FAST BLIND INVERSE HALFTONING

Niranjan Damera-Venkata, Thomas D. Kite, Mahalakshmi Venkataraman, and *Brian L. Evans*

Laboratory for Image and Video Engineering
Department of Electrical and Computer Engineering
The University of Texas at Austin, Austin TX 78712-1084, U.S.A.
{damera,tom,maha,bevans}@vision.ece.utexas.edu

ABSTRACT

We present a fast, non-iterative technique for producing grayscale images from error diffused and dithered halftones. The first stage of the algorithm consists of a Gaussian filter and a median filter, while the second stage consists of a bandpass filter, a thresholding operation, and a median filter. The second stage enhances the rendering of edges in the inverse halftone. We compare our algorithm to the best reported statistical smoothing, wavelet, and Bayesian algorithms to show that it delivers comparable PSNR and subjective quality at a fraction of the computation and memory requirements. For error diffused halftones, our technique is seven times faster than the MAP estimation method and 75 times faster than the wavelet method. For dithered halftones, our technique is 200 times faster than the MAP estimation method. A C implementation of the algorithm is available at `http://www.ece.utexas.edu/~bevans/projects/inverseHalftoning.html`.

1. INTRODUCTION

Halftoning converts a grayscale image to a binary image so that the binary image looks like the original when viewed from a distance. Blurring a halftone with a Gaussian lowpass filter is one way to recover a grayscale image. Lowpass filtering reduces the quantization noise incurred in halftoning, but also destroys high frequency edge and texture information. Blurring belongs to a class of algorithms that treats inverse halftoning as a denoising problem. The class includes statistical smoothing [1] and a wavelet approach [2]. The wavelet scheme, which reports the best PSNR results for error diffused halftones, is a single-pass algorithm that does not assume knowledge of the halftoning kernel. It uses at least 17 large filters and requires 9 floating-point copies of the image to be stored in memory; it is therefore impractical for use in low-cost devices.

A second class of algorithms, which includes the kernel estimation method [1], uses a filtering approach to obtain an initial estimate of the inverse halftone. It then successively refines this estimate with projections using an estimated halftoning kernel, obtained with a modified LMS algorithm. Application of MAP projections along with wavelet denoising [2] is another technique that falls into this category.

This research was supported by an NSF CAREER Award under Grant MIP-9702707 and by a grant from HP Laboratories, Palo Alto, CA. LIVE Web: `http://anchovy.ece.utexas.edu`.

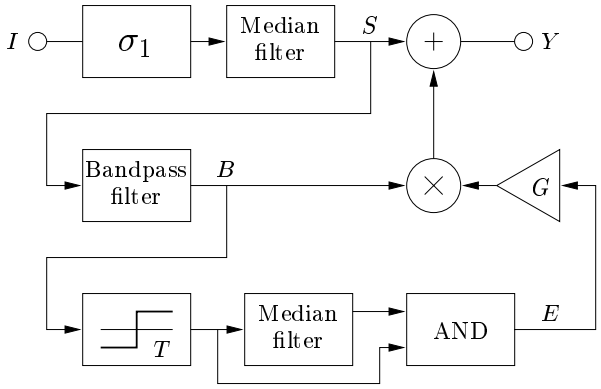


Figure 1: Block diagram of the proposed algorithm. The block labeled σ_1 is a Gaussian lowpass filter.

These techniques are also computationally intensive, and unsuitable for fast, low-cost implementations.

A third class of algorithms, which includes projection onto convex sets (POCS) [3], MAP estimation [4], and set theoretic approaches [5], formulates inverse halftoning as an inverse problem. This class assumes knowledge of the halftoning kernel, which is not always reasonable, and implements iterative non-linear constrained optimization to solve the problem, which is computationally costly.

