A New Approach to Generating Textured Binary Images Using VK Transforms

Yumnam Kirani Singh
Centre for Development of Advanced Computing
Kolkata, India
Yumnam.singh[@]cdac.in

***Abstract*—Proposed here is a new method of generating textured binary images using Varied Kernel (VK) transforms. The varied kernel transforms are perfectly invertible real kernel orthogonal transforms. A property of VK transform is that when an image is decomposed by a kernel and is reconstructed by a slightly different inverse kernel and binarized, the resulted binary image is a textured image. The generated textured binary image carries shade information as in dithered or halftone image and hence is more meaningful and artistic than normal binary image. Textured binary images of different qualities can be achieved by varying the types of the kernels used.**

***Index Terms*—Textured binary image, dithering, VK transforms, real kernel, orthogonal transforms.**

I. Introduction

We need to transform signal from one form to another for various reasons to make them suitable and useful for various applications. The transformation is performed by using transform kernels, which may be complex and real. Some of the popular complex transform kernels are Discrete Cosine Transforms (DCT), Discrete Fourier transforms (DFT), which are mainly used from signal compression and analysis. Some of the popular real kernel transforms are, real wavelet transforms, Walsh and Hadamard transforms used in signal compression and other various applications. Compared to complex transforms, real transforms are simpler and easy to implement in hardware and software. Of the real kernels, integer kernel transforms are preferred from hardware implementation point of view because of difficulty of representing real numbers in hardware. The VK transforms are real transforms similar to Walsh and Hadamard transforms and hence can be easily implemented in software as well as in hardware. For Walsh and Hadamard transforms, the initial kernels consists of 1 and -1 and are fixed 2-by-2 matrices, but in VK transforms, there are eight different initial kernels which are 4-by-4 matrices whose elements can be any four real numbers not all equal to zero. So, when a signal is decomposed with a kernel and reconstructed with a different kernel, there is reconstruction error. This reconstruction error can be used for adding shade information in binary images with textures similar to those obtained using Bayer's method [1]. In the case of Bayer method of Dithering, the textured pattern are fixed to a limited few depending on a few predefined matrix used to generate the textured binary image. In the proposed method, there is a scope of generating different binary images with different textures by choosing different appropriate kernels as desired.

The textured binary images are not as smooth as those of halftone images generated by using error diffusion methods [9] as the textures are coarser than the dots generated in halftone images. So, the textured binary images fail to reproduce in them small details or features in original gray images. However, the overall gray shades in the original image are more visibly incorporated in the textured binary images and in the halftone images. These images will be more suitable for implementing them in textile designs using Jaccard looms or other weaving machines. Also, the textured binary images represents different regions with different textures which may be carefully analyzed and used for image segmentation in certain computer vision and pattern recognition applications.

In this paper we use capital letters for vector or matrix representations. The capitals denotes vectors and bold capitals denote matrices. The elements of the vectors or matrices are written in small letters. We divide the paper into four sections. Section II describes briefly about the VKT. The proposed algorithm for generating textured binary images to incorporate shade information is suggested in section III with their possible area of applications. Section IV gives the experimental results and conclusion is given in section V.

II. Varied Kernel Transforms

We know that kernels of Walsh and Hadamard transforms has only 1 or –1 as its elements [4,5,6,7]. In the generalized Walsh and Hadamard transform [2], we can make kernels of any size with any two real numbers, both not zeros. The original Walsh and Hadamard kernels can be obtained as special cases of them. Extending the generalized Walsh and Hadamard transforms, Varied Kernel transforms [3] are based on four different real numbers k_1, k_2, k_3, k_4 as elements and the minimum kernel is a 4×4 orthogonal matrix. There are eight different types of minimum kernels of size 4×4 as given in Table-1. Type –1 and type-2 are original kernels. Type-3, type-4 and type-5 kernels are derived from type-1 minimum kernel and hence they are given in the first column.

Similarly type-6, type-7 and type-8 are obtained from type-2 and are put in the second column.

