Fast Inverse Halftoning Algorithm for Ordered Dithered Images

Pedro Garcia Freitas, Mylène C.Q. Farias, and Aletéia P. F. de Araújo
Department of Computer Science,
University of Brasília (UnB),
Brasília, Brazil
Email: sawp@sawp.com.br, mylene@ieee.org, and aleteia@cic.unb.br

***Abstract*—In this paper, we present a simple and fast inverse halftoning algorithm, targeted at reconstructing halftoned images generated using dispersed-dot ordered dithering algorithms. The proposed algorithm uses a simple set of linear filters combined with a stochastic model in order to predict the best intensity values for the binary image pixels. The algorithm produces images with a better perceptual quality than the available algorithms in the literature, preserving most of the fine details of the original gray-level image. It has a high performance, which can be further improved with the use of parallelization techniques.**

***Keywords*-halftoning, inverse halftoning, dispersed-dot ordered dithering, dithering.**

I. Introduction

Halftoning is the technique of converting continuous-tone images into binary images using patterns of white and black dots. This technique is useful for creating the illusion of seeing multiple intensity levels in a binary image, which makes it suitable for applications where a reduced number of levels is needed, such as newspapers, fax machines, and document printing processes. Halftoning algorithms can be roughly classified as *dithering* or *error diffusion* [1].

Dithering algorithms generate halftoned images by comparing the pixel intensity values of the original image with threshold scalar values or matrices, which can be generated by various methods. Error diffusion algorithms compare the pixel intensity values with a fixed threshold. The resulting error between the output value and the original value is distributed to neighboring pixels according to predefined weights. Both the ordered dithering and the dithered with error diffusion have advantages and disadvantages for specific applications. The choice of algorithm often depends on the application with which it is associated.

Inverse halftoning is the technique that allows the restoration of the original information of an image (with multiple levels of intensity) from a binary (black-and-white) representation of it. This technique can be used in several applications when the only available version of the image is a binary one. For example, it is well known that dithered images suffer great degradations when subject to simple operations, such as filtering, decimation, interpolation, sharpening, etc. In these applications, it is necessary to use inverse halftoning techniques to convert the binary images into gray-level images and, then, apply the desired operation.

Different solutions have been proposed to solve the inverse halftoning problem, such as iterative projection [2], neural networks [3], vector quantization [4], lookup table [5], [6], wavelet estimation [7][8], and least square methods [9]. Although the above techniques produce satisfactory results for some cases, they all have a high computational cost, which is mainly due to their high mathematical complexity. Among the algorithms in the literature, the work by Damera-Venkata *et al.* is one of the most computationally efficient algorithms [10][11].

A more recent and non-conventional application of inverse halftoning is in error concealment or recovery [12] and data hiding. For example, in the work by Adsumilli *et al.* a dithered version of the original video is embedded into the original itself using a spread-spectrum watermarking technique. If parts of this original are lost in the transmission or compression stages, the receiver can extract the corresponding dithered version of the original from the received frame. Then, an inverse halftoning technique is used to obtain an approximation of the lost data of the frame.

In this paper, we present a simple and fast inverse halftoning algorithm, targeted at reconstructing halftoned images generated using dispersed-dot ordered dithering algorithms. The proposed algorithm uses a simple set of linear filters combined with a stochastic model to predict the corresponding intensity values of the binary image pixels. Our goal is to obtain the best approximation of the original image with the lowest computational cost. Our target application is the recovery of lost information (e.g. error concealment) in real-time processing applications, such as video decoding.

This paper is organized as follows. In Section II, we present a description of the dithering algorithm considered in this work. In Section III, we describe the details of the proposed inverse halftoning algorithm. In Section IV, we present the results and, finally, in Section V, we give our conclusions.

II. Ordered Dithering

In this work, we focus on *ordered dithering* algorithms. These algorithms have the characteristic of generating halftoned images with sets of pixel clusters that have a predictable pattern. This, in turn, generates binary images that

have an adequate redundancy that allows successful inverse halftoning operations. Given our target applications, generating a more easily predictable image is very important, and has a significant impact on the performance of the technique.

