[2] V. Govindaraju and S. N. Shrihari, “Separating handwritten text from interfering strokes,” in *From Pixels and Features in Handwritten Recognition.* New York: Elsevier, 1992, pp. 17–28.
[3] H. Yamada, K. Yamamoto, and K. Hosokawa, “Directional mathematical morphology and reformalized Hough transformation for the analysis of topographic maps,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, pp. 380–387, Apr. 1993.
[4] D. Guillevic and C. Y. Suen, “Cursive script recognition: A fast reader scheme,” in *Proc. Second Int. Conf. Document Analysis and Recognition*, Tsukuba, Japan, Oct. 1993, pp. 311–314.
[5] Y. S. Chen and W. H. Hsu, “An interpretive model of line continuation in human visual perception,” *Pattern Recognit.*, vol. 22, pp. 619–639, 1989.

Comments on “Probability Estimation in Arithmetic and Adaptive-Huffman Entropy Coders”

Glen G. Langdon

This comment points out some omissions in [1] of credit related to scaled-count estimators that perform count-scaling when the count of the less probable symbol (LPS) reaches a limiting value. In prior work, e.g., Gallager’s adaptive Huffman code [2], the counts are scaled based on the total count reaching a limiting value. In [3], however, two algorithms (SS-adap and SSS-adap) are published that scale counts when the value of the LPS-count reaches a limiting value. Since [3] only treats binary distributions, our only interest here concerns the binary distribution aspects of [1].

References

[1] D. Duttweiler and C. Chazmas, “Probability estimation in arithmetic and adaptive-Huffman entropy coders,” *IEEE Trans. Image Processsing*, vol. 4, pp. 237–246, Mar. 1995.
[2] R. Gallager, “Variations on a theme of Huffman,” *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 668–674, Nov. 1977.
[3] G. Langdon and J. Rissanen, “Compression of black-white images with arithmetic coding,” *IEEE Trans. Commun.*, vol. COMM-29, pp. 858–867, June 1981.

Manuscript received October 8, 1995; revised July 27, 1996. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Alan C. Bovik.

The author is with the Department of Computer Engineering, University of California, Santa Cruz, CA 95064 USA (e-mail: langdon@cse.ucsc.edu).

Publisher Item Identifier S 1057-7149(97)05645-5.

An Adaptive Inverse Halftoning Algorithm

Li-Ming Chen and Hsueh-Ming Hang

***Abstract*—A class of inverse halftoning algorithms that recovers gray-scale (continuous-tone) images from halftone images is proposed. The basic structure is an optimized linear filter. Then, a properly designed adaptive postprocessor is employed to enhance the recovered image quality. Finally, a multistage space-varying algorithm is developed that uses the basic linear filter structure as before but with spatially adaptive parameters.**

***Index Terms*—Halftoning, linear filter, postprocessing.**

I. Introduction

Halftoning is a technique converting gray-scale (continuous-tone) images into binary (two-level) images [1]. The two most popular classes of (forward) halftoning techniques are *ordered dithering* and *error diffusion* [1]. However, in many image processing applications we need to recover gray-scale (continuous-tone) images from halftone images, the so-called *inverse halftoning* process. Our goal in this correspondence is to design a high-performance inverse halftoning algorithm that can work with different types of halftoning techniques and image contents.

Several inverse halftoning techniques have been reported, for example [2]–[4]. They range from simple heuristic schemes to complicated iterative schemes. A different approach is taken in this correspondence. We borrow the techniques used in adaptive signal processing [5], image noise reduction, and image compression [6] and apply them to the inverse halftoning problem. There are three concepts adopted by our inverse halftoning schemes. First, the inverse halftoning algorithm design is viewed as an inverse system identification problem using noisy observed data [5]. Second, an adaptive noise removal algorithm based on local image variance is employed to improve the reconstructed image quality [7]. The resultant images have higher peak signal-to-noise ratio (PSNR) and are more pleasant to our eyes. Third, because the image data characteristics is space-varying and localized, the concept of image classification followed by class-dependent processing is adopted [6]. Namely, instead of a single reconstruction filter, a few filters, each tuned to a specific type of local image characteristics, are used for reconstruction.

II. Inverse Halftoning Using Inverse Modeling

In this correspondence, the forward halftoning process, a nonlinear quantization operation, is treated as an unknown noisy plant. The inverse halftoning process is thus viewed as an image restoration procedure that attempts to recover the original gray-scale images from the distorted (halftone) images. Under this formulation, inverse halftoning becomes a typical system identification and signal restoration problem in adaptive signal processing [5] and, therefore, our goal

Manuscript received March 23, 1995; revised August 6, 1996. This work was supported by a grant from Opto-Electronics and Systems Laboratories, Industrial Technology Research Institute, Taiwan, R.O.C. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Charles A. Bouman.

The authors are with the Department of Electronics Engineering, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C.

Publisher Item Identifier S 1057-7149(97)05482-1.

becomes to identify the best inverse model of the forward halftoning process.

