldsymbol{\theta}$ and filter parameters $\boldsymbol{\psi}$ which are computed in an offline training process.

The training data comprises pairs of low-resolution (300 dpi) scanned images and their corresponding spatially registered high-resolution (600 dpi) continuous-tone digital originals (see Figs. 2 and 4). The scans contain residual energy from a periodic halftone screen that can produce Moiré artifacts on printing. The digital originals do not contain such Moiré causing frequencies and therefore can be printed without creating these undesirable interference patterns. The spatial alignment of low-resolution scans and high-resolution originals is accomplished using subpixel registration techniques developed in [12].

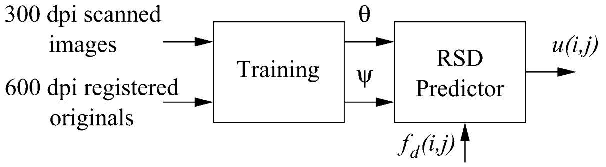


Fig. 2. Overview of the training-based RSD procedure.

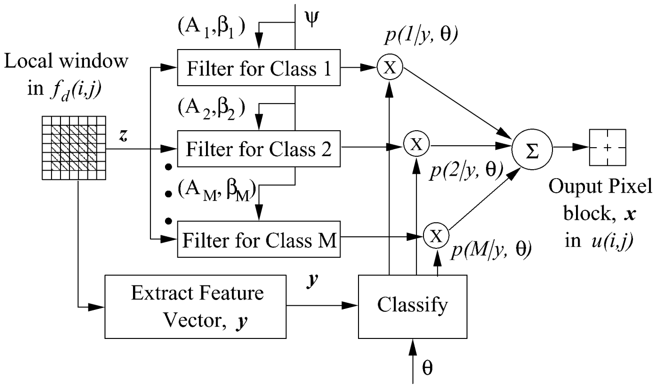


Fig. 3. Illustration of the RSD predictor.

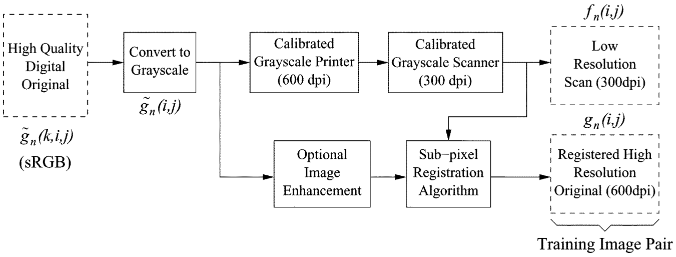

High Quality Digital Original
g̃_n(k,i,j) (sRGB)
Convert to Grayscale
g̃_n(i,j)
Calibrated Grayscale Printer (600 dpi)
Calibrated Grayscale Scanner (300 dpi)
f_n(i,j)
Low Resolution Scan (300dpi)
Optional Image Enhancement
Sub-pixel Registration Algorithm
g_n(i,j)
Registered High Resolution Original (600dpi)
Training Image Pair

Fig. 4. Generation of training image pair. $f_n(i,j)$ is a 300-dpi scanned image. $g_n(i,j)$ is the corresponding 600-dpi registered grayscale digital original image. The training image pairs $\{(f_n(i,j), g_n(i,j))\}_{n\in\{1,\ldots,N_{\text{images}}\}}$, where N_{images} denotes the number images in the training set, is used to design classification and filter parameters for the RSD predictor.

The input to the RSD predictor is the grayscale decimated luminance image $f_d(i,j)$ at 300 dpi. The decimation from 600 to 300 dpi is performed by replacing each nonoverlapping 2×2 pixel block in the high-resolution image with its average value. For each pixel in the low-resolution input, the prediction task is to compute the corresponding block of high-resolution pixels in the original. Thus, in addition to performing scaling as in the RS predictor [5], we require that RSD eliminate the periodic screen noise in scans. For prediction we use pixels in a 7×7 neighborhood centered at the input pixel. These pixels are written as the elements of an observation vector $\boldsymbol{z}$. The RSD predictor extracts the spatial feature vector $\boldsymbol{y}$ from the observation vector. It is assumed that the feature vector $\boldsymbol{y}$ is distributed with an M class mixture distribution. Using the parameters $\boldsymbol{\theta}$, the RSD classifier computes the probability that the feature vector $\boldsymbol{y}$ belongs to a particular component j in the mixture. To estimate the high-resolution pixels, the observation vector $\boldsymbol{z}$ is first filtered with the optimal predictors for the individual classes. The output vector $\boldsymbol{x}$ is then computed as a linear combination of the outputs of all M filters with the weighting function for the jth filter corresponding to the probability that the image data is in class j.

In the remainder of this section, we derive the output of the RSD predictor and discuss its optimality under certain assumptions. The derivation is similar to that given in [4]. Then we discuss efficient RSD which provides a way to reduce computation and expedite the prediction process. This is followed by a description of the feature vector used for local window classification. Next we explain how to generate the training data for estimating the prediction parameters. Finally, we give the training algorithm for optimizing the parameters of the RSD predictor.

1) Optimal Prediction: In this subsection, the output of RSD is derived as an MMSE predictor, assuming that the classifier parameters $\boldsymbol{\theta}$ and filter parameters $\boldsymbol{\psi}$ are known. In the following discussion, we shall use upper-case letters for random variables and lower-case letters for their realizations. The analysis is based on the following assumptions about the image data.

Assumption 1: The feature vector $\boldsymbol{Y}$ is distributed as a multivariate Gaussian mixture. The probability density function of $\boldsymbol{Y}$ can be written as

$$p_{\boldsymbol{Y}}(\boldsymbol{y}) = \sum_{j=1}^{M} p_{\boldsymbol{Y}|J}(\boldsymbol{y}|j)\pi_j \tag{5}$$

where $J \in \{1, \ldots, M\}$ is a random variable representing the class, π_j is the probability that $J = j$, and $p_{\boldsymbol{Y}|J}(\boldsymbol{y}|j)$ is a multivariate Gaussian density for each j. For the jth Gaussian distribution we assume that the mean vector is $\boldsymbol{\mu}_j$, while for all classes in the mixture the covariance matrix is assumed to be given by the same diagonal matrix $\boldsymbol{\Lambda}^2$. The covariance matrix could be chosen to vary with class, however, experimentally we observed that using the same covariance matrix for all classes improves the results. This happens because the reduced number of parameters makes the training more effective and accurate. We define the nonzero diagonal elements of $\boldsymbol{\Lambda}$ by the vector $[\sigma_1, \ldots, \sigma_d]^T$, where σ_i represents the standard deviation of the ith component of the feature vector $\boldsymbol{y}$ and d represents the dimension of the feature vector. The conditional density $p_{\boldsymbol{Y}|J}(\boldsymbol{y}|j)$ is given by

$$p_{\boldsymbol{Y}|J}(\boldsymbol{y}|j) = \frac{|\boldsymbol{\Lambda}|^{-1}}{(2\pi)^{d/2}} \exp\left\{-\frac{1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_j)\right\|^2\right\} \tag{6}$$

where x denotes the L_2 norm of vector x.

The parameters of the mixture distribution $p_{\boldsymbol{Y}}(\boldsymbol{y})$ are written as $\boldsymbol{\theta} = \{\{\boldsymbol{\mu}_j, \pi_j\}_{j=1}^{M}, \boldsymbol{\Lambda}\}$.

Assumption 2: The conditional distribution of $\boldsymbol{X}$ given $\boldsymbol{Z}$ and J is a multivariate Gaussian, with mean $\mathbf{A}_J\boldsymbol{Z} + \boldsymbol{\beta}_J$. The filter parameters represented by $\boldsymbol{\psi}$ are obtained directly from the parameters for these distributions, so we write $\boldsymbol{\psi} = (\{\mathbf{A}_j\}_{j=1}^{M}, \{\boldsymbol{\beta}_j\}_{j=1}^{M})$.