The ordered dithering technique can be classified into two types: dispersed-dot ordered dithering and clustered-dot ordered dithering. In the first type, the less relevant dots for the representation (e.g., white pixels in a dark area) are spread out in an apparently random order, but the density is varied in order to match the gray-level intensity of the original image. In the second type, geometric shapes of different sizes are used to create a pattern that is proportional to the gray-level intensity of the original image. That is, the pixels of the binary image are clustered to create a visual sensation of different tones.

Fig. 1 shows examples of these two types of dithering techniques. In Fig. 1, an image with decreasing gray-level intensities (top) is depicted, along with dithered versions of this image obtained using the dispersed-dot (middle) and clustered-dot dithering algorithms (bottom).

The dots pattern model used in this work is the dispersed dithering. The ten 3×3-pixels dot patterns used to generate the dithered images are depicted in Fig. 2. These patterns were generated using the following Bayer Matrix (M_B) [13]:

$$M_B = \begin{bmatrix} 6 & 8 & 4 \\ 1 & 0 & 3 \\ 5 & 2 & 7 \end{bmatrix}.$$

To generate the dithered image according to this model, we first quantize the image intensity levels using ten intervals shown in Table I. The image with quantized levels, $I_{quantized}$, is obtained from the original image, $I_{original}$, using the following expression:

$$I_{quantized} = \left\lceil \frac{10}{255} I_{original} \right\rceil.$$

Then, we filter the quantized image using the Bayer matrix. The values of each cell of the matrix are used as thresholds. If the normalized output values corresponding to the pixel

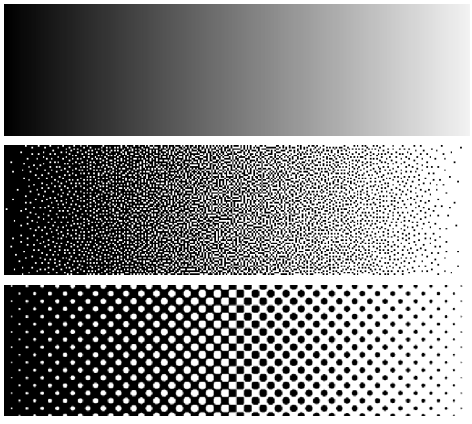

Fig. 1. Original gray-level image (top) and dithered images generated using dispersed-dot (middle) and clustered-dot (bottom) ordered algorithms.

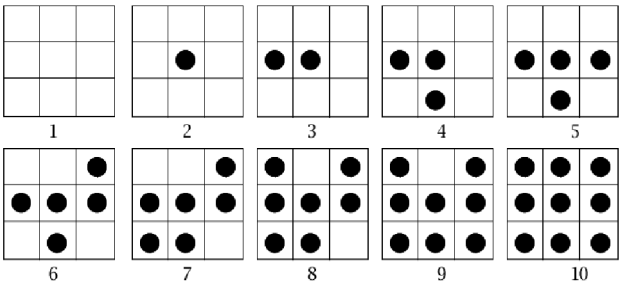

Fig. 2. Dot patterns corresponding to each level of quantization.

are smaller than the number in the matrix cell, the pixel will be substituted by a black value. Otherwise, if the values are greater than the number in matrix cell, the pixel will be replaced by a white value. In other words, the intervals are mapped into one of the patterns shown in Fig. 2.

The algorithm for generating the dithered image from the original 8-bit image is presented Algorithm 1. The resulting dithered version of the image Lena is shown in Fig. 3.

Fig. 3. Halftoned Lena image using Dispersed-dot Ordered Dithering technique.

TABLE I
MAPPED INTERVALS

Interval	Level
$[0, 25.5)$	1
$[25.5, 51)$	2
$[51, 76.5)$	3
$[76.5, 102)$	4
$[102, 127.5)$	5
$[127.5, 153)$	6
$[153, 178.5)$	7
$[178.5, 204)$	8
$[204, 229.5)$	9
$[229.5, 255]$	10