Our first attempt is using a linear sliding-window filter (SWF) structure (FIR filter) with the optimal weights derived by minimizing the reconstruction mean square errors. For each to-be-processed pixel located at (k, l), a two-dimensional (2-D) window centered around (k, l) is chosen as the filter support. For simplicity, this window, $\mathcal{S}$, has a rectangular shape. If the window size is $m \times n$, $\mathcal{S} = \{(i, j);\ -m/2 < i \leq m/2 \text{ and } -n/2 < j \leq n/2\}$. The filter weights associated with this window form a vector and is denoted by $\boldsymbol{W} = \{W(i, j);\ (i, j) \in \mathcal{S}\}$. In the reconstruction phase, the reconstructed pixel is computed by linearly combining the halftone pixels inside the window, as follows:

$$\hat{I}(k, l) = \sum_{(i, j) \in \mathcal{S}} \sum b(k-i, l-j) W(i, j) \tag{1}$$

where $\hat{I}(k, l)$ represents the reconstructed gray-scale pixel, and $b(i, j)$, the binary pixel. This procedure is repeated for every pixel in the image, i.e., this window is slid across the image one pixel at a time. In the training phase, $\boldsymbol{W}$ is derived using the well-known least-mean-square (LMS) algorithm [5]. The training data vector, $\boldsymbol{b}_t = \{b(k-i, l-j);\ (i, j) \in \mathcal{S}\}$, is made of the halftone image pixels inside the filter support $\mathcal{S}$ at every legal pixel location (k, l), where t is a sequential index assigned to pixel (k, l). A complete block of training data consists of the original gray-scale image pixel, $I_t = I(k, l)$, and the corresponding halftone training vector $\boldsymbol{b}_t$. Then $\boldsymbol{W}$ is obtained by applying the following formula to the entire set of training data blocks iteratively [5], $\boldsymbol{W}_{t+1} = \boldsymbol{W}_t + 2\beta\varepsilon_t \boldsymbol{b}_t$, where ε_t is the reconstruction error at (k, l), $\varepsilon_t = I_t - \boldsymbol{b}_t^T \boldsymbol{W}_t$, and β is a small updating rate parameter (around 10^{-8}). The above iterative procedure terminates when the mean-square-error (MSE) decrease is insignificant from the previous iteration.

III. Adaptive Postprocessing

Examining carefully the images reconstructed by using the SWF scheme described in the previous section, we can find small "lumps"—clusters of brighter or darker pixels. They are generally of several pixels wide and occur often in the smooth regions. Shift-invariant linear filters cannot be used to remove them because linear filters with low cut-off frequency cause excessive blurring. Median filters are not effective in this situation either because these reconstruction noises are clusters of several contiguous pixels wide rather than being impulsive. In order to retain sharp-edge images, we choose a spatially varying postprocessing algorithm designed based on image local variance. It is a modified version of a previously known algorithm [7].

At each pixel location (k, l), a small neighborhood (window) $\mathcal{R}$ surrounding it is chosen. We compute the mean μ and the standard deviation ν of the pixels inside $\mathcal{R}$. Then, the center pixel $I(k, l)$ is modified according to the following rule:

If

$$\nu \leq K$$

then

$$I'(k, l) = \mu + \left(\frac{\nu}{\nu + K}\right)[I(k, l) - \mu]$$

else

$$I'(k, l) = I(k, l) \tag{2}$$

where the window size is 5×5. Although the parameter K was decided by the noise variance in [7], for simplicity, we choose K value empirically between 25 and 200. The major difference between our algorithm and [7] is that we split pixels into two groups according to their local variance and the low-variance pixels are not adjusted while [7] adjusts every pixel all the time. The above operation (2) is repeated on every pixel over the entire image. When the window $\mathcal{R}$ falls in an edge or texture region, its local variance tends to be larger than that of smooth regions. Therefore, its pixel value is less likely to be adjusted. Hence, such an algorithm provides the desirable smoothing effect without overly blurring edges. Compared to the original algorithm in [7], this modified version has a better visual quality.

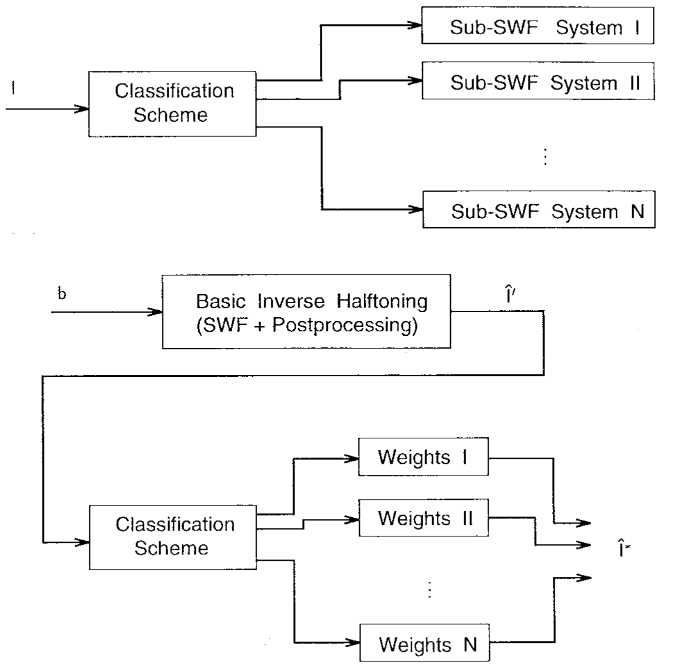


Fig. 1. Adaptive inverse halftoning structure using image classification and sliding-window filters.

TABLE I
PSNR (dB) Produced by Fixed SWF

Test Images	Training Images		
	Lena	Pepper	Jet
Lena	31.3	30.6	31.1
Pepper	30.9	30.9	30.9
Jet	30.3	30.1	30.6

IV. Adaptive Inverse Halftoning Algorithm

Because forward halftoning systems are nonlinear depending upon image local structures, the performance of a single shift-invariant reconstruction filter is limited by its fixed parameters. If different inverse filters are designed aiming at handling different types of image contents, better recovered images can be obtained. Thus, the concept of a piecewise linear system is adopted. We first classify image pixels into a few categories (segments, pieces) according to their local variances and then different sliding-window filters are designed for each of them separately.

