Available online at www.sciencedirect.com

ScienceDirect

J. Vis. Commun. Image R. 17 (2006) 1209–1226

JOURNAL OF VISUAL Communication & IMAGE Representation

www.elsevier.com/locate/jvci

A hybrid gray image representation using spatial- and DCT-based approach with application to moment computation ☆

Kuo-Liang Chung [a,*], Yau-Wen Liu [b], Wen-Ming Yan [b]

[a] *Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672, ROC*

[b] *Department of Computer Science and Information Engineering, National Taiwan University, No.1, Section 4, Roosevelt Road, Taipei, Taiwan 10617, ROC*

Received 19 September 2005; accepted 24 January 2006
Available online 9 March 2006

Abstract

In this paper, a novel hybrid gray image representation using spatial- and DCT-based approach is presented. In the first phase, according to the bintree decomposition principle under the specified error, an S-tree spatial data structure (SDS) is used to represent the decomposed bintree of the input gray image. In the constructed S-tree SDS, the leaves are partitioned into two types, namely the homogeneous leaves and the nonhomogeneous leaves. The homogeneous leaf is used to represent one rectangular or square homogeneous subimage with smooth, i.e., low frequency, content and the nonhomogeneous leaf is used to represent one nonhomogeneous subimage with nonsmooth, i.e., high frequency, content. In the second phase, each nonhomogeneous leaf is encoded by the DCT-based coding scheme for reducing the memory requirement. Based on some real gray images, experimental results show that our proposed gray image representation over the previously published S-tree- and shading-based SDS has about 63.08% memory-saving improvement ratio in average. Finally, we investigate the computational benefit when computing moments on our proposed gray image representation directly.
© 2006 Elsevier Inc. All rights reserved.

Keywords: DCT; Gray image representation; Linear interpolation; Moment computation; PSNR; Spatial data structures

1. Introduction

Using spatial data structures (SDSs) for representing binary images has a long history [23]. Based on these different kinds of developed SDSs directly, many efficient algorithms have been presented for many applications [3,4,1,2,24,16,17], such as image processing, pattern recognition, computational geometry, computer graphics, databases, cryptography, and so on. For compressing binary images, the compression standard,

☆ K.-L. Chung was supported by the National Science Council of ROC under contract NSC93-2213-E011-019.
* Corresponding author. Fax: +886 2 273 76777.
E-mail address: klchung@csie.ntust.edu.tw (K.-L. Chung).

1047-3203/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2006.01.002

e.g., JBIG [28,12], always has better compression performance when compared to any existing SDS for representing binary images, but it is impossible to manipulate the compressed form of JBIG for many applications due to the involvement of entropy coding. Among these developed SDSs for representing binary images, the well-known ones are the linear quadtree [10], the S-tree representation [14], the DF-expression [15], the improved linear quadtrees [18,19], etc.

Based on the B-tree triangular coding (BTTC) approach, Distasi et al. [7] presented the first SDS for representing gray images. The BTTC-based SDS by Distasi et al. is really a pioneer work to extend the SDS design from binary images to gray images. In [7], for each partitioned block in the gray image domain, a triangle is used to represent that block. Later, based on the S-tree data structure and the Gouraud shading method [9], a new S-tree coding (STC)-based approach [5] for representing gray images was presented. Experimental results show that the STC-based SDS has a superior performance in the execution time (less than half of the BTTC-based SDS) without sacrificing compression ratio and image quality. Based on the STC-based gray image representation, an efficient image algorithms for lower order moment computation [6] has been successfully developed. Surprisingly, low-order moment computation can be done in $O(K)$ time where K denotes the number of partitioned blocks in the STC-based SDS. For representing gray images, the main drawback in the BTTC-based SDS or the STC-based SDS is that it needs a large amount of memory to represent nonsmooth regions, i.e., regions with high frequency contents.