We present a fast algorithm that incorporates high frequency information from the halftone into the inverse halftone. The algorithm employs entirely local operations, and applies linear and non-linear filtering followed by non-linear edge enhancement and noise suppression. For error diffused halftones, the algorithm produces results of comparable visual quality to iterative schemes [1, 3, 4]. Unlike these schemes, our algorithm produces high quality inverse halftones from dithered halftones. We compare the inverse halftones produced by our algorithm with those produced by wavelet [2] and Bayesian [4] techniques.

2. ERROR DIFFUSED HALFTONES

Figure 1 shows the block diagram of the proposed inverse halftoning scheme. The first stage obtains a smooth estimate S of the original image I by linear filtering followed by non-linear smoothing. The linear filter is a separable 9×9

Algorithm, Reference	PSNR (dB)		Time (min)
	lena	*peppers*	
Wong [1]	31.00	29.30	-
Wavelet[2]	31.47	30.43	15
Proposed	31.51	31.17	0.20

Table 1: PSNR and timing comparison of inverse halftoning algorithms for error diffused halftones.

Gaussian lowpass filter whose variance depends on the type of halftone. For error diffused halftones, we choose $\sigma_1^2 = 1.4$. The output of this filter is quite smooth, but contains some small-scale noise. To reduce this noise without blurring the edges, we apply a 3×3 graylevel median filter to the output of the linear filter.

Because of smoothing by the Gaussian filter, S has a blurred appearance. The second stage enhances the edges in S without increasing the small-scale noise. It employs a fixed 13×13 bandpass filter to estimate the magnitude of the local image gradient from S and suppress small-scale noise. We threshold the bandpass filter output B with a global integer threshold $T \in \{0, 1, 2, 3\}$ to obtain a binary edge map. Because of the presence of noise in B, the binary edge map may contain isolated pixels that have been incorrectly identified as edges. We logically AND the edge map with a 5×5 binary median filtered version of itself to compute a refined edge map E. That is, we retain an edge pixel if at least half of its neighbors have also been identified as edge pixels. Figure 2(a) shows the noisy edge map computed from the error diffused *lena* image, while Figure 2(b) shows the refined map. The non-linear filtering eliminates most of the small-scale noise in the refined edge map.

At pixels where the edge map E is not zero, we add to the smoothed image S a scaled version of the corresponding pixel from the bandpass image B. This process inserts edge information into S to produce the inverse halftone

$$Y(i,j) = S(i,j) + G\,B(i,j) \quad \forall (i,j) \text{ s.t. } E(i,j) = 1\ , \quad (1)$$

where G is an integer scaling factor, $G \in \{1, 2, 3, 4, 5, 6\}$. The free parameters are the gain G, which determines the level of edge enhancement, and the threshold T, which determines the degree of denoising. For this paper, we used $T = 0$ and $G = 4$ for error diffused halftones. These values gave good results on all the test images.

Figure 3 compares the inverse halftoned *lena* and *peppers* images obtained by the proposed method with those obtained by the wavelet technique [2]. The images produced by our algorithm are of comparable visual quality with those produced using wavelet denoising. Table 1 lists PSNR results and execution times for the proposed scheme, the wavelet scheme, and an iterative method that uses halfband filtering and statistical smoothing [1]. We assume that the error diffusion kernel is unknown. For the *lena* image, the proposed method not only yields a higher PSNR, but runs at 75 times the speed of the wavelet scheme. The large increase in PSNR for the *peppers* image is due in part to a fault in the original image, which was reported to the authors of [2], and was corrected for this work.

Algorithm, Reference	PSNR (dB)		Time (min)
	Clustered	Dispersed	
Bayesian [4]	26.3	27.2	60
Proposed	25.6	27.6	0.30

Table 2: PSNR and timing comparison of Bayesian and proposed methods for dithered *lena* halftones.

For 512×512 error diffused halftones, our algorithm executes in 2.4 seconds on a 167 MHz Sun Ultra-2 workstation, and in 12 seconds on an RS/6000 workstation (a 22-MFLOP machine). On an RS/6000 workstation, the Bayesian method executes in 90 seconds and the wavelet scheme takes 15 minutes. For the *lena* image, our algorithm obtains a PSNR that exceeds the best reported blind result [2], and has the shortest reported execution time. Table 3 compares the complexity, memory usage, and objective performance of the proposed method with other reported blind inverse halftoning results.