Since smooth image regions have already been recovered rather well by a single SWF, we expect that the improvement offered by this multifilter approach mainly resides in the nonsmooth regions. Another advantage of this approach is that it is less sensitive to image contents variation. When a small area of an image is examined,

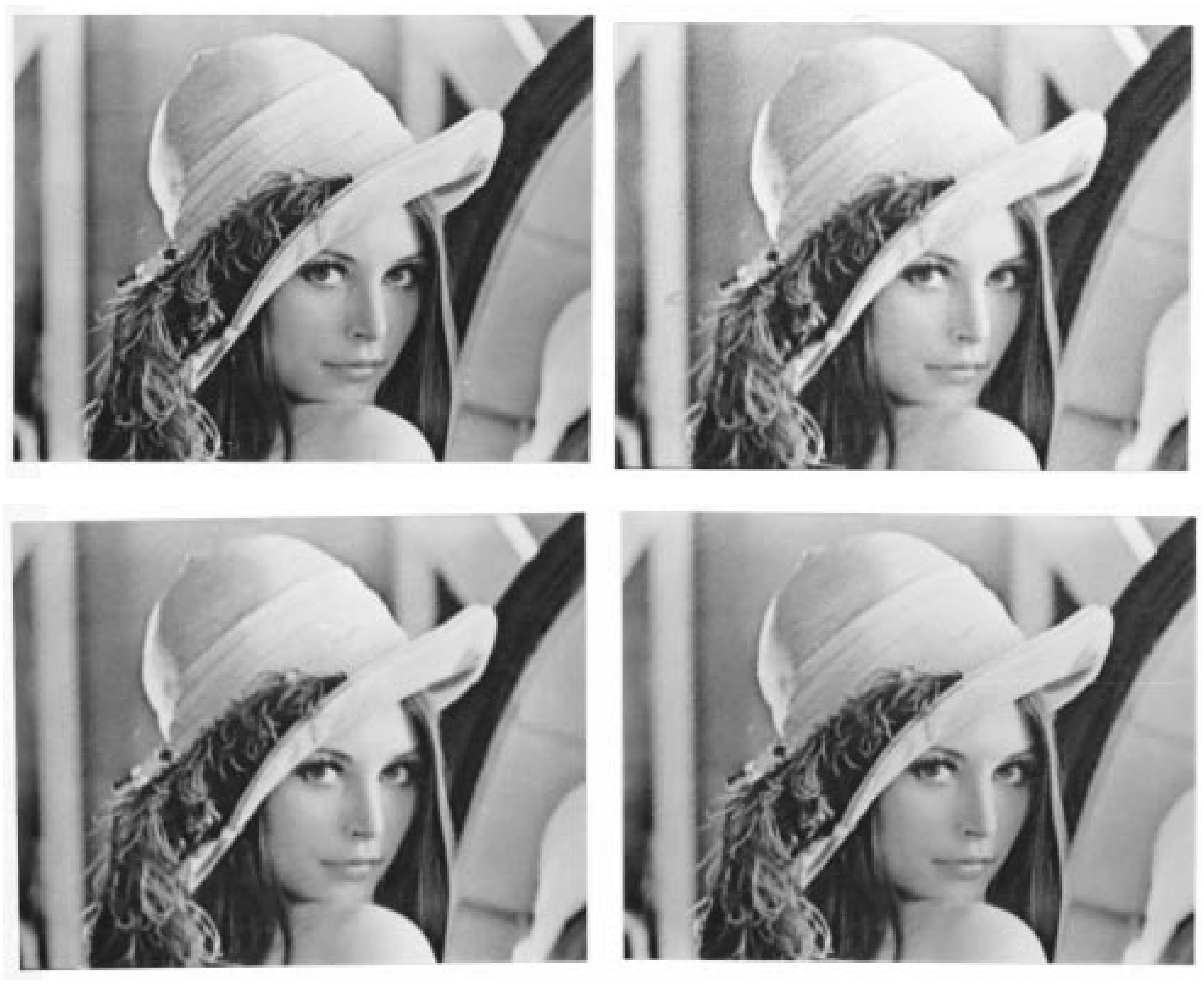


Fig. 2. Reconstructed Lena picture using SWF, SWF with postprocessing, and SV-SWF. Top left: original. Top-right: reconstructed using SWF. Bottom left: reconstructed using SWF with postprocessing. Bottom right: reconstructed using SV-SWF.

TABLE II
PSNR (dB) Produced by Fixed SWF and Postprocessor

Test Images	Training Images		
	Lena	Pepper	Jet
Lena	31.8	30.9	31.7
Pepper	31.6	31.4	31.6
Jet	30.8	30.4	31.1

TABLE III
PSNR (dB) Produced by SV-SWF

Test Images	Training Images		
	Lena	Pepper	Jet
Lena	32.2	31.8	32.2
Pepper	31.7	31.8	31.7
Jet	31.6	31.6	32.0

it often can be classified into one of the generic classes of image patterns such as smooth areas, vertical edges, and horizontal edges. These generic classes are often similar, although the global contents of two images can be very different (Lena image versus jet image). Thus, the image classifier and reconstruction filters designed based on one set of training images can be applied to the other images without significant performance loss.

A. Reconstruction Using a Space-Varying Sliding-Window Filter (SV-SWF)

The general space-varying SWF (SV-SWF) inverse halftoning architecture is shown in Fig. 1. The processing steps are summarized below.