Assumption 3: The class J is conditionally independent of the high- and low-resolution vectors $\boldsymbol{X}$ and $\boldsymbol{Z}$, given the feature vector $\boldsymbol{Y}$. Formally, this means that

$$p_{J|\boldsymbol{X},\boldsymbol{Z}}(j|\boldsymbol{x},\boldsymbol{z}) = p_{J|\boldsymbol{Y}}(j|\boldsymbol{y}). \tag{7}$$

With these assumptions, the MMSE [25] estimate is given by

$$\begin{aligned}\hat{\boldsymbol{X}} &= E[\boldsymbol{X}|\boldsymbol{Z}] &(8)\\ &= \sum_{j=1}^{M} E[\boldsymbol{X}|\boldsymbol{Z}, J=j]p_{J|\boldsymbol{Z}}(j|\boldsymbol{Z}) &(9)\\ &= \sum_{j=1}^{M}(\mathbf{A}_j\boldsymbol{Z}+\boldsymbol{\beta}_j)p_{J|\boldsymbol{Z}}(j|\boldsymbol{Z}) &(10)\\ &= \sum_{j=1}^{M}(\mathbf{A}_j\boldsymbol{Z}+\boldsymbol{\beta}_j)p_{J|\boldsymbol{Y}}(j|\boldsymbol{Y}). &(11)\end{aligned}$$

Equation (10) is obtained by invoking assumption 2. To obtain (11), we invoke assumption 3 and use the fact that $p_{J|\boldsymbol{Z}}(j|\boldsymbol{Z}) = \int p_{J|\boldsymbol{X},\boldsymbol{Z}}(j|\boldsymbol{x},\boldsymbol{Z})p_{X|\boldsymbol{Z}}(\boldsymbol{x}|\boldsymbol{Z})d\boldsymbol{x}$. The distribution $p_{J|\boldsymbol{Y}}(j|\boldsymbol{y},\boldsymbol{\theta})$ can be computed using Bayes' rule and assumption 1 as follows:

$$p_{J|\boldsymbol{Y}}(j|\boldsymbol{y},\boldsymbol{\theta}) = \frac{\exp\left(\frac{-1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_j)\right\|^2\right)\pi_j}{\sum_{l=1}^{M}\exp\left(\frac{-1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_l)\right\|^2\right)\pi_l}. \tag{12}$$

Inserting (12) into (11), we obtain the equation for optimal filtering in terms of $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$

$$\hat{\boldsymbol{X}} = \sum_{j=1}^{M}(\mathbf{A}_j\boldsymbol{Z}+\boldsymbol{\beta}_j)\cdot\frac{\exp\left(\frac{-1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_j)\right\|^2\right)\pi_j}{\sum_{l=1}^{M}\exp\left(\frac{-1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_l)\right\|^2\right)\pi_l}. \tag{13}$$

2) Efficient RSD: An important parameter that affects the performance of the RSD predictor is the number of classes M in the mixture model. In our application, we use $M = 60$. From (13), we observe that since the output pixel block $\hat{\boldsymbol{X}}$ is computed as a linear combination of all M of the RSD filters, choosing $M = 60$ would require an excessive amount of computation. Thus, for a given pixel, Atkins [4] suggested using only those classes for averaging that best represent the local texture in the image. We determine the output for efficient RSD as follows:

$$\hat{\boldsymbol{X}}_{\mathrm{ERSD}} = \sum_{j\in J^*}(\mathbf{A}_j\boldsymbol{Z}+\boldsymbol{\beta}_j) \cdot\frac{\exp\left(\frac{-1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_j)\right\|^2\right)\pi_j}{\sum_{l\in J^*}\exp\left(\frac{-1}{2}\left\|\boldsymbol{\Lambda}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}_l)\right\|^2\right)\pi_l} \tag{14}$$

where $J^* \subset \{1, \ldots, M\}$ is the set containing classes which are the most probable representatives of the local texture. The set J^* is characterized by the following set of filter selection criterion:

$$J^* = \left\{j : \frac{p(j|\boldsymbol{y})}{p(\tilde{j}|\boldsymbol{y})} \geq e^{-\delta^2}\right\} \tag{15}$$

where $\tilde{j}$ is the most likely class for the local feature vector $\boldsymbol{y}$ and $\delta \geq 0$. Note that $\delta = 0$ is equivalent to using only one filter per pixel, while a large value of δ, for example $\delta = 15$, causes all filters to be used for determining the current pixel value.

3) Extraction of Local Feature Vector: The spatial feature vector $\boldsymbol{y}$ is used for local window classification in the decimated luminance image, $f_d(i,j)$. It is important to select spatial features that satisfy the following criteria. 1) The feature vector should be able to distinguish between smooth textures and edges of different orientation in the presence of halftone noise. 2) Extraction of the feature vector from the local mask should not be computationally intensive.

We considered a variety of methods for feature extraction including the nonlinear projection operator in [4], Gabor filters of different radial frequencies and orientations [36], and Laws 2-D convolution kernels [20]. For our purposes, we found that Laws energy filters worked well in comparison to the other methods. Laws 2-D convolution kernels are separable, providing a faster implementation, and can be generated using the following set of 1-D convolution kernels: level, $L5 = [1\ 4\ 6\ 4\ 1]$; edge, $E5 = [-1\ -2\ 0\ 2\ 1]$; spot, $S5 = [-1\ 0\ 2\ 0\ -1]$; wave, $W5 = [-1\ 2\ 0\ -2\ 1]$; and ripple $R5 = [1\ -4\ 6\ -4\ 1]$.

From these 1-D kernels, we can produce 25 different 2-D kernels by convolving a vertical 1-D kernel with a horizontal 1-D kernel. Thus, convolving a vertical L5 with a horizontal E5 would produce the 2-D kernel L5E5. However most of the 2-D kernels produced with this method would be sensitive to the halftone noise in the scanned document. Thus, we select only those 2-D kernels that are obtained by convolution of a 1-D vector with an L5 vector. The inherent directional smoothing performed by these 2-D kernels helps to produce features that give stable classification even in the presence of halftone noise. Thus, we choose the following eight 2-D kernels for extracting the feature vector: $L5E5$, $E5L5$, $L5S5$, $S5L5$, $L5W5$, $W5L5$, $L5R5$, and $R5L5$. Each of these kernels is normalized and applied in succession to the local window in the image $f_d(i,j)$ to get the 8-D feature vector $\boldsymbol{y}$.

In [20], Laws has also proposed 3 × 3 and 7 × 7 convolution kernels. However, we have found the 5 × 5 kernels to give the best performance. We must emphasize that at the considered resolution of 600 dpi, the captured halftone dot size in the digital scan is substantially large which renders the 5 × 5 masks unsuitable for direct application to the image data at full scanner resolution. Decimating the luminance image $f(i,j)$, as in Fig. 1(c), blends the periodic dots and enables the small kernels to give a meaningful description of the underlying image texture.

4) Generation of Training Set: Fig. 4 illustrates how the training image pairs can be obtained for optimizing parameters of the RSD predictor. The high-quality electronic original $\tilde{g}_n(k,i,j)$ is a digital document that is carefully chosen to represent a wide variety of textures and text of different fonts and sizes. We assume that the digital document is a color image in sRGB coordinates. The color digital document is converted to grayscale image $\tilde{g}_n(i,j)$ by computing a weighted sum of the nonlinear sRGB values: $\tilde{g}_n(i,j) = 0.30\tilde{g}_n(0,i,j) + 0.59\tilde{g}_n(1,i,j) + 0.11\tilde{g}_n(k,i,j)$. The grayscale digital original $\tilde{g}_n(i,j)$ is then optionally enhanced to incorporate built-in sharpening in the RSD prediction filters. To produce the low-resolution image, the grayscale original is printed at a resolution of 600 dpi followed by scanning at 300 dpi. Thus, the grayscale digital original is roughly twice the size of the scan. However, to be used in the training procedure, it is important that the pairs of scans and originals be perfectly registered together. This task is accomplished by using the subpixel registration algorithm developed by Han [12]. Note that the registered high-resolution image $g_n(i,j)$ will be exactly twice the size of the low-resolution scan $f_n(i,j)$.

Apart from misregistration, the print-to-scan process in Fig. 4 can cause the low-resolution and high-resolution images in the training pair $(f_n(i,j), g_n(i,j))$ to mismatch in tone. To ensure that both the images in the registered training pair have the same tone, it is important that calibrated printer and scanner be used to generate the training data.

The training set plays a critical role in the quality of RSD output. With the selection of a proper training set, it is possible to tune the RSD predictor for a particular level of detail preservation. Using a training set with more smooth textures produces more suppression of the halftone screen. On the other hand, including more examples of nonhalftoned detail in the training set results in improved rendition of text, edges, and high-frequency image detail.

5) Estimating Predictor Parameters: Our objective here is to compute the maximum likelihood (ML) estimates [25] of $\boldsymbol{\psi}$ and $\boldsymbol{\theta}$ from training images by extracting example realizations of the pair $(\boldsymbol{Z}, \boldsymbol{X})$, which we assume are independent. This is generally difficult since the data does not reveal realizations of the class label J. This is known as the incomplete data problem, where observations of the triplet $(\boldsymbol{Z}, \boldsymbol{X}, J)$ would be complete data. To address the problem, the expectation-maximization (EM) algorithm [9], [35] has been used. The training algorithm, derived in [4], has the following steps.

1) Build the training set using the method discussed in Section II-A4. The training set is represented as $\{(f_n(i,j), g_n(i,j))\}_{n\in\{1,\ldots,N_{\text{images}}\}}$, where $f_n(i,j)$ are 300-dpi scanned images, $g_n(i,j)$ are 600-dpi registered digital original images, and N_{images} is the number of images in the training set.

2) Extract training vectors from the training set. A training vector is a pair of low- and high- resolution vectors $(\boldsymbol{z}, \boldsymbol{x})$ corresponding to any one of the pixels in the low-resolution images. We will denote the set of low-resolution pixels in the training set as S, and the extracted training vectors as $(\boldsymbol{z}_s, \boldsymbol{x}_s)_{s\in S}$. By writing $\boldsymbol{y}_s$, we refer to the spatial feature vector extracted from the local observation window using Laws masks discussed in Section II-A3. We will denote as N the number of training vectors. In our experiments N was selected to be 100 000.