Algorithm 1 Produce 1-bit dithered image from 8-bit image

Input: 8-bit gray-scale input image I_{gray}
Output: 1-bit binary output image I_{dither}
1: Generate $pattern$ as a dictionary data structure that maps a level-value to a dot-pattern, as shown in Fig. 2.
2: Filter the image I_{gray} with a high-pass unsharp filter to obtain $I_{unsharp}$
3: Define a new image $I_{quantized} = \lceil \frac{10}{255} I_{unsharp} \rceil$, which is $I_{unsharp}$ quantized with 10 gray levels.
4: **for all** pixel $p \in I_{quantized}$ **do**
5: $\quad level = I_{quantized}[p]$
6: $\quad I_{dither}[p] = pattern[level]$
7: **end for**

III. Proposed Fast Inverse Halftoning Algorithm

Inverse Halftoning is the process of finding the most appropriate value in a gray-level interval to represent a black-and-white pixel of a dithered image, i.e. it is basically the inverse of the process used for obtaining the dithered image discussed in the previous section. Let us define $M(p)$ as a spatial mask of size 3×3 surrrounding a pixel p of the image, I_{gray} as the original gray-level, I_{dither} as the the dithered image, and $D(p)$ as the distribution of the area surrounding the pixel p. To reconstruct an 8-bit pixel image from an 1-bit pixel halftoned image, we first calculate the local distribution $D(p)$ for all pixels in I_{dither}. Then, we find the most probable mapped intervals corresponding to the original pixel intensity levels in I_{gray}. Once this interval is found, we randomly select a value within it, following a *random walk on fluctuating lattice model* [14].

In order to randomly select the best value in an interval, the random walk model uses a *renormalization group* transformation, which is a transformation of the space of spatially connected pixels. In this work, we chose to use a simple model with only spatial dependencies for the renormalization transformation. With this model, the transformation rescales the value of a particular pixel depending on the levels (see Table I) of the pixels in the surrounding 3×3 area. More specifically, for a given pixel p, we must first find the most frequent level value in the neighborhood. If this value is the same as the level of p, or it is not in the immediate vicinity, we choose a random value in the level interval. If the most frequent level is higher than the level of p, we set the upper-bound as mid-value of the upper interval and the lower-bound as the current value of the pixel. If the most frequent is lower than the level of p, we set the upper-bound as the current value of the pixel and lower-bound as the mid-value of the lower interval. From these new bounds, we choose a new random value for pixel.

The result of applying this inverse halftoning technique to the dithered version of the image Lena (see Fig. 3) is shown in Fig. 4(b). For comparison, the original Lena image is shown in Fig. 4(a). As expected, although a consistent gray-level image was obtained, lots of false-contours defects are present as a consequence of the quantization of the intensity levels.

To mitigate the problem of false contours, we apply an unsharp filter before the dithering process. The unsharp filter is used to highlight the image details and, therefore, to avoid the loss of important information. Then, we classify the regions of the image as 'smooth' or 'active' using a simple standard deviation measure over a 3×3 pixel area. If the region has a low value of standard deviation, it is replaced by its average. Otherwise, it remains unchanged. Using this small modification, we obtain a much better result. The result of applying this technique for the Lena dithered image is depicted in Fig. 4(c).

Fig. 4. Inverse Halftoning and Filtering Effects: (a) Original, (b) Random Walk, (c) Random Walk with unsharp masking, and (d) Random walk with unsharp masking and Gaussian filtering.

As can be observed in Fig. 4(c), unfortunately, the random choice of gray level intensity produces a fair amount of visible noise in the reconstructed image. We can smooth it out by applying a low-pass filter after the reconstruction process. We use a Gaussian blurring filter, generating the image in Fig. 4(d). As can be seen in this image, excess noise was greatly reduced in this final image, without affecting the image details. The proposed algorithm for inverse halftoning is presented Algorithm 2.

To find better filter parameters for the proposed algorithm, we conducted a set of tests, which basically consisted of choosing the optimal filter configurations to maximize the PSNR values for the resulting reconstructed images. The spatial masks that offered the results for the unsharp masking and Gaussian filters are:

$$M_{unsharp} = \begin{bmatrix} -0.489 & -0.022 & -0.489 \\ -0.022 & 3.044 & -0.022 \\ -0.489 & -0.022 & -0.489 \end{bmatrix}$$

and

$$M_{Gaussian} = \begin{bmatrix} 0.052 & 0.124 & 0.052 \\ 0.124 & 0.297 & 0.124 \\ 0.052 & 0.124 & 0.052 \end{bmatrix}.$$

Algorithm 2 Reconstruct 8-bit image from 1-bit image

Input: 1-bit binary input image I_{dither}.
Output: 8-bit gray-scale output image I_{gray}.