1) Reconstruction Phase: We first compute $\hat{I}(k, l)$, a rough approximation of $I(k, l)$, by applying a space-invariant SWF to the binary pixels centered at (k, l), and then compute $\hat{I}'(k, l)$ using the postprocessing algorithm described in Section III. And then, pixel (k, l) is classified into one of the predesigned categories by the local variance classification scheme (see Section IV-B). Once we know which category the pixel (k, l) belongs to, the corresponding SWF weights (in Fig. 1) can be retrieved and convolve with $b(k, l)$ and its neighbors and the reconstructed pixel $\hat{I}^*(k, l)$ is obtained. We may apply another postprocessing step to $\{\hat{I}^*(k, l)\}$; however, because $\{\hat{I}^*(k, l)\}$ is often very close to the original gray-scale image $\{I(k, l)\}$, this additional postprocessing step usually does not provide noticeable improvement.

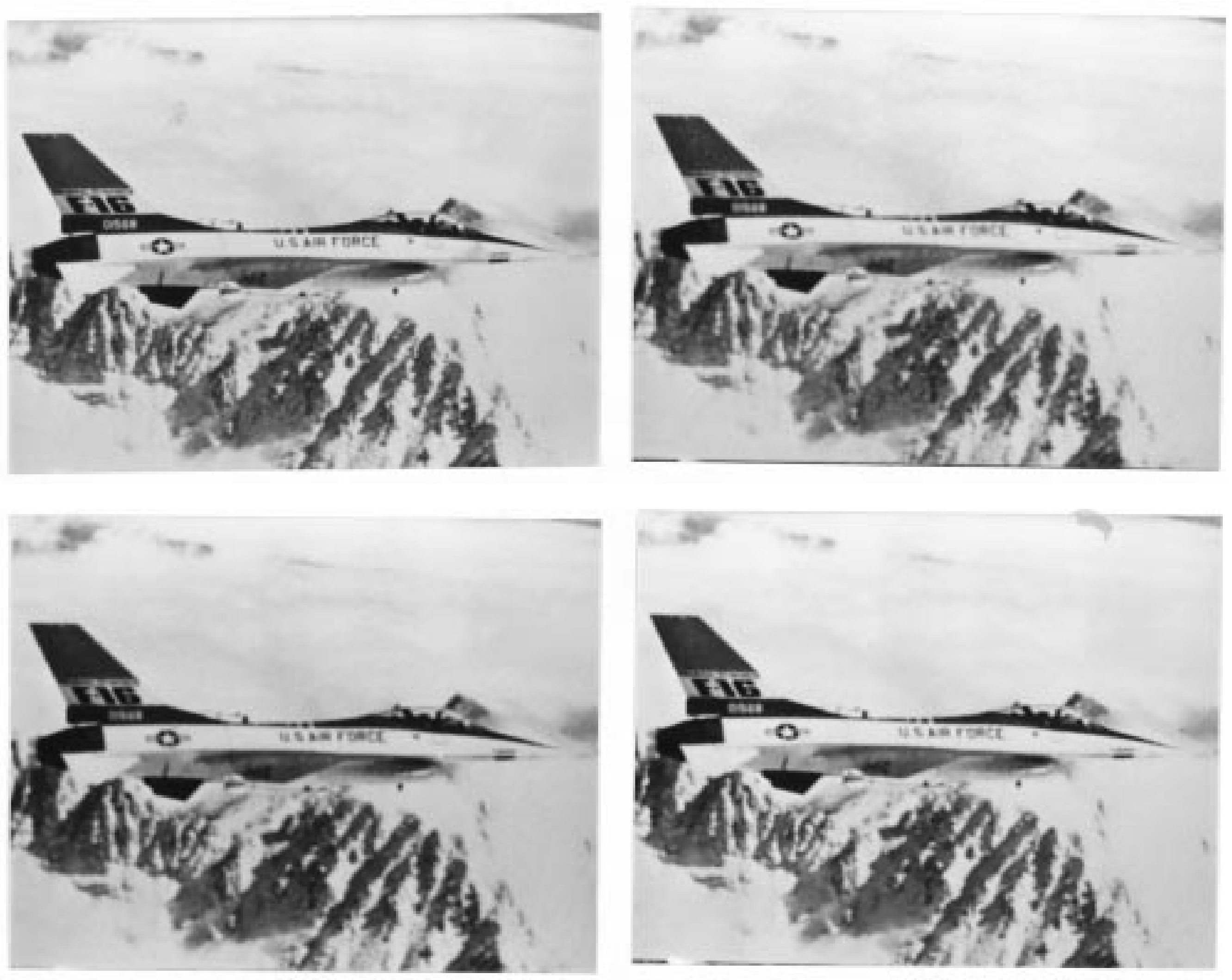


Fig. 3. Reconstructed jet picture using SWF, SWF with postprocessing, and SV-SWF. Top left: original. Top right: reconstructed using SWF. Bottom left: reconstructed using SWF with postprocessing. Bottom right: reconstructed using SV-SWF.