Table-1: Eight different Kernels used in VK transforms

${}_h\mathbf{K}_{1.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_2 & -k_1 & k_4 & -k_3 \\ k_3 & -k_4 & -k_1 & k_2 \\ k_4 & k_3 & -k_2 & -k_1 \end{bmatrix}$	${}_h\mathbf{K}_{2.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_2 & -k_1 & -k_4 & k_3 \\ k_3 & k_4 & -k_1 & -k_2 \\ k_4 & -k_3 & k_2 & -k_1 \end{bmatrix}$
${}_h\mathbf{K}_{3.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_4 & k_3 & -k_2 & -k_1 \\ k_3 & -k_4 & -k_1 & k_2 \\ k_2 & -k_1 & k_4 & -k_3 \end{bmatrix}$	${}_h\mathbf{K}_{6.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_3 & k_4 & -k_1 & -k_2 \\ k_2 & -k_1 & -k_4 & k_3 \\ k_4 & -k_3 & k_2 & -k_1 \end{bmatrix}$
${}_h\mathbf{K}_{4.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_2 & -k_1 & k_4 & -k_3 \\ k_4 & k_3 & -k_2 & -k_1 \\ k_3 & -k_4 & -k_1 & k_2 \end{bmatrix}$	${}_h\mathbf{K}_{7.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_4 & -k_3 & k_2 & -k_1 \\ k_3 & k_4 & -k_1 & -k_2 \\ k_2 & -k_1 & -k_4 & k_3 \end{bmatrix}$
${}_h\mathbf{K}_{5.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_3 & -k_4 & -k_1 & k_2 \\ k_2 & -k_1 & k_4 & -k_3 \\ k_4 & k_3 & -k_2 & -k_1 \end{bmatrix}$	${}_h\mathbf{K}_{8.1}=\begin{bmatrix} k_1 & k_2 & k_3 & k_4 \\ k_2 & -k_1 & -k_4 & k_3 \\ k_4 & -k_3 & k_2 & -k_1 \\ k_3 & k_4 & -k_1 & -k_2 \end{bmatrix}$

The VK transform can use any of the following eight orthogonal kernels as seed kernel for generation of a forward and its corresponding inverse kernel for transforming a signal. Larger transformation kernels can be obtained from any of these kernels either using Hadarmard way or Walsh way of kernel generation [3]. In this paper, only Hadamard kernels are considered and hence the suffix h is given on the left of the kernel types. To uniquely identify the types of the kernels, we use 1, 2, 3,..8, as the first suffix on the right of the kernel name. Also, a second suffix is used to denote the level of the kernel, which indicates the size of the kernel.

In table-1, ${}_h\mathbf{K}_{1.1}$ denotes the type-1 at level 1, ${}_h\mathbf{K}_{2.1}$ denotes type-2 at level 1 and so on. The values of k_1, k_2, k_3, k_4 in a kernel can be any real numbers, all of which are not simultaneously equal to zero. The array consisting of any four real numbers used in a kernel will be known as seed array.

From any of minimum kernels, other higher level or larger size kernels can be obtained using the Hadamard way of kernel generation method as given in equation (1).

$${}_h\mathbf{K}_{r.2n}=\begin{bmatrix} {}_h\mathbf{K}_{r.n} & {}_h\mathbf{K}_{r.n} \\ {}_h\mathbf{K}_{r.n} & -{}_h\mathbf{K}_{r.n} \end{bmatrix}\cdots\cdots\cdots(1)$$

Where r=1, 2, …, 8 denotes type of the kernel and n=1,2, 3 .., denotes the level of the kernel.

All the kernels obtained by using equation (1) are orthogonal but they are not symmetric. But once we get a minimum kernel of type r, then we can find two larger kernels ${}_h\mathbf{K}_{r.n}$ denotes the kernels generated by using the Hadamard generator relation (1). As we have only eight classes, the value of r ranges from 1 to 8 and n is any positive integer. The value of n=1 corresponds to the minimum kernel of a particular type. Any kernel generated using the generator relations (1) has size $2^{n+1}\times 2^{n+1}$ for any integer n, as our minimum kernel is of size 4×4.

Once a kernel of desired size of a specified type is obtained using from any four real numbers using equation (1), we can use it to transform a signal.

Let **K** be a generic kernel of size $2^n\times 2^n$ of any types and S is signal vector of length 2^n, then the forward transform is by the matrix multiplication of S and **K** as

$$T=S*\mathbf{K}\cdots\cdots\cdots(2)$$

Where T is a vector of transform coefficients having the same length as S and * denotes matrix multiplication.