3) Select a value for the number M of classes. For best performance we recommend using 50–60 classes, however, satisfactory results can be obtained with as few as 25–30 classes.

4) Initialize estimate of $\boldsymbol{\theta}$. For each $j \in \{1, \ldots, M\}$, set $\pi_j^{(0)}$ to $1/M$, and set each cluster mean $\boldsymbol{\mu}_j^{(0)}$ to equal one of the feature vectors from the training set. Set $\sigma_i^{2^{(0)}}$, $i \in \{1, \ldots, d\}$ as the sample variance of the ith element in the feature vector. Note that d is the dimensionality of the feature and equals 8 in our experiment.

5) Iterate the update (16)–(19) for $1 \le j \le M$ to estimate $\boldsymbol{\theta}$

$$N_j^{(k+1)} = \sum_{s\in S} p_{J|\boldsymbol{Y}}\left(j|\boldsymbol{y}_s, \boldsymbol{\theta}^{(k)}\right) \tag{16}$$

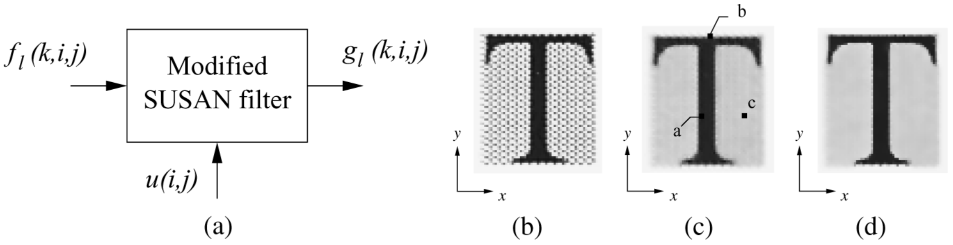


Fig. 5. Modified SUSAN filter. (a) Block diagram. (b) Scanned image $f_l(k,i,j)$. (c) Denoised image $u(i,j)$. (d) Processed image $g_l(k,i,j)$.

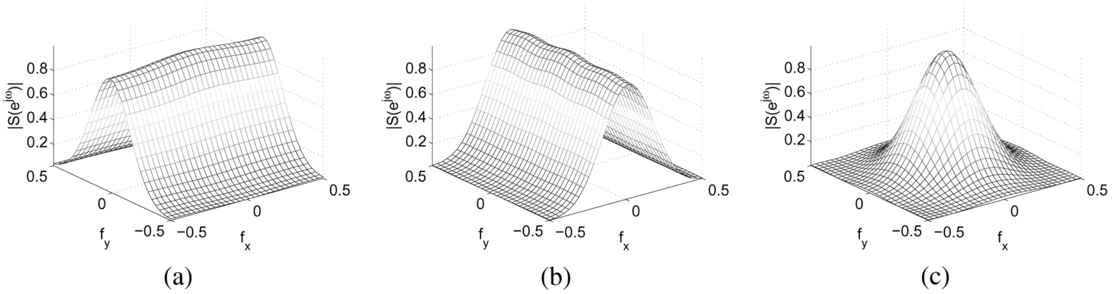


Fig. 6. Frequency response of modified SUSAN filter computed using image $u(i,j)$ in Fig. 5(c). (a) Response at pixel a. (b) Response at pixel b. (c) Response at pixel c.

$$\pi_j^{(k+1)} = \frac{N_j^{(k+1)}}{N} \tag{17}$$

$$\boldsymbol{\mu}_j^{(k+1)} = \frac{1}{N_j^{(k+1)}} \cdot \sum_{s\in S} \boldsymbol{y}_s p_{J|\boldsymbol{Y}}\left(j|\boldsymbol{y}_s, \boldsymbol{\theta}^{(k)}\right) \tag{18}$$

$$\sigma_i^{2(k+1)} = \sum_{j=1}^{M} \left[\pi_j^{(k+1)} \Xi_{j,i}\right] \tag{19}$$

where $i \in \{1,\ldots,d\}$ and

$$\Xi_{j,i} = \frac{1}{N_j^{(k+1)}} \sum_{s\in S} \left(y_{s,i} - \mu_{j,i}^{(k+1)}\right)^2 \cdot p_{J|\boldsymbol{Y}}\left(j|\boldsymbol{y}_s, \boldsymbol{\theta}^{(k)}\right). \tag{20}$$

Stop iterating through the update equations when $|N_j^{(k+1)} - N_j^{(k)}| < \epsilon$. At this point, we regard the resulting estimate $\hat{\theta}$ as the final estimate of θ.

Note that $y_{s,i}$ and $\mu_{j,i}$ are the i^{th} components of the d-dimensional vectors $\boldsymbol{y}_s$ and $\boldsymbol{\mu}_j$, respectively.

6) Estimate $\boldsymbol{\psi}$. For $1 \le j \le M$, define

$$N_j = \sum_{s\in S} p_{J|\boldsymbol{Y}}(j|\boldsymbol{y}_s, \hat{\boldsymbol{\theta}}) \tag{21}$$

$$\boldsymbol{b}_s = \begin{pmatrix} \boldsymbol{x}_s \\ \boldsymbol{z}_s \end{pmatrix} \tag{22}$$

$$\nu_j = \begin{pmatrix} \boldsymbol{\nu}_{\boldsymbol{x}|j} \\ \boldsymbol{\nu}_{\boldsymbol{z}|j} \end{pmatrix} \tag{23}$$

$$= \frac{1}{N_j} \sum_{s\in S} \boldsymbol{b}_s p_{J|\boldsymbol{Y}}(j|\boldsymbol{y}_s, \hat{\boldsymbol{\theta}}) \tag{24}$$

$$\boldsymbol{\Sigma}_j = \begin{pmatrix} \boldsymbol{\Sigma}_{\boldsymbol{xx}|j} & \boldsymbol{\Sigma}_{\boldsymbol{xz}|j} \\ \boldsymbol{\Sigma}_{\boldsymbol{xz}|j}^t & \boldsymbol{\Sigma}_{\boldsymbol{zz}|j} \end{pmatrix} \tag{25}$$

$$= \frac{1}{N_j} \sum_{s\in S} \boldsymbol{b}_s \boldsymbol{b}_s^t p_{J|\boldsymbol{Y}}(j|\boldsymbol{y}_s, \hat{\boldsymbol{\theta}}). \tag{26}$$

TABLE I

COLOR CHARACTERIZATION OF HP SCANJET 8250. THE R, G, AND B CHANNELS OF THE SCANNER ARE DENOTED BY $k = 0$, 1, AND 2, RESPECTIVELY. THE NONLINEAR RESPONSE FUNCTION OF THE k-TH COLOR CHANNEL IS MODELED BY $\varphi_s(x,k) = a_s(k)(x/255)^{\gamma_s(k)} + b_s(k)$, WHERE $x \in \{0,\ldots,255\}$ REPRESENTS THE INPUT R, G, OR B DIGITAL VALUE. THE 3 × 3 MATRIX T GIVES TRANSFORMATION FROM LINEAR RGB TO CIE XYZ COLOR SPACE

k	$a_s(k)$	$\gamma_s(k)$	$b_s(k)$	T
0	0.931	2.039	0.015	$\begin{bmatrix} 0.4268 & 0.1584 & 0.3036 \\ 0.3502 & 0.7750 & -0.1735 \\ 0.1606 & 0.7026 & 0.1224 \end{bmatrix}$
1	0.935	2.161	0.009	
2	0.919	2.013	0.011	

TABLE II

CHARACTERIZATION OF GRAYSCALE RESPONSE FUNCTION OF HP SCANJET 8250 AND HP LASER JET 4050. THE RESPONSE FUNCTION OF THE SCANNER IS MODELED BY $\varphi_s(x) = a_s(x/255)^{\gamma_s} + b_s$. THE RESPONSE FUNCTION OF THE PRINTER IS MODELED BY $\varphi_p(x) = a_p(x/255)^{\gamma_s} + b_p$. $x \in \{0,\ldots,255\}$ REPRESENTS THE INPUT GRAYSCALE DIGITAL VALUE

a_s	γ_s	b_s	a_p	γ_p	b_p
0.925	2.127	0.001	0.795	1.545	0.087

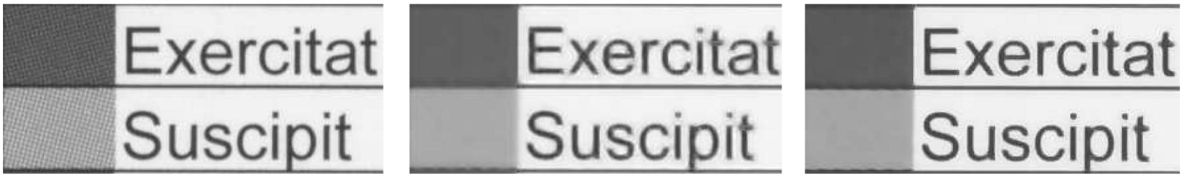
The filter parameters $\boldsymbol{\psi} = (\{\mathbf{A}_j\}_{j=1}^M, \{\boldsymbol{\beta}_j\}_{j=1}^M)$ are then given by

$$\hat{\mathbf{A}}_j = \boldsymbol{\Sigma}_{\boldsymbol{xz}|j} \boldsymbol{\Sigma}_{\boldsymbol{zz}|j}^{-1} \tag{27}$$

$$\hat{\boldsymbol{\beta}}_j = \boldsymbol{\nu}_{\boldsymbol{x}|j} - \boldsymbol{\Sigma}_{\boldsymbol{xz}|j} \boldsymbol{\Sigma}_{\boldsymbol{zz}|j}^{-1} \boldsymbol{\nu}_{\boldsymbol{z}|j}. \tag{28}$$

B. SUSAN Filter

The SUSAN noise filtering algorithm [26] is a nonlinear filtering technique that can preserve fine image structure like thin lines and sharp corners even in the presence of significant noise. The algorithm works by smoothing over only those neighbors in a local mask which have a brightness level similar to the center pixel.