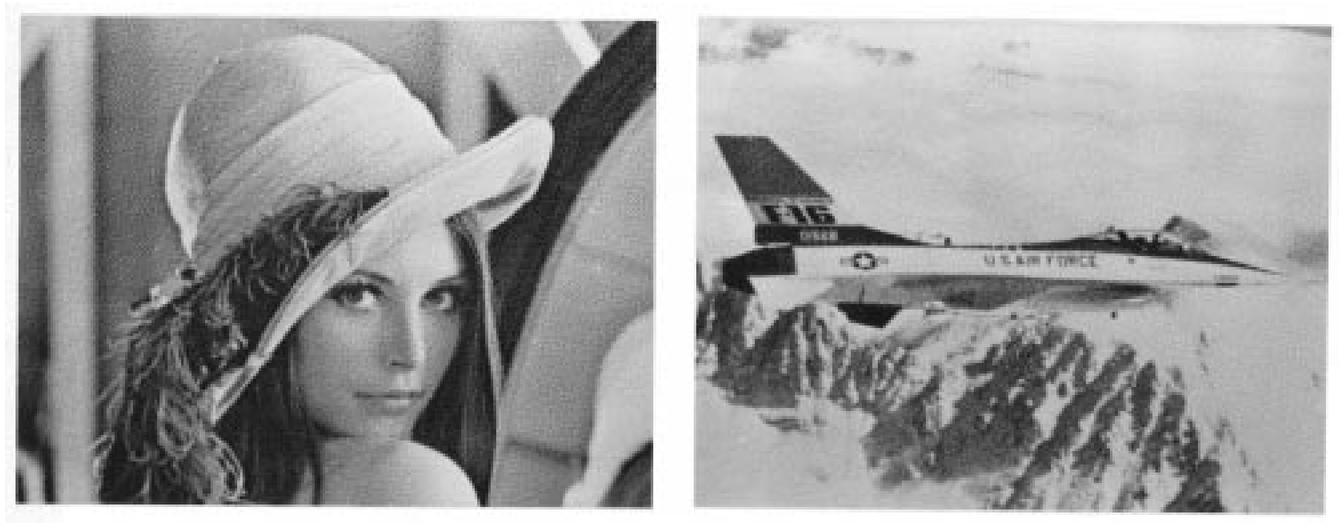


Fig. 4. Halftoned Lena and jet images using error-diffusion method.

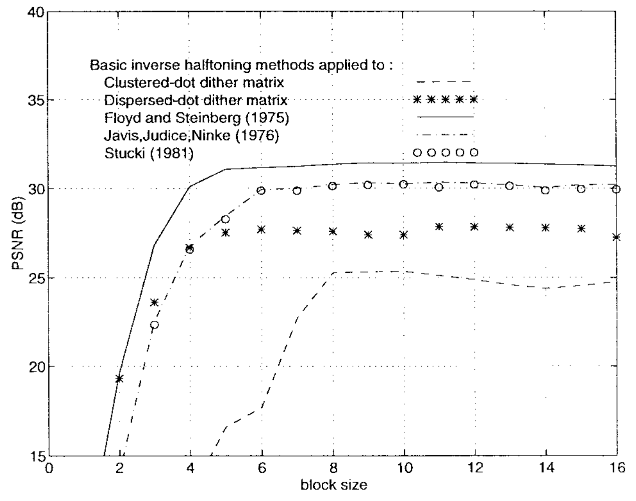
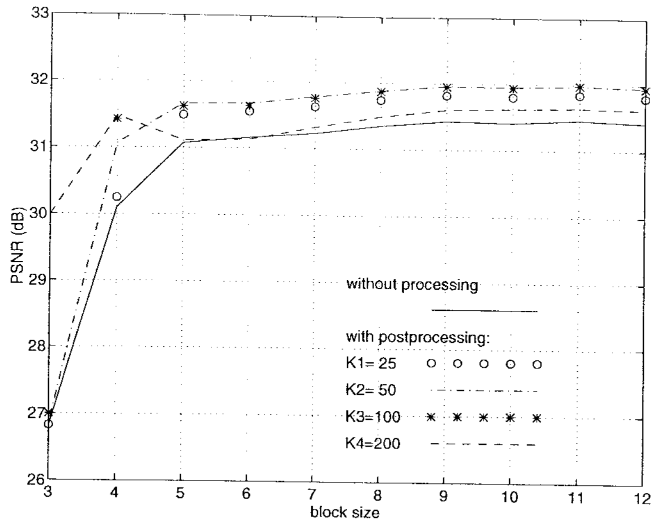
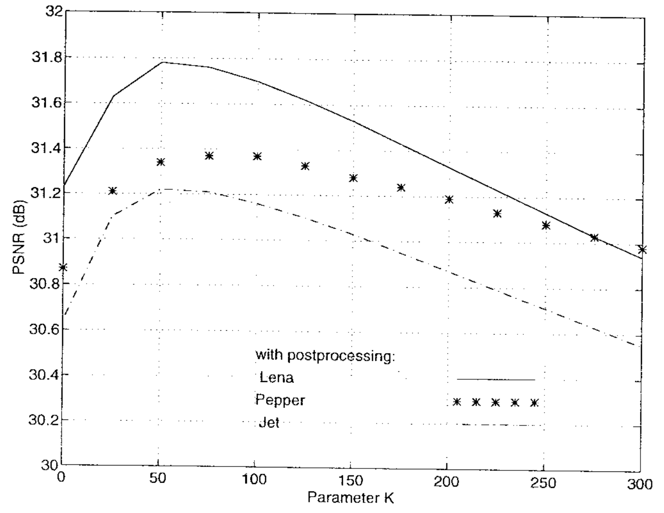


Fig. 5. SWF tested on various halftoning techniques, Lena image.



(a)



(b)

Fig. 7. Postprocessing evaluation. (a) Block size, Lena. (b) K value—using a 7×7 SWF.