The inverse transform of T to reconstruct S back from it is

$$S=\frac{1}{d}T*K'\cdots\cdots\cdots(3)$$

Where $d=2^{n-2}(k_1^2+k_2^2+k_3^2+k_4^2)$ is the normalizing factor and K', is the transpose of the kernel **K**.

From forward equation (2), we can find the transform coefficients of signal decomposed using various types of kernels by substituting the values of the appropriate kernels in it. And the original signal can be reproduced from the decomposed components from equation (3).

For a two dimensional signal, such as image, the forward transform (2) is applied twice, once along the rows and then next along the columns.

Let **X** is an image of size $2^n\times 2^n$ and **K** is a kernel of corresponding size of any one of the eight types. Then, the transform image **Y** is given by

$$\mathbf{Y}=(\mathbf{X}*\mathbf{K})'*\mathbf{K}\cdots\cdots\cdots(5)$$

Similarly, the inverse transform of **Y** to obtain **X** back is

$$\mathbf{X}=\frac{1}{d^2}(\mathbf{Y}*\mathbf{K}')'*\mathbf{K}'\cdots\cdots\cdots(6)$$

We have seen that in our new VK transform, there are eight minimum transform kernels from which we can get 16 different types of higher kernels. And we can choose any four real numbers, not all four are zero, as elements of kernel of any size greater than 4×4. In other words we have more alternatives in choosing kernel elements, which may be useful in cryptography, coding or in filtering purposes

III. Algorithm for generating Textured binary Image

Adding shade information in binary image is done using halftoning techniques [1, 9]. The shade information in binary image can also added using VK transforms of Hadamard type kernels. The underlying technique used for adding shade information here is simple. An image is transformed using equation (5) with a kernel generated a seed array and is reconstructed using equation (6) with a kernel generated from slightly different seed array. The reconstructed image is not exactly the same as the original image. The reconstructed image or the error image between the original image and the reconstructed image is thresholded, a textured image with gray shades is generated.

Let $\mathbf{X}$ be the original image, and F be a seed array to generate a kernel for forward transformation and G be a seed array to generate different kernel for inverse transformation to give reconstructed image $\hat{\mathbf{X}}$. Then the textured binary image $\mathbf{B}$ is given by

$$\left.\begin{aligned}\mathbf{B} &= (\mathbf{X}-\hat{\mathbf{X}}) > t \\ &= \left\|\mathbf{X}-\hat{\mathbf{X}}\right\| > t\end{aligned}\right\} \cdots\cdots\cdots(7)$$

Where t is a real number representing a threshold value and $\|.\|$ denotes absolute value.

Note here that different kernels may be generated from F and G even if they are equal, if the types of the kernels to be generated are different.

The algorithm for generating texturized binary image is as follows.

1. Read an image $\mathbf{X}$ and convert it to gray if required. If image (matrix) is not a square matrix of order of power of 2, make it so by appending necessary zeros.
2. Generate a forward transform kernel **K** of appropriate size with seed array F for type r.
3. Transform the image using **K** to get the transformed image **Y**
4. Generate an inverse transform kernel **K**' with a seed array G for type r (for $F \neq G$) or s.
5. Perform inverse transformation on **Y** using **K**' to get the reconstructed image $\hat{\mathbf{X}}$.
6. Binarize the reconstructed image $\hat{\mathbf{X}}$ or the error image $\mathbf{X}-\hat{\mathbf{X}}$ at proper threshold to get the texturized binary image **B**.

It may be noted that kernels generated from the same seed array will be different for different types. Also, as the seed array can be any four real numbers, we can easily generate different kernels of same type. However, all kernels do not give good quality textured binary images, so a suitable seed array must be chosen.

IV. Experimental Results

We use for our experiment the Lena image commonly used in the image processing community. The tool used in the experiment is the Image Processing Tool-1, available in SCILAB ATOMS distribution center. We use the vk, vkt2 and ivkt2 of IPT-1 [10]. First, we generate a kernel of appropriate size from a seed array and transform the image using vkt2. The transformed image is then reconstructed using ivkt2 with a different kernel generated either from a slightly different seed array or from a different type. The reconstructed image is then binarized by choosing an appropriate threshold value. Table-1 gives a list of types of textures generated, when an image decomposed with a kernel of given type from a seed array is reconstructed with kernels of different types from the same seed array.