(a) (b) (c)

Fig. 7. (a) Scanned image. Descreened images using (b) RSD alone and (c) a combination of RSD and the modified SUSAN filter.

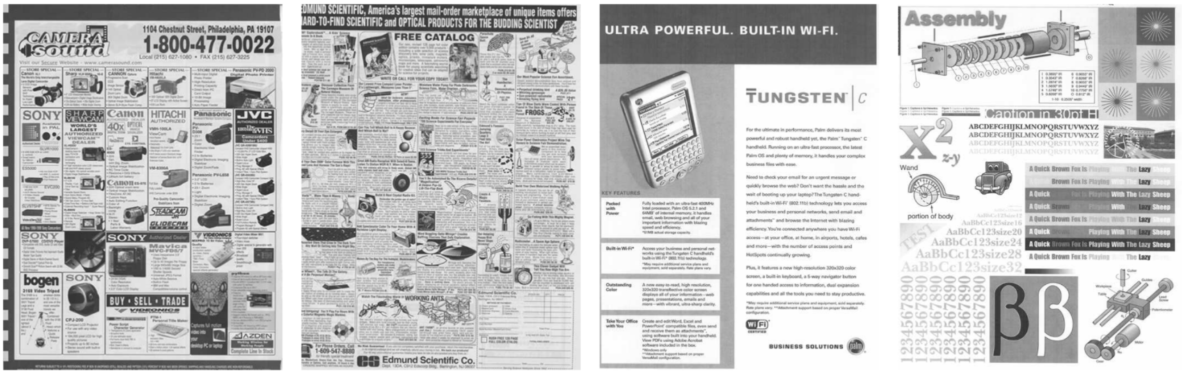


Fig. 8. Thumbnails of sample test documents.

The SUSAN filter, as originally proposed by Smith *et al.* [26], has one input and one output. Let $f_l(k,i,j)$ be the noisy input image, $\tilde{g}_l(k,i,j)$ be the processed output image, (i_o,j_o) be the current pixel, and $N \times N$ be the size of the local window. Furthermore, let σ_s and σ_b be two parameters of the SUSAN filter which control the spatial extent and brightness range of pixels which are smoothed by the filter.

The weight assigned by the filter to the pixel $(i+i_o, j+j_o)$ is given as

$$\begin{aligned}\tilde{w}&(k,i+i_o,j+j_o)\\ &= \exp\left[-\left(\frac{f_l(k,i+i_o,j+j_o)-f_l(k,i_o,j_o)}{\sigma_b}\right)^2\right] \qquad (29)\end{aligned}$$

where $i,j \in [-N,N]$. The output pixel value is then determined using the equation

$$\begin{aligned}\tilde{g}_l(k,i_o,j_o) = \frac{1}{\tilde{c}(k,i_o,j_o)} &\sum_{\substack{i=-N\\ i\neq 0, j\neq 0}}^{N} \sum_{j=-N}^{N} h(i,j)\\ &\cdot \tilde{w}(k,i+i_o,j+j_o) f_l(k,i+i_o,j+j_o) \qquad (30)\end{aligned}$$

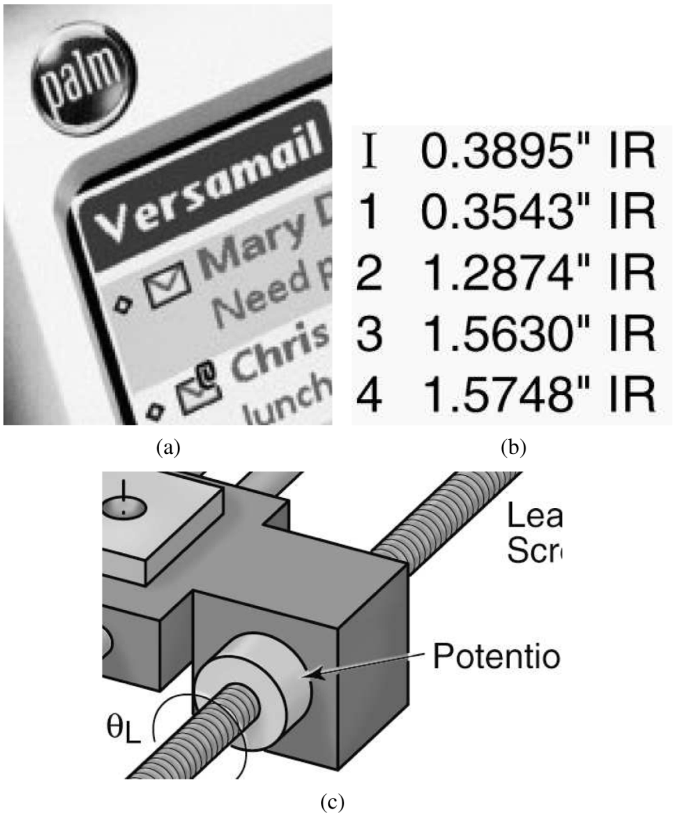
where $h(i,j)$ is a spatial filter weighting given by

$$h(i,j) = \exp\left(-\frac{i^2+j^2}{2\sigma_s^2}\right) \qquad (31)$$

and $\tilde{c}(k,i_o,j_o)$ is a normalizing constant defined by

$$\tilde{c}(k,i_o,j_o) = \sum_{\substack{i=-N\\ i\neq 0, j\neq 0}}^{N} \sum_{j=-N}^{N} h(i,j)\tilde{w}(k,i+i_o,j+j_o). \qquad (32)$$

From (30), we observe that the SUSAN filter is a space-variant smoothing operator that adapts itself spatially so as to protect the local image content. While the standard deviation σ_s controls the scale of the spatial smoothing, the brightness threshold σ_b controls the scale of the "brightness smoothing" [26]. The essential concept of the SUSAN filter is to compute a weighted average of local pixels where the weights are determined by both the spatial distance and lightness distance from



(a) (b)

(c)

Fig. 9. Selected closeup views of digital original images.

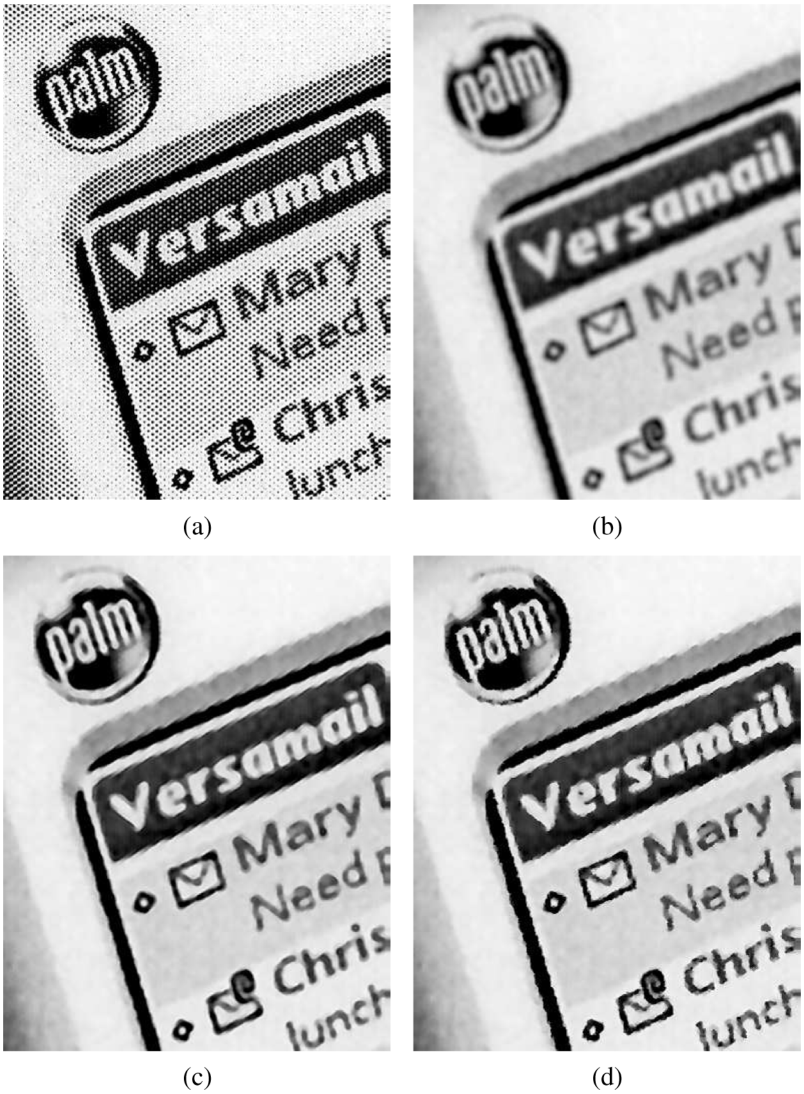


Fig. 10. (a) Scanned image, 600 dpi (source printer: HP Laserjet 4050, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7×7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).

the center pixel. From (30) and (32), it should be noted that the center pixel (i_o, j_o) is not used for averaging in the original SUSAN filter. This is done to provide a better suppression of shot noise. In case the denominator in (30) turns out to be zero, Smith *et al.* suggested using the median of the eight closest neighbors to estimate the pixel's correct value.