1: Generate $inversePattern$ as a dictionary data structure that maps a 3×3 dot-pattern to a level-value, as shown in Fig. 2.
2: **for all** pixel $p \in I_{dither}$ **do**
3: Define as $mask$ the 3×3 matrix populated with values of pixels around $p = I_{dither}(x, y)$

$$mask = \begin{bmatrix} p(x-1, y-1) & p(x-1, y) & p(x-1, y+1) \\ p(x, y-1) & p(x, y) & p(x, y+1) \\ p(x+1, y-1) & p(x+1, y) & p(x+1, y+1) \end{bmatrix}$$

4: Use $mask$ as key in $inversePattern$, obtaining the correspondent $level$

$$level = inversePattern[mask]$$

5: Calculate the level corresponding to pixel p

$$I_{levels}[p] = level$$

6: **end for**
7: **for all** pixel $p \in I_{dither}$ **do**
8: Get $level = I_{levels}[p]$
9: With $level$, set min and max as the lower and upper limits of range mapped by $level$, as illustrated in Table I.
10: Choose a random value between min and max, storing the result in a new variable $grayvalue$.
11: Create a matrix v containing the pixels in 3×3 neighborhood around p.
12: Call the $renormalization(v)$ procedure and store the result in $I_{reconstructed}[p]$ (random walk model)

$$I_{reconstructed}[p] = renormalization(v)$$

13: **end for**
14: Filter the $I_{reconstructed}$ with a gaussian low-pass filter, save the result in I_{gray} and return.

$$I_{gray} = gaussianLowPassFilter(I_{reconstructed})$$

IV. Experimental Results

We tested the proposed algorithms using a set of gray-level images. The results obtained for the images 'Rose', 'Einstein', 'Chester Cathedral', 'Paper machine', 'Bear', 'Hurricane', 'Peppers' and 'Pills' are depicted in Figs. 5 and 6. In both figures, the first column shows the original gray level images, the middle column shows the dithered version, and the last column shows the reconstructed image using the proposed algorithm.

The images chosen to test the proposed algorithm have a high level of details, what represents a big challenge for inverse dithering algorithms. Notice that the proposed algorithm is able to recover the details of the original image, even in high contrast areas (see 'Rose' and 'Bear' images). It is also able to recover the large variations in luminance (see 'Paper machine' image). The algorithm does add some graininess to the uniform areas. This is a characteristic of most sharpness enhancement algorithms, including the sharpness algorithm used in the proposed technique used compensate for the blurring effect of the dithering.

We also compared the algorithm proposed in this paper with other inverse halftoning algorithms. The following state-of-the-art algorithms were considered for comparison: *Fast Blind Inverse Halftoning* (FBIH) [10] and *Wavelet-based Inverse Halftoning via Deconvolution* (WinHD) [7]. Fig. 7 shows the reconstruction images using the proposed algorithm and these two reference algorithms. We can observe from this figure that the image reconstructed using our approach preserves much more details than the images reconstructed using the other two methods.

In our simulations, we also used three metrics for estimating the objective quality of the reconstructed images: Peak signal-to-noise ratio (PSNR), Universal Image Quality Index (UIQI) [15] and Structural similarity (SSIM) [16].

The results obtained with PSNR are listed in Table II. From the data, we can observe that the proposed method has significantly lower PSNR scores than the other two algorithms. As the results in Fig. 7 show, the low PSNR values listed in the Table II do not necessarily represent a lower visual quality, but only bigger differences in relation to the original. PSNR is a fidelity metric that simply estimates error differences between original and test images, not being able to identify if these differences cause an improvement or a degradation in quality. Over the years, PSNR have been widely criticized for not correlating with quality as perceived by human viewers [17].