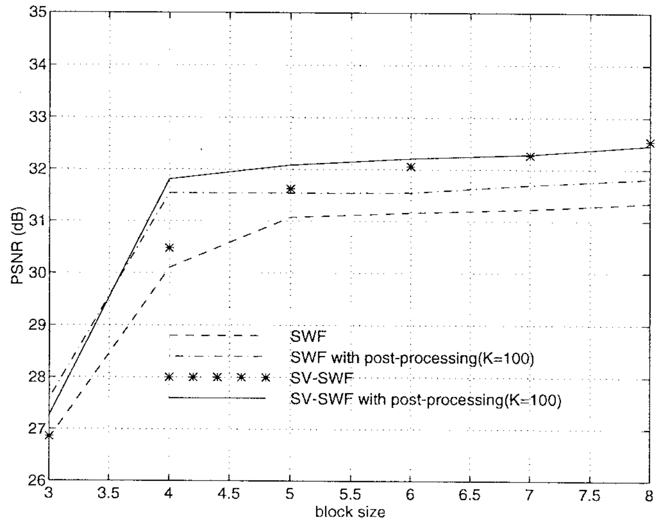


Fig. 6. Performance of different filter sizes and inverse schemes on Lena.

2) Training Phase: The entire inverse halftoning procedure is made of several sliding window filters. In the training phase, we find the best weights in each SWF based on the classified training data. Every pixel in the original image is classified into one of the predesigned categories according to the local variance classification algorithm. And then, the corresponding halftone pixel $b(k, l)$ together with its neighbor pixels inside the filter support and the original image $I(k, l)$ constitute a block of training data. Separate SWF is designed by applying the LMS algorithm (described in Section II) to the associated set of training data blocks.

B. Classification Based on Local Variance

A few classification schemes have been tested. The one we found quite effective and at the same time does not require extensive computation is based on local variance. One consideration worth mentioning here is that we conduct classification on the gray-scale images rather than on the binary images. This is because the gray-scale pixels have more variety and contain more information than the binary pixels of the same window size.

We classify pixels according to its local variance. The number of partitioned groups and the threshold values used to separate them are chosen empirically. It is found sufficient to merely partition data into three groups: low-variant, middle-variant, and high-variant regions. Therefore, two thresholds, ν_1 and ν_2, need to be specified. A larger number of classes provides little gain in PSNR. This is because a finer partition of image local features such as edges with different orientations can not be distinguished using just the first- and second-order statistics. The typical parameters we use are $\nu_1 = 10$ and $\nu_2 = 100$. Compared to the simple SWF reconstruction, the SV-SWF scheme equipped with image classification has roughly a 1 dB gain in PSNR.

V. Simulation Results

We have described postprocessing technique and the adaptive structure of our algorithm. In total, we could have four inverse halftoning schemes: i) SWF, ii) SWF with postprocessing, iii) SV-SWF, and iv) SV-SWF with postprocessing. In this section, we focus on the performance evaluation of these algorithms. Because the fourth algorithm—SV-SWF with postprocessing—does not offer significant improvement over the SV-SWF algorithm, the simulation results we present here are mostly related to the first three algorithms. The issues

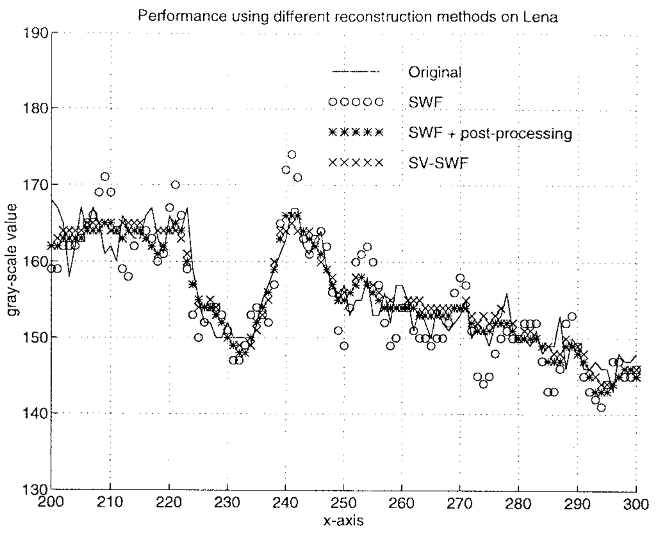


Fig. 8. Scan line of Lena in the smooth regions; coordinates: (200, 450) to (300, 450).

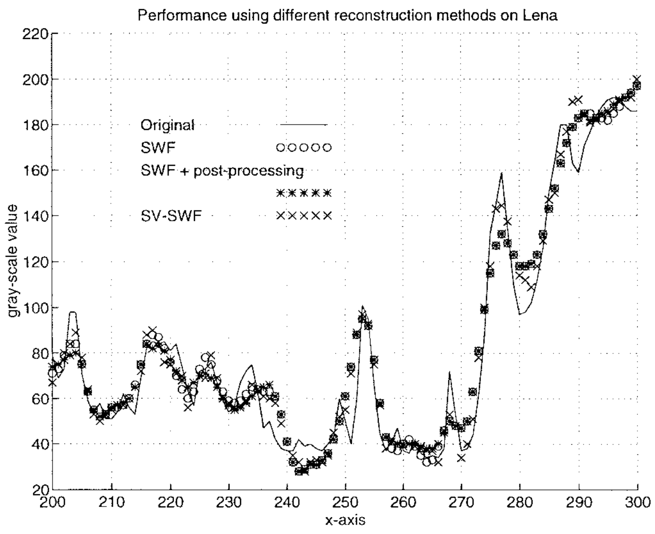


Fig. 9. Scan line of Lena including several edge regions; coordinates: (200, 210) to (300, 210).

to be discussed are robustness, window size, postprocessing, and the reconstructed signal characteristics.