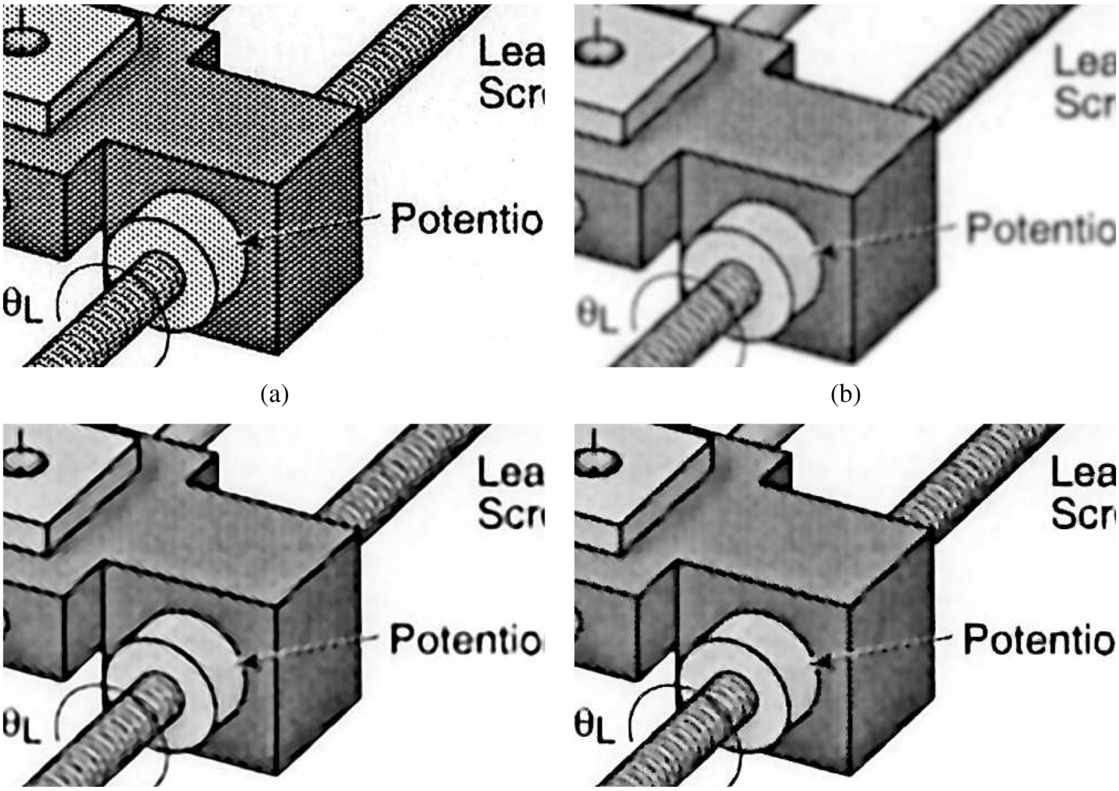
The SUSAN filter is not well suited for removing halftone noise. This is because pixels corresponding to what should be similar underlying continuous-tone gray levels may have very different luminance after halftoning. Consequently, the SUSAN filter will not properly remove the halftone noise.

1) Modified SUSAN Filter: In this section, we develop a modified SUSAN filter that will effectively remove halftoning noise from scanned document. Fig. 5 shows the block diagram of the modified SUSAN filter. The filter has two inputs, $f_l(k, i, j)$ and $u(i, j)$, and one output, $g_l(k, i, j)$. From Fig. 1, we recall that $u(i, j)$ is the grayscale denoised version of the color scanned image. The weights assigned by the modified SUSAN filter to the neighbors around the current pixel (i_o, j_o) are computed using the gray scale denoised luminance image $u(i, j)$ as follows:

$$w(i + i_o, j + j_o) = \exp\left[-\left(\frac{u(i + i_o, j + j_o) - u(i_o, j_o)}{\sigma_b}\right)^2\right] \tag{33}$$

where, as before, $i, j \in [-N, N]$. While the weights are determined using the denoised image, the output of the modified SUSAN filter is computed by averaging over pixels in the color scanned data $f_l(k, i, j)$. The output pixel value is computed using the equation

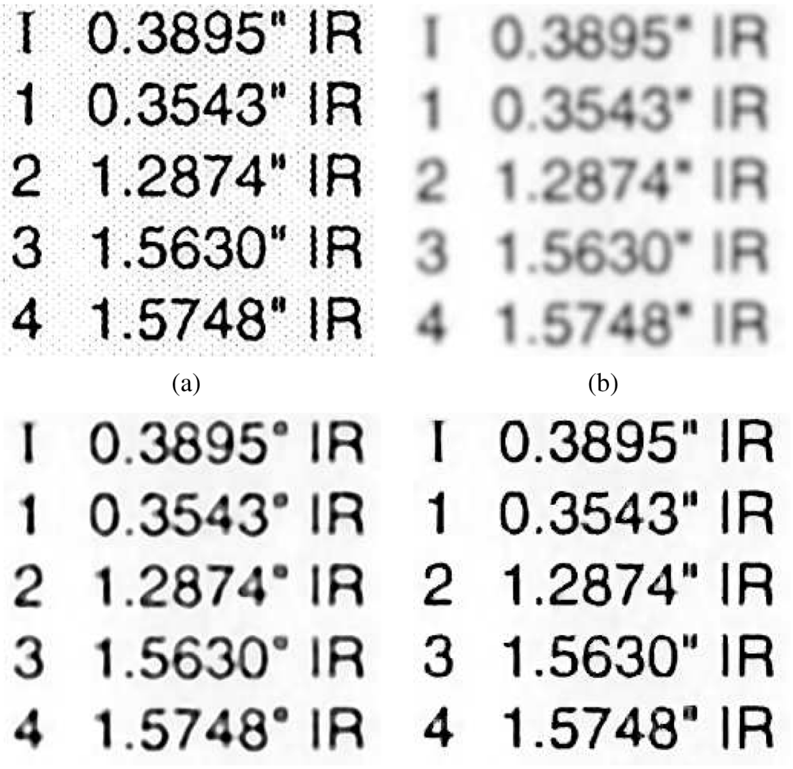
$$g_l(k, i_o, j_o) = \frac{1}{c(i_o, j_o)} \sum_{i=-N}^{N} \sum_{j=-N}^{N} h(i, j) \cdot w(i + i_o, j + j_o) f_l(k, i + i_o, j + j_o) \tag{34}$$



(a) (b)

(c) (d)

Fig. 11. (a) Scanned image, 600 dpi (source printer: HP Laserjet 4050, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7 × 7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).


I 0.3895" IR
1 0.3543" IR
2 1.2874" IR
3 1.5630" IR
4 1.5748" IR

(a) (b)

(c) (d)

Fig. 12. (a) Scanned image, 600 dpi (source printer: HP Laserjet 4050, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7 × 7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).