TABLE II
Results using Peak signal-to-noise ratio (PSNR)

Image	FBIH	WinHD	Proposed
Cameraman	24.7800	31.1112	24.2410
Galaxy	33.6986	35.1189	25.6677
Peppers	27.4723	31.0780	24.8490
Chester cathedral	20.9075	23.5957	22.1560
Hurricane	26.8154	26.0245	25.4556
Pills	28.0535	31.3776	25.7641
Dollar	22.1636	21.4012	24.2334
Lena	29.5776	32.4471	25.3655
Bear	27.0785	30.5968	25.9224
Einstein	31.3399	33.0679	24.2774
Paper machine	27.6452	29.9093	25.4409
Rose	31.0464	30.2284	29.6955

Table III shows the results using the metric UIQI as a comparison criteria. In this case, we observed a strong fluctuation, making the results obtained with UIQI inconclusive.

Table IV shows the results obtained with the quality metric SSIM. The SSIM metric is one of the most reliable quality metrics available in the literature today, being able to give a better estimate of quality instead of only measuring the error differences between a pair of images [17]. When compared to the WinHD algorithm, a very computationally intense algorithm, the proposed method presents better results in around 60% of the tested images. When compared to the FBIH algorithm, the proposed method presents better results in all tested images. This is in agreement with the visual (perceptual) results presented in Fig. 7. In particular, notice

that the proposed method performed very well (according to SSIM) for images with a high level of detail, such as 'Rose', 'Pills' and 'Papermachine'.

TABLE III
RESULTS USING UNIVERSAL IMAGE QUALITY INDEX (UIQI)

Image	FBIH	WinHD	Proposed
Cameraman	0.9413	0.9675	0.9978
Galaxy	0.9993	0.9992	0.9290
Peppers	1.0003	0.9945	0.9688
Chester cathedral	0.99867	1.0006	0.9915
Hurricane	0.9840	0.9822	0.8949
Pills	0.9976	1.0054	0.9742
Dollar	0.9973	0.9981	0.9914
Lena	0.9989	0.9999	0.9804
Bear	0.9158	0.9427	0.9092
Einstein	0.9998	0.9999	0.9875
Paper machine	0.9710	0.9995	0.9596
Rose	1.0513	0.6299	0.5358

TABLE IV
RESULTS USING STRUCTURAL SIMILARITY (SSIM)

Image	FBIH	WinHD	Proposed
Cameraman	0.7375	0.9139	0.8599
Galaxy	0.8640	0.9191	0.9076
Peppers	0.7894	0.8469	0.8590
Chester cathedral	0.6648	0.8337	0.8352
Hurricane	0.7451	0.6842	0.8529
Pills	0.8597	0.9157	0.9279
Dollar	0.7352	0.6783	0.8921
Lena	0.8318	0.8896	0.8698
Bear	0.8092	0.8807	0.8418
Einstein	0.8513	0.9173	0.8988
Paper machine	0.8384	0.8788	0.9169
Rose	0.6697	0.6590	0.8278

We also compared the elapsed time for executing the proposed algorithm and the two others algorithms. In Table V, we list these times (in seconds) for all images in our dataset. As can be observed from the results, the proposed model has a much better performance than the other two reference methods, which makes it very adequate to any application that requires real-time processing. It is worth pointing out that both the dispersed-dot ordered algorithm and the proposed inverse halftoning algorithms are highly parallelizable algorithms. So, the processing time could be reduced even further for both the codification and the decodification phases.

TABLE V
ELAPSED TIME FOR RECONSTRUCTION (IN SECONDS).

Image	Proposed	FBIH	WinHD
Cameraman	1.79	2.46	35.14
Dollar	1.01	3.00	52.27
Einstein	1.53	3.00	53.36
Galaxy	1.44	3.01	65.78
Hurricane	1.24	2.99	75.48
Rose	1.24	2.02	73.13
Lena	1.26	2.40	36.29

V. CONCLUSIONS AND FUTURE WORK

We have presented a simple and fast approach for reconstructing halftoned images generated using dispersed-dot ordered dithering algorithms. The proposed algorithm produces images with a better perceptual quality than the available algorithms in the literature, while preserving most of the fine details of the original gray-level image. The proposed method has a high performance, which can be further improved with the use of parallelization techniques. Also, perceptual results would certainly benefit from the use of low-pass and high-pass filters optimized by taking into account a quality metric that is better correlated with perceptual quality, as opposed to PSNR.