A. Image Quality

Two sets of typical reconstructed images using SWF, SWF with postprocessing, and SV-SWF are shown in Figs. 2 and 3. In these experiments, picture *Lena*, 512 $\times$ 512 and 8 b/pixel, is used as the training picture in designing the filter weights and classification categories. The other two pictures, pepper and jet serve as the *outside* pictures. Limited by the space, only some of the processed pictures of Lena and jet are printed here. It is clear that the contents of these three pictures are quite different. In this and the following experiments, if not noted, all the halftone pictures (Fig. 4) are generated by the error-diffusion method with Floyd–Steinberg kernel [1]. One may notice that Fig. 4 is a photo of a binary image recorded by a firm recorder instead of a printed two-tone image. This is because a photo can faithfully represent a gray-scale image and, thus, it is used to show both the original and the reconstructed images.

All the reconstructed images have rather good subjective quality. With careful examination, small "lumps" are visible on the SWF reconstructed images. This artifact can be largely reduced by postprocessing. The side effect of postprocessing and SV-SWF is that they may sometimes remove fine textures such as the rocky features in the jet picture. Overall, the output pictures of either SWF with postprocessing or SV-SWF are very close to the original gray-scale pictures.

B. Robustness

Owning to the diversity of picture contents and forward halftoning techniques, it is expected that the performance of the proposed algorithms would depend on both picture contents and halftoning techniques. In our experiments, different sets of training data and halftoning methods (except for the clustered-dot dither) often result in only minor variation on the magnitude and shape of the reconstruction filters. Hence, although the SWF weights are training-data dependent, the performance degradation of a properly designed SWF on the outside-training pictures is often acceptable. We first examine the effect of picture contents variation.

Three pictures, Lena, pepper, and jet, are used as training pictures to generate three separate sets of filters (of size 7 $\times$ 7). Then these filters are applied to all the three test pictures. The results (PSNR) are shown in Tables I–III. As indicated by these data, the PSNR decrease (from their own best trained result) is less than 1 dB. The SV-SWF method is particularly less sensitive to picture variation (less than 0.5 dB). Therefore, when the test pictures are generated using the same error diffusion methods, the sliding window filters designed for one picture is adequate for the other pictures for inverse halftoning purpose.

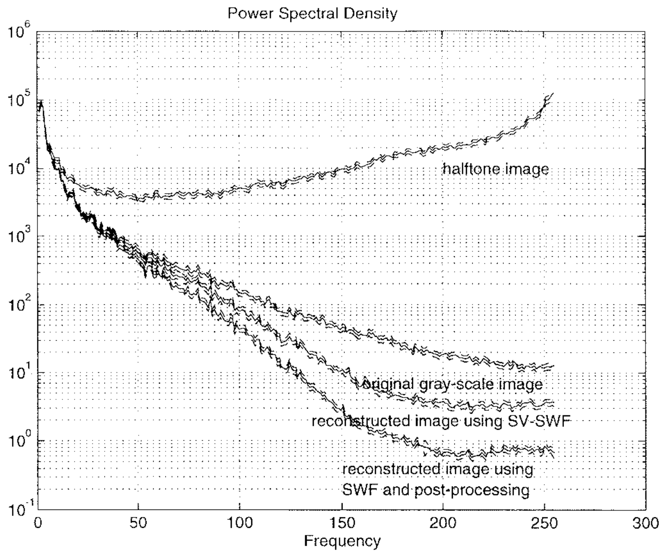


Fig. 10. Power spectral density functions of the original gray-scale image, the halftone image, the SWF reconstructed image with postprocessing, and the SV-SWF reconstructed image of Lena.

We next look into the forward halftoning issue. Lena is used as the test picture. It is halftoned by five different methods: two dither matrices in Fig. 11 and three error diffusion kernels: Floyd–Steinberg (1975); Javis, Judice, and Ninke (1976); and Stucki (1981) [1]. Typically, error-diffusion methods produce better visual quality halftone images (on nearly ideal display monitors). Dither method with dispersed-dot matrix comes next. Clustered-dot dither matrix often produces the poorest subjective image quality because a large amount

43	44	45	46	47	48	49	50
42	21	22	23	24	25	26	51
41	20	7	8	9	10	27	52
40	19	6	1	2	11	28	53
39	18	5	4	3	12	29	54
38	17	16	15	14	13	30	55
37	36	35	34	33	32	31	56
64	63	62	61	60	59	58	57

0	32	8	40	2	34	10	42
48	16	56	24	50	18	58	26
12	44	4	36	14	46	6	38
60	28	42	20	62	30	44	22
3	35	11	43	1	33	9	41
51	19	59	27	49	17	57	25
15	47	7	39	13	45	5	37
63	31	45	23	61	29	43	21

Fig. 11. Ordered dither matrices. Left: 8 × 8 clustered-dot dither matrix. Right: 8 × 8 dispersed-dot dither matrix.

of information is lost due to the strong cluster constraint posed by the dither matrix. Fig. 5 shows the results of SWF designed for the above five halftoning techniques separately. As we expect, the error-diffusion halftoned pictures have the best reconstructed PSNR. Dispersed-dot dither method has a lower but close performance. Clustered-dot dither method is quite a bit lower. Also shown on the same plot is the window (block) size effect. It is particularly important to use a larger (greater than 8 × 8) window size for the clustered-dot dithering.

On the other hand, the filter weights (7 × 7) trained on one halftoning method degrade modestly on the other methods if the clustered-dot method is not counted. Table IV shows the results of five halftoned pictures of Lena processed by the filter weights derived using its own data, the Floyd–Steinberg data, and the dispersed-dot data. Except for the clustered-dot dithering case, the PSNR differences are around 1–2 dB. It is interesting to notice that for the outside halftoning data the postprocessing is able to increase PSNR quite effectively and its PSNR is often slightly larger than that of the SV-SWF scheme. We may, thus, conclude that the pictures synthesized by three error diffusion kernels have rather similar characteristics in terms of their performance in inverse halftoning. Therefore, filters designed for one halftoning kernel works reasonably well for the other. The clustered-dot dithered picture has very different characteristics; hence, the inverse filters designed based on error-diffused pictures perform poorly on it. The dispersed-dot dither method lies between the above two extremes but closer to the error-diffusion method.