3. DITHERED HALFTONES

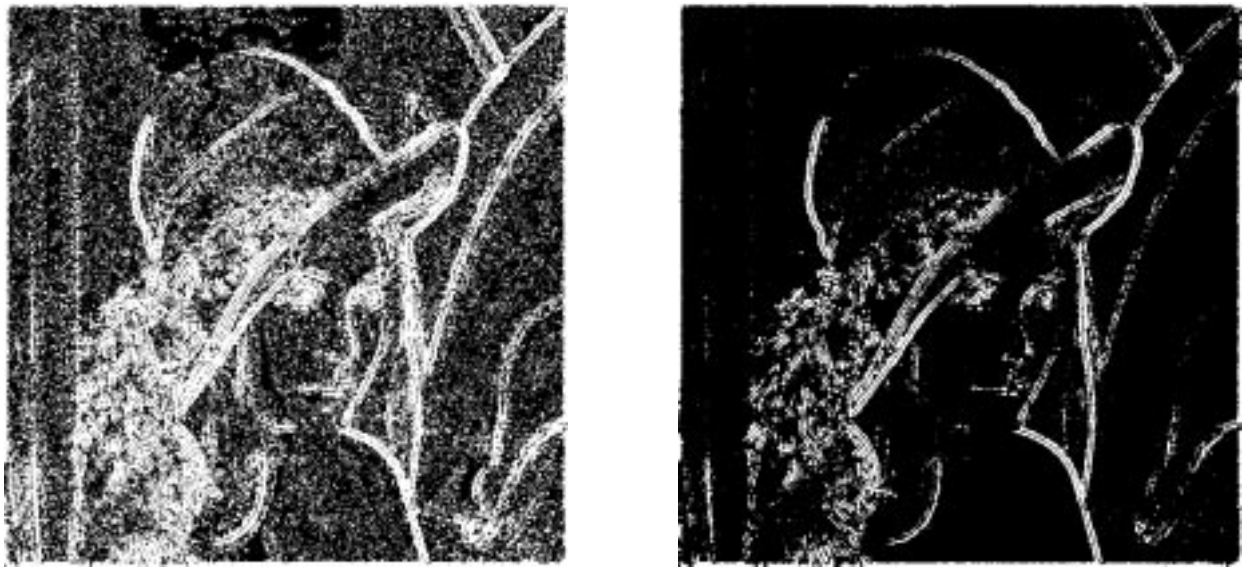
Because of the large variation in screen sizes, we cannot yet claim blind inverse halftoning from dithered halftones. The results we quote here are restricted to the screens used in [4]. For dithered halftones, the structure of the algorithm remains the same as in Figure 1. The Gaussian filter remains 9×9, the bandpass filter becomes 17×17, and grayscale median filter becomes 5×5. We choose $\sigma_1^2 = 2.5$ for 8×8 dispersed-dot dither and $\sigma_1^2 = 8$ for 4×4 clustered-dot dither. For other screen sizes, an estimation procedure may be used to determine the value of σ_1^2 that gives optimum performance.

Figure 4 compares *lena* images obtained by the proposed and Bayesian [4] methods from clustered-dot halftones. Figure 5 shows the results obtained from dispersed-dot halftones. For dispersed-dot halftones, the proposed algorithm produces a sharper result than the Bayesian method. For clustered-dot halftones, the performance of the Bayesian method is better on fine detail, and comparable elsewhere. This disparity is offset by the fact that the proposed scheme is 200 times faster. Table 2 compares PSNRs and execution times of the two schemes for dithered halftones. The PSNRs of the inverse halftones are comparable.