ACKNOWLEDGMENT

This work was supported in part by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), in part by a grant by Fundação de Empreendimentos Científicos e Tecnológicos (Finatec), and in part by a grant from DPP - University of Brasília (UnB).

REFERENCES

[1] J.-M. Guo and H. Lee, "Watermarking in halftone images with mixed halftone techniques," *Int. J. Imaging Syst. Technol.*, vol. 17, pp. 303–314, February 2007.

[2] M. Analoui and J. Allebach, "New results on reconstruction of continuous-tone from halftone," *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 3, pp. 313–316, 1992.

[3] Y. Mita, S. Sugiura, and Y. Shimomura, "High quality multi-level image restoration from bi-level image," in *Proc. Sixth Int. Congress Advances in Non-impact Printing Technologies for Black-and-White and Color*, 1990, pp. 791–802.

[4] J. Z. C. Lai and J. Y. Yen, "Inverse error-diffusion using classified vector quantization," *IEEE Transactions on Image Processing*, vol. 7, no. 12, pp. 1753–1758, 1998.

[5] M. Yuan, E. A. Riskin, T. Eve, and A. Riskin, "Error diffused image compression using a binary-to-grayscale decoder and predictive pruned tree-structured vector quantization," *IEEE Transactions on Image Processing*, vol. 3, pp. 854 – 858, 2002.

[6] Z. Karni, D. Freedman, and D. Shaked, "Fast inverse halftoning," *31st International Congress on Imaging Science*, 2010.

[7] R. Neelamani, R. D. Nowak, and R. G. Baraniuk, "Winhd: Wavelet-based inverse halftoning via deconvolution," *Rejecta Mathematica*, pp. 84–103, July 2009.

[8] Z. Xiong, M. T. Orchard, and K. Ramch, "Inverse halftoning using wavelets," in *in Proc. IEEE Int. Conf. Image Proc*, 1996, pp. 569 – 572.

[9] P.-C. Chang, C.-S. Yu, and T.-H. Lee, "Hybrid lms-mmse inverse halftoning technique," *IEEE Transactions on Image Processing*, pp. 95–103, 2001.

[10] N. Damera-venkata, T. D. Kite, M. Venkataraman, and B. L. Evans, "Fast blind inverse halftoning," in *in Proc. IEEE Int. Conf. Image Proc*, 1998, pp. 64–68.

[11] T. D. Kite, N. Damera-venkata, B. L. Evans, and A. C. Bovik, "A high quality, fast inverse halftoning algorithm for error diffused halftones," in *in Proc. IEEE Int. Conf. Image Proc*, 1998, pp. 59–63.

[12] C. Adsumilli, M. C. Q. de Farias, S. K. Mitra, and M. Carli, "A robust error concealment technique using data hiding for image and video transmission over lossy channels," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 15, no. 11, pp. 1394–1406, 2005.

[13] D. E. Knuth, "Digital halftones by dot diffusion," *ACM Trans. Graph.*, vol. 6, pp. 245–273, October 1987.

[14] C. D. Levermore, W. Nadler, and D. L. Stein, "Random walks on a fluctuating lattice: A renormalization group approach applied in one dimension," *Physical Review E*, vol. 51, no. 4, pp. 2779–2786, Apr 1995.

[15] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 9, pp. 81–84, 2002.

[16] Z. Wang, L. Lu, and A. C. Bovik, "Video Quality Assessment Based on Structural Distortion Measurement," *Signal Processing: Image Comm.*, vol. vol19, pp. 121–132, 2004.

[17] Z. Wang and A. C. Bovik, "Mean Squared Error : Love It or Leave It?" *IEEE Signal Processing Magazine*, no. January, pp. 98–117, 2009.

Fig. 5. Results obtained for the images 'Rose', 'Einstein', 'Chester cathedral ' and 'Paper machine': original image (left), halftoned image (center), and reconstructed image using the proposed algorithm (right).