TABLE IV
PSNR (dB) OF LENA GENERATED BY VARIOUS HALFTONING TECHNIQUES

Test Images	Training Images	Schemes		
		SWF	SWF+Postproc	SV-SWF
Clustered-dot	Clustered-dot	22.1	22.2	22.4
	Floyd & Steinberg	12.5	12.4	13.3
	Dispersed-dot	15.6	15.6	16.3
Dispersed-dot	Dispersed-dot	27.8	28.1	28.4
	Floyd & Steinberg	26.3	27.1	26.7
Javis et al.	Javis et al.	30.1	30.7	31.0
	Floyd & Steinberg	28.2	29.3	28.5
	Dispersed-dot	28.1	28.7	27.0
Stucki	Stucki	29.9	30.6	30.8
	Floyd & Steinberg	28.4	29.4	28.6
	Dispersed-dot	28.1	28.7	27.2
Floyd & Steinberg	Floyd & Steinberg	31.3	31.8	32.2
	Dispersed-dot	29.1	29.1	29.3

C. Window Size

As indicated by Figs. 5 and 6, the inverse halftoning results depend on the filter window size. For error-diffusion and dispersed-dot halftone images, window size of 5 × 5 or larger is adequate. The window size needs to be larger (8 × 8) for cluster-dot dithered images. Schemes with postprocessing seem to reach their performance plateau at 4 × 4 window size, a value smaller than the other schemes without postprocessing. For the SV-SWF scheme, larger window size increases PSNR slightly; however, a 6 × 6 sliding-window is often close to the performance upper limit. If we like to reduce computational complexity, a 5 × 5 SWF with postprocessing is adequate in producing good quality pictures in most cases.

D. Postprocessing

The adaptive postprocessor described in Section III improves both objective and subjective image quality, especially in the smooth areas. In our experiments, good results are obtained at the (postprocessing) window size around 5 × 5 and the parameter K around 100. A large K value blurs the reconstructed images, whereas a very small K value cannot effectively remove the undesirable "lumps." In addition to the quite noticeable subjective improvement, the proposed postprocessing scheme can increase PSNR up to 1 dB gain, as shown in Fig. 7(a). In this figure, the binary pictures are again produced by the Floyd–Steinberg error diffusion kernel and the gray-scale images are reconstructed by an SWF with 7 × 7 weights trained on Lena.

To see more clearly the effectiveness of postprocessing, two scan lines of the processed Lena are shown in Figs. 8 and 9. Fig. 8 shows a smooth region example, while Fig. 9 shows an edge region example. The proposed adaptive postprocessor can reduce the reconstruction errors generated by the SWF method in the smooth regions and leave the edge region almost untouched to retain the edge sharpness. Also demonstrated in these two figures are the advantages of using the SV-SWF method. It automatically selects adequate reconstruction filters to recover the smooth monotone regions and the high-contrast regions separately. Therefore, postprocessing cannot provide significant further improvement on the SV-SWF reconstructed pictures.

E. Power Spectrum Analysis

In order to explain the motivation behind our adaptive algorithm, we compute the power spectra of the reconstructed images generated by different inverse halftoning methods using one-dimensional periodogram averaging (subroutine *spectrum* in Matlab). Four sets of power spectra are shown in Fig. 10. In each set of the curves, the middle solid line is the average spectrum, the upper and the lower dash lines indicate the 95% confidence interval. Typically, high-frequency noises are injected into the halftone images via the halftoning process. In order to reduce the aforementioned noises, the traditional space-invariant lowpass filter cuts off the frequency components higher than $f_s/4$. This results in seriously blurred pictures due to the lack of high-frequency image components.

The proposed SWF is the optimum linear filter that minimizes the mean squared errors between the original image and the reconstructed image. However, it is chosen for the entire picture. Although the low-frequency response is retained very well, it suppresses excessively the high-frequency components. In the case of SV-SWF, it applies narrowband filters to the smooth regions and wideband filters to the high-contrast regions. Therefore, a certain amount of high-frequency components (in the edge regions) is properly recovered. As a result, we obtain sharper images.

Acknowledgment

The authors are grateful to T.-Y. Chao for generating some of the simulation results and photos.

References

[1] R. Ulichney, *Digital Halftoning.* Cambridge, MA: MIT Press, 1987.
[2] C. M. Miceli and R. J. Parker, "Inverse halftoning," *J. Electron. Imaging*, vol. 1, pp. 143–151, Apr. 1992.
[3] P. W. Wong, "Inverse halftoning and kernel estimation for error diffusion," *IEEE Trans. Image Processing,* vol. 4, pp. 486–498, Apr. 1995.
[4] Y.-T. Kim, R. Arce, and N. Grabowski, "Inverse halftoning using binary permutation filters," *IEEE Trans. Image Processing,* vol. 4, pp. 1296–1310, Sept. 1995.
[5] B. Widrow and S. D. Stern, *Adaptive Signal Processing.* Englewood Cliffs, NJ: Prentice-Hall, 1985.
[6] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications.* New York: Academic, 1990.
[7] J. Lee, "Digital image enhencement and noise filtering by use of local statistics," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. PAMI-2, pp. 165–168, Mar. 1980.