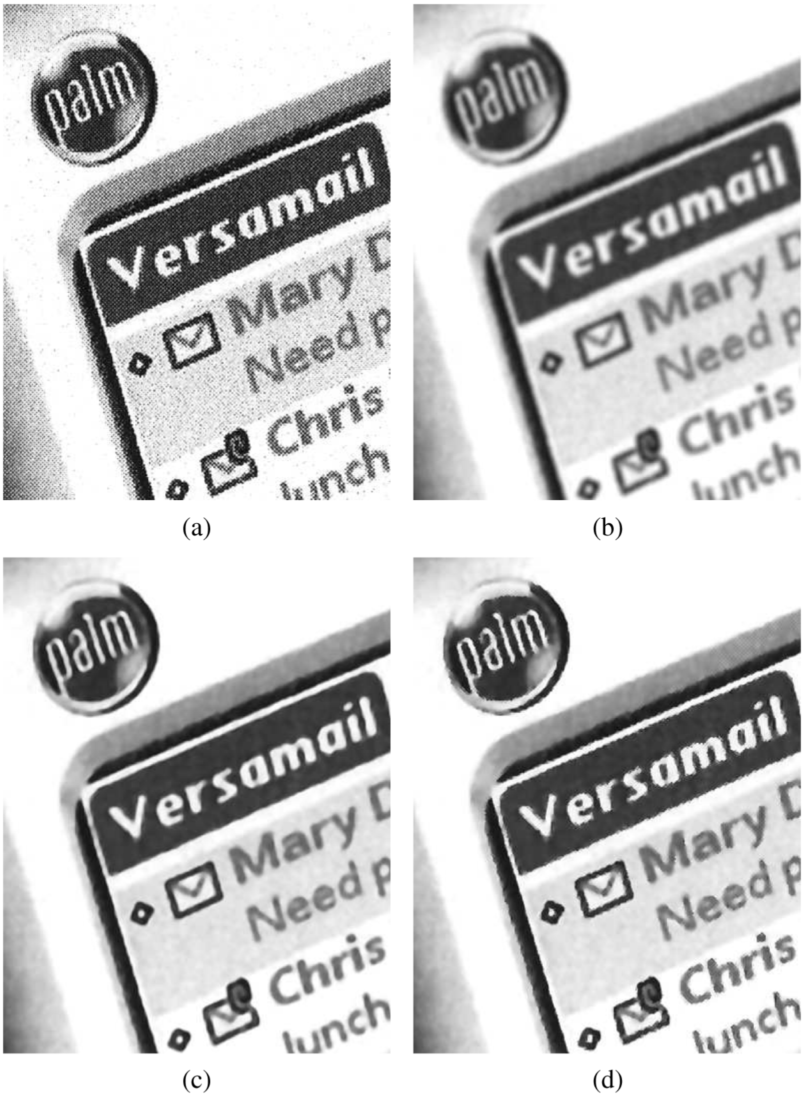

(a) (b) (c) (d)

Fig. 13. (a) Scanned image, 600 dpi (source printer: HP Laserjet 5500, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7×7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).

where $h(i,j)$ is as defined in (31) and the normalizing constant $c(i_o,j_o)$ is given by

$$c(i_o,j_o) = \sum_{i=-N}^{N} \sum_{j=-N}^{N} h(i,j) w(i+i_o, j+j_o). \tag{35}$$

Fig. 5 shows the images at the various inputs and outputs of the modified SUSAN filter. The denoised image $u(i,j)$ acts as a "control input" based on which the filter coefficients are adjusted spatially while processing the linearized scanned image $f_l(k,i,j)$ to produce the output $g_l(k,i,j)$. Fig. 6 shows the transfer function of the filter at three different pixel locations in the letter "T" illustrating how the filter adapts to local variations in the spatial content. As already mentioned, the denoised image $u(i,j)$ could be produced using a conventional Gaussian filter or the halfband filter discussed in [34], but the drawback is that these filters can result in a loss of detail information in the denoised image, such as closely spaced lines. The RSD predictor performs better in terms of preserving details in the denoised image and hence in the final descreened output.

It should be observed that we use a luminance image to derive weights in the modified SUSAN filter. This could lead to blurring in isoluminant regions of the image that are chromatically different. An extension of the existing method that could avoid blurring in the isoluminant regions would be to independently process the R, G, and B channels with the RSD denoiser, and then use the denoised channels to derive the weights in the modified SUSAN filter. However, after evaluating the descreening performance on a wide range of test documents, we felt that processing of the luminance channel with the RSD represented a good trade off between computation and quality.

Finally, comparing (34) and (35) with (30) and (32), we see that the center pixel (i_o, j_o) is included for averaging in the modified SUSAN filter. Experimentally, we found that including the center pixel had no disadvantage, in fact, it helped improve the rendition of fine text and high-frequency image detail in the descreened output.

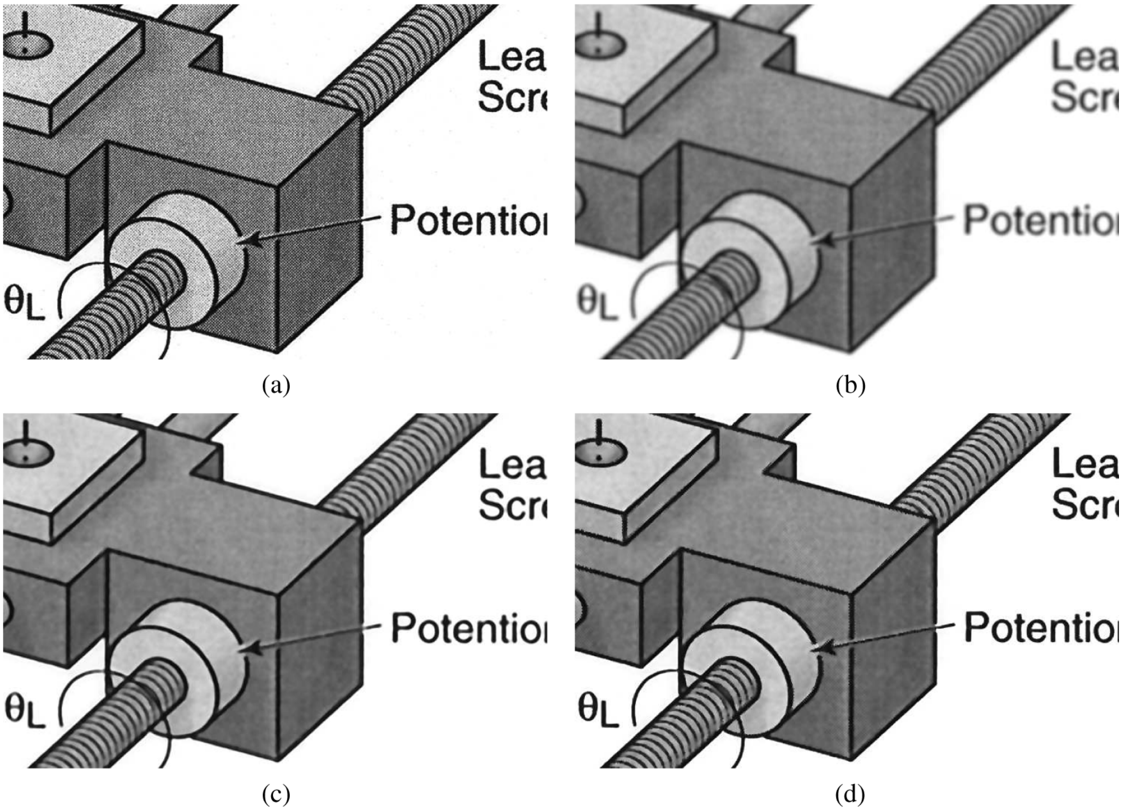


Fig. 14. (a) Scanned image, 600 dpi (source printer: HP Laserjet 5500, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7 × 7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).

III. EXPERIMENTAL RESULTS

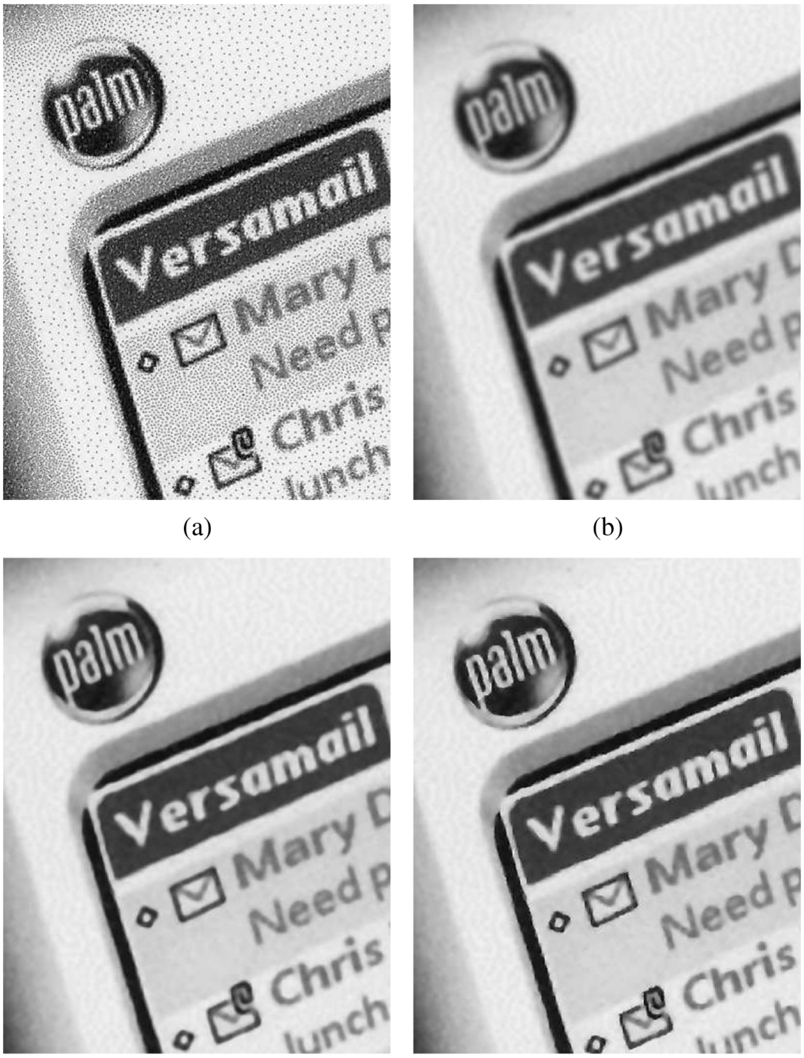
The performance of the descreening algorithm was evaluated on 20 different test images. The test images were obtained by scanning printed originals on an HP Scanjet 8250 at a resolution of 600 dpi. The printed originals included both grayscale and color images. The scanning mode was selected as "Millions of Colors" which produces 24-bit color values in the scanner's device-dependent RGB color space. The grayscale thumbnails for some of the test images are shown in Fig. 8. The printed originals were obtained from a number of different sources: newspapers; magazines; print samples from two different electrophotographic printers, an HP Laserjet 4050 and HP Laserjet 5500; and print samples from an inkjet printer, an HP Deskjet 6540. While newspapers, magazines, and electrophotographic printers use periodic clustered-dot screening methods with varying screen frequencies, the inkjet printer uses a stochastic halftoning method. Thus, the test images were representative of a variety of printing methods commonly used in practice.