For a 512×512 dithered halftone, our algorithm executes in 3.2 seconds on a 167 MHz Sun Ultra-2 workstation, and in 18 seconds on an RS/6000 workstation. On an RS/6000 workstation, the Bayesian method requires one hour. Table 3 compares the complexity, memory usage, and objective performance of the proposed method with other reported inverse halftoning results.

4. IMPLEMENTATION

All of the linear filters are separable, linear phase, and have integer coefficients. The number of multiplications is drastically reduced over a non-separable implementation. Since a halftone is binary, the first separable filtering operation



(a) Raw edge map. (b) Refined edge map.

Figure 2: Refinement of the raw edge map using non-linear filtering.

is performed with only additions. All linear filtering operations, except the scaling of filter outputs, are implemented with integer arithmetic. The grayscale median filter is implemented using a linear-time algorithm based on a selection sort using partitioning [7]. The binary processing in the second stage consists primarily of pointwise operations. The binary median filter is implemented using comparisons and increments. Only 28 image rows need to be stored.

Because the second stage is implemented with binary operations, it becomes possible to allow the user to adjust T and G interactively to obtain the best visual result for the halftone in question. To implement this feature, we store the pre-computed S and B images and merely re-evaluate (1) as the user adjusts T and G.

5. SUMMARY

We present an efficient blind inverse halftoning algorithm which yields results that are visually comparable to the best iterative techniques, at a fraction of the computational cost and memory usage. Our algorithm is applicable to error diffused and dithered halftones. Since the algorithm uses only local operations and integer arithmetic, it is well-suited to implementation in embedded hardware and software.

Algorithm, Reference	Memory usage	Complexity	PSNR (dB)	
			lena	*peppers*
POCS [3]	$8N^2$	High	30.4	–
Bayesian [4]	$8N^2$	High	–	–
Wong [1]	$8N^2$	Medium	31.0	29.3
Wavelet [2]	$36N^2$	Medium	31.5	30.4
Proposed	$28N$	Very Low	31.5	31.2

Table 3: Comparison of inverse halftoning schemes for images of size $N \times N$ pixels. Memory requirements are estimated in bytes. "Very low" denotes fewer than 250 operations per pixel, "medium" denotes 500–2000 operations per pixel, and "high" denotes more than 2000 operations per pixel. PSNR figures are from publications, where available.

6. REFERENCES

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

[2] Z. Xiong, M. T. Orchard and K. Ramchandran, "Inverse halftoning using wavelets", *Proc. IEEE Int. Conf. Image Proc.*, vol. 1, pp. 569–572, Sep. 1996.

[3] S. Hein and A. Zakhor, "Halftone to continuous-tone conversion of error-diffusion coded images", *IEEE Trans. Image Proc.*, vol. 4, no. 2, pp. 208–216, 1995.

[4] R. L. Stevenson, "Inverse halftoning via MAP estimation", *IEEE Trans. Image Proc.*, vol. 6, no. 4, pp. 574–583, Apr. 1997.

[5] N. T. Thao, "Set theoretic inverse halftoning", *Proc. IEEE Int. Conf. Image Proc.*, vol. 1, pp. 783–786, Oct. 1997.

[6] T. D. Kite, N. Damera-Venkata, B. L. Evans and A. C. Bovik, "A high quality, fast inverse halftoning algorithm for error diffused halftones", *Proc. IEEE Int. Conf. Image Proc.*, Oct. 1998.

[7] D. E. Knuth, *The Art of Computer Programming: Fundamental Algorithms*, vol. I. Reading, MA: Addison-Wesley, 1968.

(a) Proposed method.

(b) Wavelet method.

(c) Proposed method.

(d) Wavelet method.

Figure 3: Inverse halftones obtained from error diffused halftones.

(a) Proposed method.

(b) Bayesian method.

Figure 4: Inverse halftones obtained from dispersed-dot dithered halftones.

(a) Proposed method.

(b) Bayesian method.

Figure 5: Inverse halftones obtained from clustered-dot dithered halftones.