Table-1: Texture types from using Seed Array [1, 2, 3, 4]

Types	1	2	3	4	5	6	7	8
1	0	+	--	--	--	+	--	+
2	+	0	+	|	|	--	+	--
3	--	+	0	--	--	|	--	+
4	**--**	**+**	**--**	**0**	**--**	**+**	**--**	**+**
5	**--**	**+**	**--**	**--**	**0**	**+**	**--**	**+**
6	|	--	|	+	+	0	+	--
7	|	+	|	+	|	|	0	+
8	+	|	+	+	+	|	+	0

In the Table, the first horizontal column denotes the types of the kernels used for forward transformation and the first row indicates the types of kernels used for inverse transformation. All kernels are generated from the same seed array [1, 2, 3, 4] chosen for simplicity. Using this seed array, we can get mainly three different types of textured binary image namely smooth or normal binary (~), cross textured (+), horizontal line (--) and vertical line (|). For example, when an image is forward transformed with a type-4 or 5 kernel generated from the seed array [1, 2, 3, 4] and are reconstructed using rest of seven kernels generated from the same seed array, the four types of textured binary images can be obtained as indicated by two bold rows in Table-1. By cross textured, we will mean any texture pattern which are neither horizontal lines nor vertical lines nor slant lines. In the table, the 0 entries along the diagonal lines signifies zero reconstruction error or perfect reconstruction. Also, from the table, we see that in most of the cases, we get cross textured binary images and smooth binary images occurs in least cases.

Figure-1 shows the original Lena image in gray. The textured binary image generated by using Bayer's kernel of size 32 is shown in Figure-2. Only such type of cross textured binary images can be produced by using Bayer's kernels. However, using the proposed method, we can produce several different types of textures in binary images with artistic appeal.

Figure-3 shows different textured binary images obtained using the proposed algorithm with seed array [1, 2, 3, 4]. Figure-3(a) is a textured binary image in which horizontal lines are dominant in the texture when an image is decomposed using type-1 kernel and reconstruct using either of type -3, 4 or 5. Figure-3 (b) and (c) are the cross textured binary images when the Lena image is decomposed using type-2 or type-3 and reconstructed using type-3 or 2. From these two figures, it can be observed that different types of cross textured binary images may be obtained by different kernels used in the decomposition and reconstruction process in different order. Figure-3(d) is the textured binary image obtained when the Lena image is decomposed using kernel 7 and reconstructed using kernel 5. From the figure, it is seen that the texture in the binary image consists dominantly of vertical lines.

Other different types of textured binary images, where the textures consists of both vertical and horizontal lines, netted grids etc., can be obtained when we successively apply decomposition using two or more different kernels and reconstruct successively using same or slightly different kernels. However, results are not shown for paucity of space.

V. Conclusions

A new method of generating textured binary image based on VK transform has been proposed. The method is simple, effective and can be used to generate different types of textured binary images. It is better than the Bayer's method of generating textured binary images in the sense that textures are more regular and different qualities of textured binary images can be obtained from the same error image at different thresholds or from different error images obtained by inverse transformation with different kernels. In some of the cases, binarization of the reconstructed image in place of error image gives good textured binary images. A limitation of the proposed method is that the kernels are square matrices of order of power of 2 and images of different size should be made of power of 2 before applying this method.

References

[1] Bryce Bayer, “An optimum method for two-level rendition of continuous-tone pictures”, International Conference on Communications vol-1, pp. 11–15, June, 1973.

[2] Y. Kirani Singh, “Generalized Walsh and Hadamard Transform”, http://www.scribd.com/doc/18091224/Generalized-Walsh-and-Hadamard-Transform

[3] Y. Kirani Singh, “VK Transform, as a generalized version to Walsh and Hadamard Transforms”, ttp://www.researchgate.net/publication/235741610_VK_transform_as_generalized_version_to_Walsh_and_Hadamard_transforms

[4] Beer, T. Walsh transforms. American Journal of Physics. 49(5):466-472, 1981.

[5] Corrington, M. Solution of Differential and Integral Equations with Walsh Functions. IEEE Transactions on Circuit Theory. CT-20(5): 470-476, 1973.

[6] Hadeyat A., Wallis W. D. "Hadamard matrices and their applications", Ann. Stat. 6:pp 1184-1238, 1978

[7] R. K. Rao Yarlagadda, J. E. Hershey Hadamard Matrix Analysis and Synthesis: With Applications to Communications and Signal/Image Processing. (Kluwer International Series in Engineering), 1997.