The scanner's device-dependent RGB color values were converted to device-independent CIE XYZ color values using the method developed in [37]. The scanner was characterized by three nonlinear tone curves, $\varphi_s(x, k)$ where $k \in \{0, 1, 2\}$, and a 3 × 3 transformation matrix, T, from linear RGB to CIE XYZ space. The results of color scanner characterization are summarized in Table I. Using the CIE XYZ color values, the device-independent sRGB color coordinates for the scanned image were computed as in [3].

We used three different methods to produce the descreened images in this section: 1) low-pass Gaussian filtering; 2) proposed Algorithm-I, which uses a low-pass Gaussian filter for denoising [see Fig. 1(a) and (b)] followed by modified SUSAN filtering; and 3) proposed Algorithm-II, which uses RSD prediction for denoising [see Fig. 1(a) and (c)] followed by modified SUSAN filtering.

Based on the observed performance of descreening, the following parameters were picked up manually. For the low-pass Gaussian filter, we selected the mask size as 7 × 7 and the standard deviation as 2.5. For the modified SUSAN filter, we selected mask size = 7 × 7, $\sigma_s = 2.5$, and $\sigma_b = 21$. For the RSD predictor, we selected the classification window size as 5 × 5, the prediction filter size as 7 × 7, the number of classes in GMM as 60, and the delta factor δ as 2.2 [see (15)]. Choosing this value of δ selects 3–4 most likely classes for processing the input pixel.

The classification and filter parameters of the RSD predictor were generated in an offline training procedure. We used a training set comprising 27 different image pairs. Each training image pair included a 300-dpi grayscale scanned image and its corresponding 600-dpi registered grayscale continuous-tone digital original. The process through which the training image pairs were generated is illustrated in Fig. 4. For the calibrated grayscale source printer shown in Fig. 4, we used an HP Laserjet 4050, while for the calibrated grayscale source scanner, an HP Scanjet 8250 was used in "256 Gray Shades" mode. The results of grayscale printer and scanner characterization are given in Table II.



(a) (b)

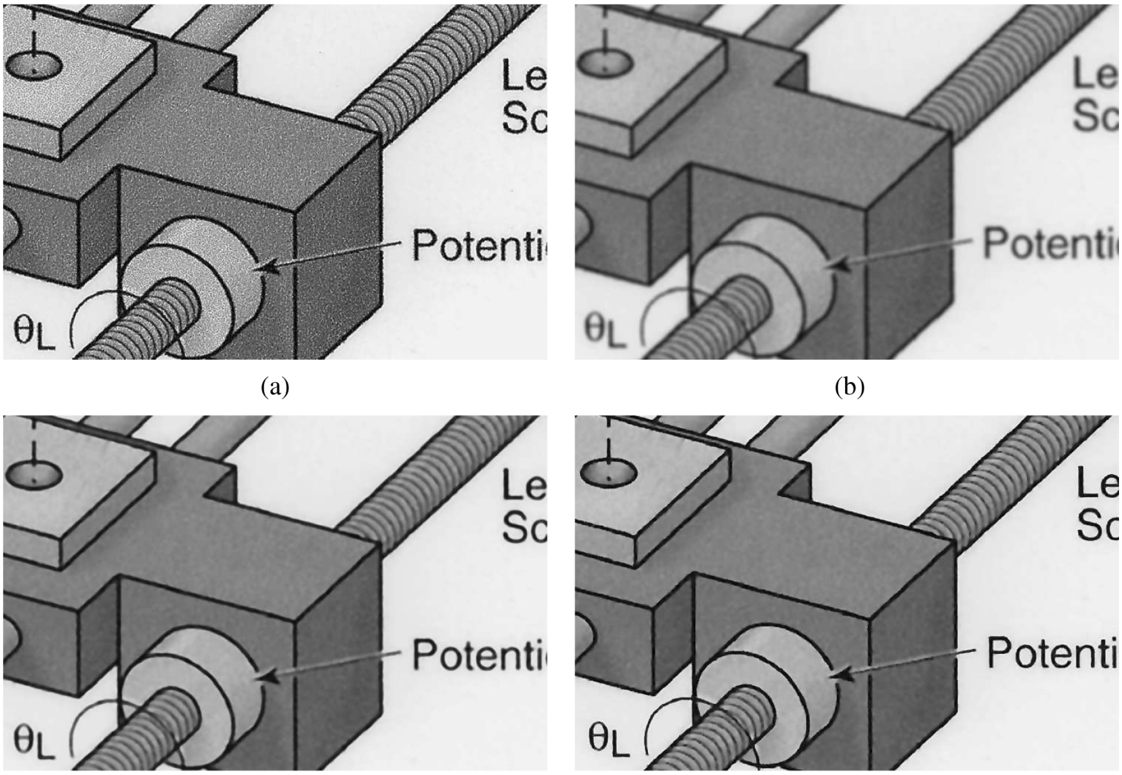
(c) (d)

Fig. 15. (a) Scanned image, 600 dpi (source printer: HP Deskjet 6540, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7×7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).

One important observation that we make is that, while the test images were obtained from a multitude of print sources, only one printer was used for generating the training images. It is also important to emphasize that the training images used for optimizing the RSD predictor were different from the test images used for evaluating the performance of the algorithm.

The RSD predictor performs reasonably well in suppressing halftone noise and preserving image detail, however, the image quality can be substantially improved when a combination of both RSD and modified SUSAN filters is used to generate the final output of the descreening algorithm. This is illustrated in Fig. 7, where we show a comparison of the processed images when RSD is used alone, and when RSD and modified SUSAN filters are used together to descreen the input scanned image.

To demonstrate the performance of the descreening algorithms, we select closeup views of three different regions from images in our test image database. Fig. 9(a)–(c) shows the continuous-tone digital original images. The scanned halftone images, shown in Figs. 10–16(a), are generated by printing the digital originals on three different printers—HP Laserjet 4050, HP Laserjet 5500, and HP Deskjet 6540—followed by scanning on HP Scanjet 8250 at a resolution of 600 dpi. We use an HP Laserjet 4050 for the halftones in Figs. 10–12(a), an HP Laserjet 5500 for the halftones in Figs. 13 and 14(a), and an HP Deskjet 6540 for the halftones in Figs. 15 and 16(a). Figs. 10–16(b) show the results of processing the scanned halftone images with the low-pass Gaussian filter. We notice that, while the Gaussian kernel successfully suppresses the halftone noise, the processed images are excessively blurred. The results in Figs. 10–16(c) and (d) are produced using the proposed algorithms. Proposed Algorithm-I is fast and produces a much higher image quality than the conventional Gaussian blur mask. A further improvement in image quality is achieved using proposed Algorithm-II, which uses training-based RSD prediction for denoising, but at the cost of increased computation. Proposed Algorithm-II produces sharper images than proposed Algorithm-I and performs better in terms of preserving high-frequency nonhalftoned image detail. This is particularly obvious when we compare the



(a) (b)

(c) (d)

Fig. 16. (a) Scanned image, 600 dpi (source printer: HP Deskjet 6540, source scanner: HP Scanjet 8250). Descreened images using (b) Gaussian filter (7 × 7, $\sigma = 2.5$), (c) proposed Algorithm-I (uses Gaussian filter for denoising: filter size $= 7 \times 7$, standard deviation $= 2.5$), and (d) proposed Algorithm-II (uses RSD prediction for denoising).

output images in Fig. 12(c) and (d). We can see that the tiny inch symbols ($''$) are totally blurred in the former, whereas the latter preserves the high-frequency image detail.

Overall, the Algorithm-II yields good quality on text and graphics; however, in some cases, there may be concern about it oversharpening detail. In such cases, it may be desirable to classify regions of the image as text or photographic before descreening, and then apply the appropriate algorithm accordingly.

Finally, we mention the computational complexity of the algorithms. For a 600-dpi letter-sized document, proposed Algorithm-I executes in 28.4 s on a Linux server with 3.20 Intel Xeon CPU. The execution time for proposed Algorithm-II on the same machine is 73.5 s. The algorithms were implemented in C and were not optimized for computation speed. The execution times do not include the image I/O overhead.