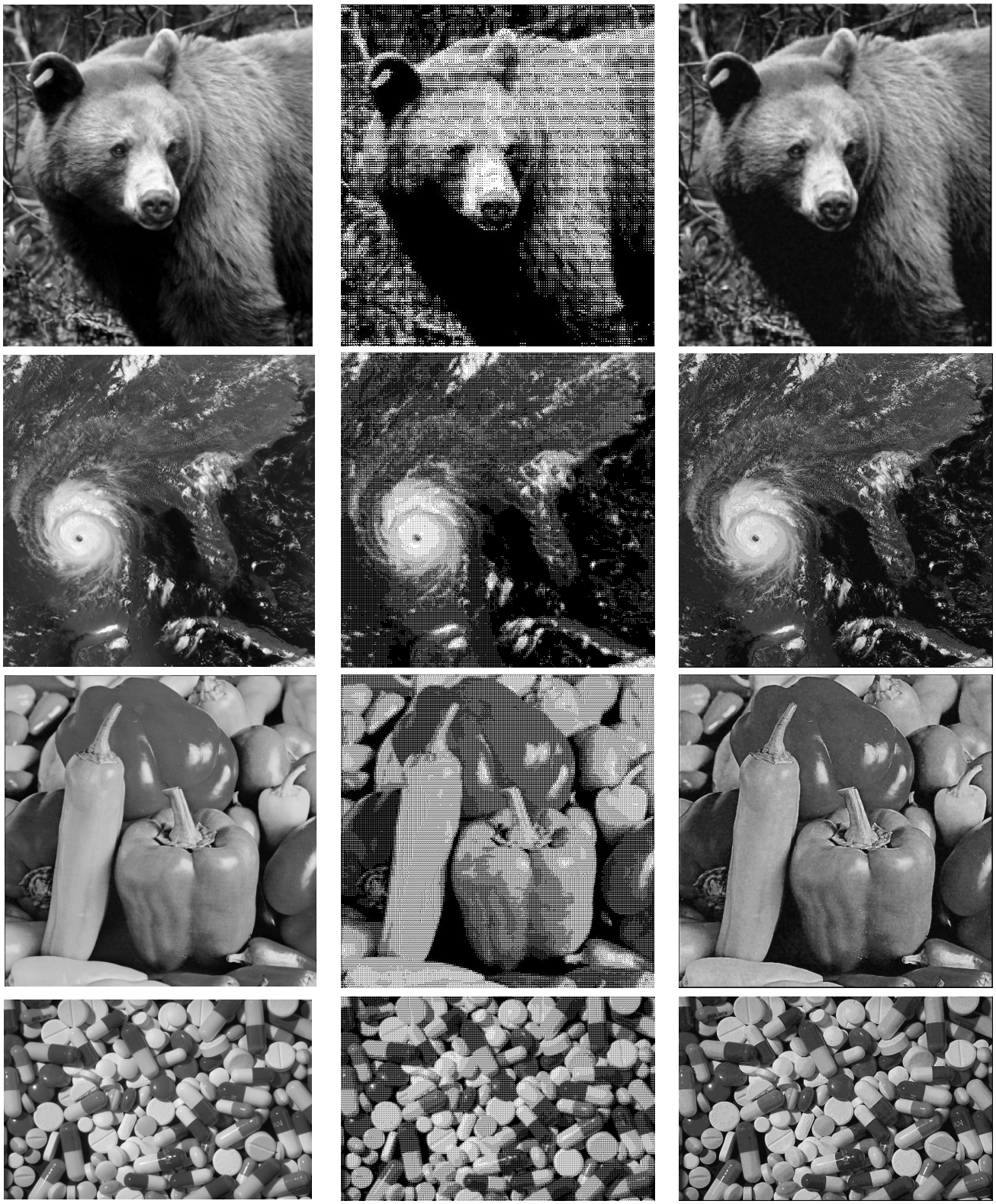

Fig. 6. Results obtained for the images ‘Bear’, ‘Hurricane’, ‘Peppers’ and ‘Pills’: original image (left), halftoned image (center), and reconstructed image using the proposed algorithm (right).

Fig. 7. Results obtained using the Proposed (left), FBIH (center), and WinHD (right) inverse halftoning methods for the images 'Bear', 'Chester cathedral', 'Lena' and 'Peppers'.