[8] Hadeyat A., Wallis W. D. "Hadamard matrices and their applications", Ann. Stat. 6:pp 1184-1238, 1978.

[9] R. Ulichney, “A Review of Halftoning Techniques,” Proceedings of SPIE, Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts V, San Jose, CA, USA, vol. 3963, pp. 378–391, January 2000.

[10] Scilab modules: http://atoms.scilab.org/toolboxes/IPT2/1.0

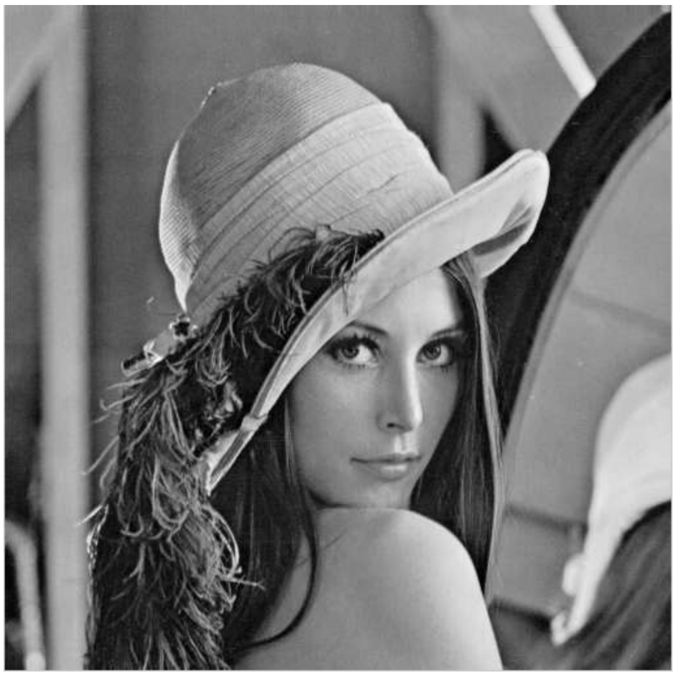
Figure-1: Original Lena Image

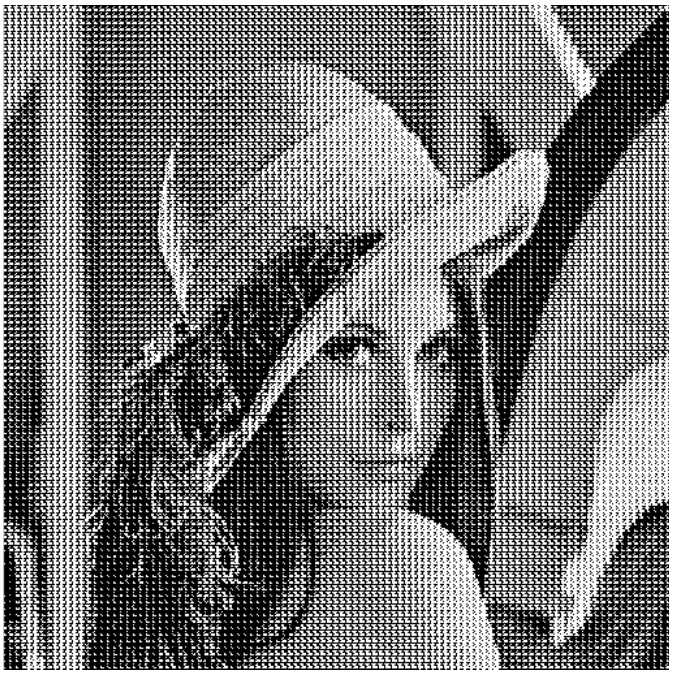
Figure-2: Textured Binary image using Bayer kernel (32x32)

About the Author:

Yumnam Kirani Singh, has completed Master’s Degree in Electronics Science from Guwahati University in 1997 and got Ph. D. degree from Indian Statistical Institute, Kolkata in 2006. Served as a lecturer in Electronics in Shri Shankaracharaya College of Engineering & Technology from Jan, 2005 to May, 2006. Joined CDAC Kolkata in May 2006 and worked there before coming to CDAC Silchar, in March 2014. Developed Bino’s Model of Multiplication, ISITRA, YKSK Transforms and several other image binarization and edge detection techniques. Interested to work in the application and research areas of Signal Processing, Image Processing, Pattern recognition and Information Security. Published several papers in national and international journals and conferences