IV. CONCLUSION

We developed an efficient descreening algorithm targeted for scan-to-print pipeline in an MFP. The algorithm can effectively remove a wide range of clustered dot screen frequencies, as well as error diffusion noise, in the scanned document while preserving or even enhancing the overall image quality in terms of image sharpness and edge detail. The proposed methodology uses training images generated from a real MFP or scanner and provides a method for automatically optimizing the descreening parameters with respect to the particular imaging hardware.

REFERENCES

[1] J. P. Allebach and B. Liu, “Analysis of halftone dot profile and aliasing in the discrete binary representation of images,” *J. Opt. Soc. Amer.*, vol. 67, no. 9, pp. 1147–1154, 1977.

[2] M. Analoui and J. Allebach, “New results on reconstruction of continuous-tone from halftone,” in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Mar. 23–26, 1992, vol. 3, pp. 313–316.

[3] M. Anderson, R. Motta, S. Chandrasekar, and M. Stokes, “Proposal for a standard default color space for the internet-sRGB,” in *Proc. IS&T and SID’s 4th Color Imaging Conf.: Color Science, Systems, and Applications*, Scottsdale, AZ, Nov. 1996, pp. 238–246.

[4] C. B. Atkins, “Classification-based methods in optimal image interpolation,” Ph.D. dissertation, Purdue Univ., West Lafayette, IN, 1998.

[5] C. B. Atkins, C. A. Bouman, and J. P. Allebach, “Optimal image scaling using pixel classification,” in *Proc. Int. Conf. Image Processing*, Thessaloniki, Greece, 2001, vol. 3, pp. 864–867.

[6] B. E. Bayer, “An optimum method for two-level rendition of continuous-tone pictures,” in *Proc. IEEE Int. Conf. Communications*, Jun. 11–13, 1973, vol. 1, pp. 11–15.

[7] P. C. Chang, C. S. Yu, and T. H. Lee, “Hybrid LMS-MMSE inverse halftoning technique,” *IEEE Trans. Image Process.*, vol. 10, no. 1, pp. 95–103, Jan. 2001.

[8] N. Damera-Venkata and B. L. Evans, “Adaptive threshold modulation for error diffusion halftoning,” *IEEE Trans. Image Process.*, vol. 10, no. 1, pp. 104–116, Jan. 2001.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *J. Roy. Statist. Soc. B*, vol. 39, pp. 1–38, 1977.

[10] Z. Fan, “Retrieval of gray images from digital halftones,” in *Proc. IEEE Int. Symp. Circuits and Systems*, May 3–6, 1992, vol. 5, pp. 2477–2480.

[11] R. W. Floyd and L. Steinberg, “An adaptive algorithm for spatial greyscale,” in *Proc. SID*, 1976, vol. 17, no. 2, pp. 75–77.

[12] Y. Han, “Sub-pixel registration and image analysis with application to image comparison,” Ph.D. dissertation, Purdue Univ., West Lafayette, IN, 2005.

[13] Z. He and C. A. Bouman, "AM/FM halftoning: digital halftoning through simultaneous modulation of dot size and dot density," *J. Electron. Imag.*, vol. 13, no. 2, pp. 286–302, Apr. 2004.

[14] S. Hein and A. Zakhor, "Halftone to continuous-tone conversion of error-diffusion coded images," *IEEE Trans. Image Process.*, vol. 4, no. 2, pp. 208–216, Feb. 2001.

[15] A. Jaimes, F. Mintzer, A. R. Rao, and G. Thompson, "Segmentation and automatic descreening of scanned documents," in *Proc. SPIE*, San Jose, CA, 1999, vol. 3648, pp. 517–528.

[16] T. D. Kite, N. D. Venkata, B. L. Evans, and A. C. Bovik, "A fast, high-quality inverse halftoning algorithm for error diffused halftones," *IEEE Trans. Image Process.*, vol. 9, no. 9, pp. 1583–1592, Sep. 2000.

[17] B. Kolpatzik and C. A. Bouman, "Optimized error diffusion for image display," *J. Electron. Imag.*, vol. 1, no. 3, pp. 277–292, 1992.

[18] D. Lau, G. R. Arce, and N. Gallagher, "Green-noise digital halftoning," *Proc. IEEE*, vol. 86, no. 12, pp. 2424–2444, Dec. 1998.

[19] D. L. Lau, A. Khan, and G. R. Arce, "Stochastic Moiré," in *Proc. IS&T Conf. Image Processing, Image Quality, and Image Capture Systems*, Montréal, QC, Canada, Apr. 2001, pp. 96–100.

[20] K. Laws, "Rapid texture identification," *Proc. SPIE*, vol. 238, pp. 376–380, 1980.

[21] R. Levien, "Output dependent feedback in error diffusion halftoning," in *Proc. IS&T 8th Int. Congr. Advances in Non-Impact Printing Technology*, Oct. 1992, vol. 76, pp. 280–282.

[22] J. Luo, R. de Queiroz, and Z. Fan, "A robust technique for image descreening based on the wavelet transform," *IEEE Trans. Signal Process.*, vol. 46, no. 4, pp. 1179–1184, Apr. 1998.

[23] M. Mese and P. P. Vaidyanathan, "Recent advances in digital halftoning and inverse halftoning methods," *IEEE Trans. Circuits Syst.*, vol. 49, no. 6, pp. 790–805, Jun. 2002.

[24] R. Neelamani, R. Nowak, and R. Baraniuk, "WInHD: Wavelet-based inverse halftoning via deconvolution," *IEEE Trans. Image Process.*, to be published.

[25] L. L. Scharf, *Statistical Signal Processing*. Reading, MA: Addison-Wesley, 1991.

[26] S. M. Smith and J. M. Brady, "SUSAN–A new approach to low level image processing," *Proc. Int. J. Comput. Vis.*, vol. 23, no. 1, pp. 45–78, 1997.

[27] K. E. Spaulding, R. L. Miller, and J. Schidkraut, "Methods for generating blue-noise dither matrices for digital halftoning," *J. Electron. Imag.*, vol. 6, no. 2, pp. 208–230, 1997.

[28] R. L. Stevenson, "Inverse halftoning via MAP estimation," *IEEE Trans. Image Process.*, vol. 4, no. 4, pp. 486–498, Apr. 1997.

[29] J. C. Stoffel and J. F. Moreland, "A survey of electronic techniques for pictorial reproduction," *IEEE Trans. Commun.*, vol. 29, pp. 1898–1925, 1981.

[30] J. Sullivan, L. Ray, and R. L. Miller, "Design of minimum visual modulation halftone patterns," *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 1, pp. 33–38, Jan.–Feb. 1991.

[31] N. T. Thao, "Set theoretic inverse halftoning," in *Proc. IEEE Int. Conf. Image Process.*, Santa Barbara, CA, Oct. 26–29, 1997, vol. 1, pp. 783–786.

[32] R. A. Ulichney, "Dithering with blue noise," *Proc. IEEE*, vol. 76, no. 1, pp. 56–79, Jan. 1988.

[33] N. D. Venkata, M. Venkataraman, T. D. Kite, and B. L. Evans, "Fast blind inverse halftoning," in *Proc. IEEE Int. Conf. Image Processing*, Oct. 4–7, 1998, vol. 2, pp. 64–68.

[34] P. W. Wong, "Inverse halftoning and kernel estimation for error diffusion," *IEEE Trans. Image Process.*, vol. 4, no. 4, pp. 486–498, Apr. 1995.

[35] C. F. J. Wu, "On the convergence properties of the EM algorithm," *Ann. Statist.*, vol. 11, no. 1, pp. 95–103, 1983.

[36] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognit.*, vol. 24, no. 12, pp. 1167–1186, Dec. 1991.

[37] ——, "Using a digital camera for colorimetry of human teeth," in *Proc. IS&T Conf. Image Processing, Image Quality, Image Capture, Systems*, Portland, OR, May 1998, pp. 37–42.

[38] Z. Xiong, M. T. Orchard, and K. Ramchandran, "Inverse halftoning using wavelets," *IEEE Trans. Image Process.*, vol. 8, no. 10, pp. 1479–1483, Oct. 1999.

Hasib Siddiqui received the B.S. degree in electrical engineering from NED University, Pakistan, in 2000, and the M.S. degree in electrical engineering from Purdue University, West Lafayette, IN, in 2003. He is currently pursuing the Ph.D. degree at the School of Electrical and Computer Engineering, Purdue University.

His research interests are in image and signal processing.

Charles A. Bouman (S'86–M'89–SM'97–F'01) received the B.S.E.E. degree from the University of Pennsylvania, Philadephia, in 1981, the M.S. degree in electrical engineering from the University of California, Berkeley, in 1982, and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, in 1989.

From 1982 to 1985, he was a full staff member at the Lincoln Laboratory, Massachusetts Institute of Technology, Lexington. In 1989, he joined the faculty of Purdue University, West Lafayette, IN, 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 Department of Biomedical Engineering. His research focuses on the use of statistical image models, multiscale techniques, and fast algorithms in applications, including medical and electronic imaging.

Dr. Bouman is 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 member of the SPIE professional society, a recipient of IS&T's 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 of the IEEE TRANSACTIONS ON IMAGE PROCESSING, secretary 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 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.