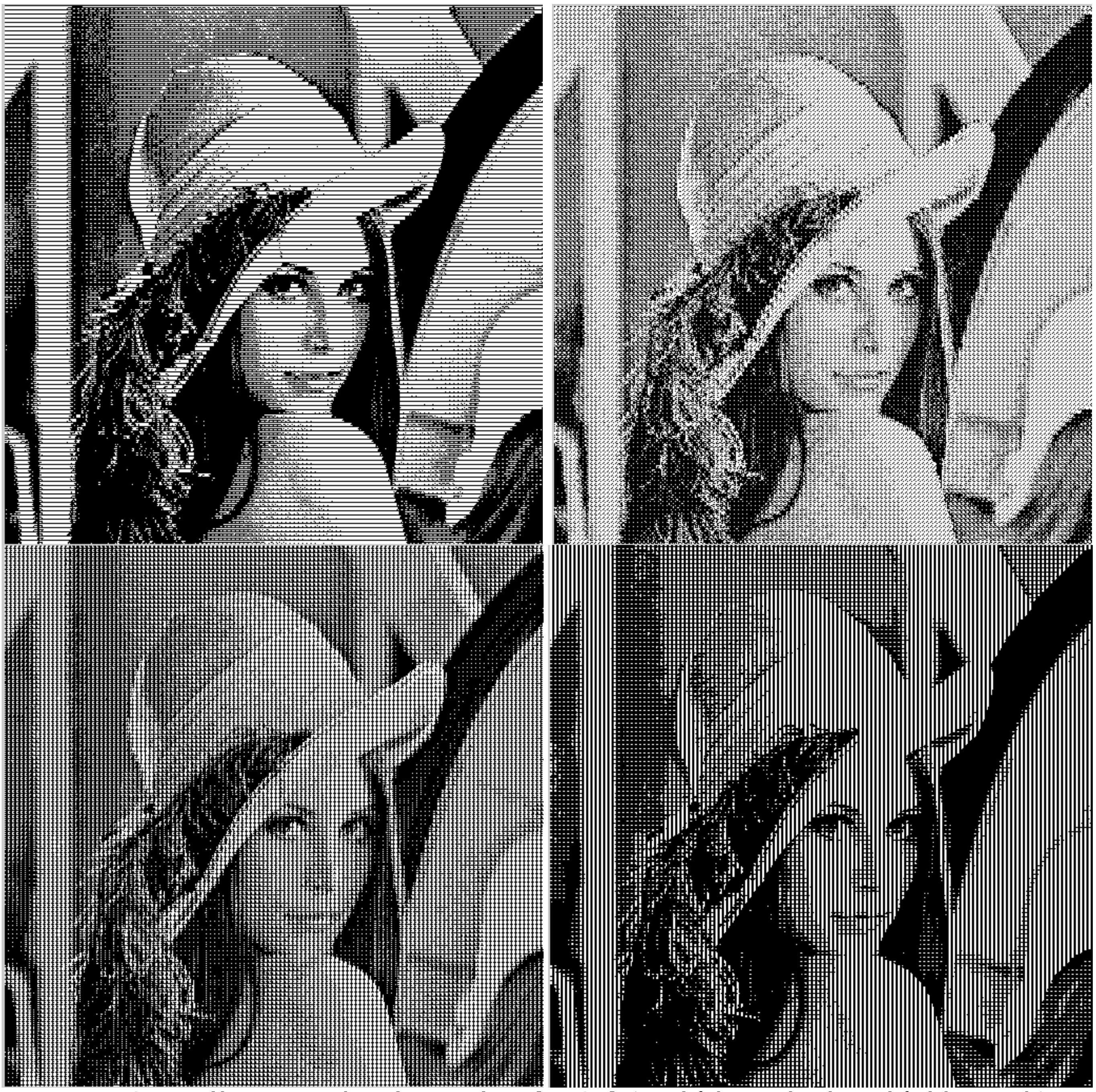

Figure-3:Textured binary images obtained using seed array [1, 2, 3, 4] (a-top left, b- top right, c-bottom left, d- bottom right)
a: Decomposed using type-1 and reconstructed using type-3 b: Decomposed using type-2 and reconstructed using type-3
c: Decomposed using type-3 and reconstructed using type-2 d: Decomposed using type-7 and reconstructed using type-5