In this paper, a novel spatial- and DCT-based (SDCT-based) image representation is presented for representing gray images. In the first phase, according to the bintree decomposition rule under the specified error, a conventional S-tree SDS is used to represent the decomposed bintree of the input gray image. In the constructed S-tree SDS, the leaves are partitioned into two types, namely the homogeneous leaves and nonhomogeneous leaves. The homogeneous leaf is used to represent one rectangular or square homogeneous subimage with low frequency, i.e., smooth, content, and the nonhomogeneous leaf is used to represent one nonhomogeneous subimage with high frequency content. In the second phase, each nonhomogeneous leaf obtained from the first phase is encoded by the DCT-based coding scheme. Our proposed SDCT-based image representation has a better memory-saving effect when compared to the previous STC-based SDS and experimental results reveal about 63% memory-saving improvement ratio. Finally, we discuss the computational benefit when computing moments on our proposed SDCT-based image representation directly.

The remainder of this paper is organized as follows. Section 2 gives the survey of the previous STC-based SDS for representing gray images. Section 3 presents our proposed SDCT-based image representation for representing gray images. In Section 4, we demonstrate how to compute the low-order moments efficiently on the proposed image representation directly. In Section 5, some experiments are carried out to demonstrate the memory-saving and computational advantages of our proposed SDCT-based image representation. Finally, some concluding remarks are addressed in Section 6.

2. Past work: STC-based SDS for representing gray images and its shortcoming

Since the STC-based SDS has a superior performance in the execution time (less than half of the BTTC-based SDS) without sacrificing compression ratio and image quality, we only introduce the past STC-based SDS for representing gray images. Then its main shortcoming is pointed out. The main problem is that the STC-based SDS does decompose a nonhomogeneous block with high frequency content into many smaller homogeneous blocks and it leads to a memory-redundant side effect. In fact, besides the STC-based SDS, the BTTC-based SDS also has the same memory-redundant problem for representing high frequency content. In next section, our proposed SDCT-based gray image representation not only preserves the memory-saving advantage of the STC-based SDS for representing low frequency content, but also resolves the memory-redundant problem for representing high frequency content.

In the STC-based SDS, the input gray image is partitioned into a set of homogeneous blocks according to the bintree decomposition rule. At each decomposition step, it partitions the subimage into two smaller subimages, each with equal size, in y- and x-axes by turns. When the partitioned subimage is nonhomogeneous, the decomposing step is proceeded until each subimage is homogeneous. For convenience, a homogeneous subimage is called a homogeneous block. In next paragraph, the formal definition of a homogeneous block is shown.

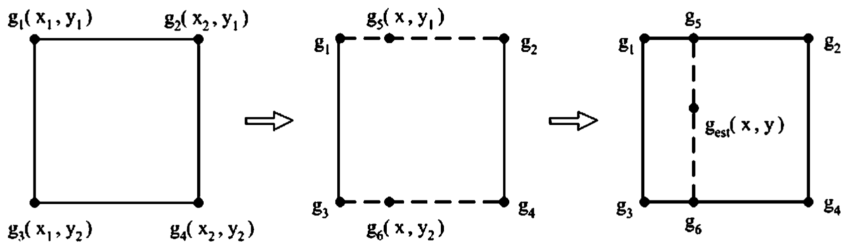


Fig. 1. A homogeneous block and the linear interpolation.

As shown in Fig. 1, there are four corner pixels which must be recorded for a homogeneous block. Suppose the coordinates of the four corner pixels are (x_1, y_1), (x_2, y_1), (x_1, y_2), and (x_2, y_2); their gray values are equal to g_1, g_2, g_3, and g_4, respectively. According to Gouraud shading method [9], i.e., the linear interpolation, the estimated gray level at position (x, y) in the block is calculated by

$$g_{\text{est}}(x,y) = g_5 + (g_6 - g_5) \times i_1, \tag{1}$$

where $g_5 = g_1 + (g_2 - g_1) \times i_2$, $g_6 = g_3 + (g_4 - g_3) \times i_2$, $i_1 = \frac{y-y_1}{y_2-y_1}$, and $i_2 = \frac{x-x_1}{x_2-x_1}$.

By definition [9], g_5 and g_6 are the estimated gray values of the pixel at position (x, y_1) and (x, y_2), respectively; i_1 and i_2 are the linear interpolation of the gray value in y- and x-axes, respectively. Thus, given a specified error tolerance ϵ, the block is homogeneous if the following image quality condition holds:

$$|g(x,y) - g_{\text{est}}(x,y)| \leqslant \epsilon. \tag{2}$$

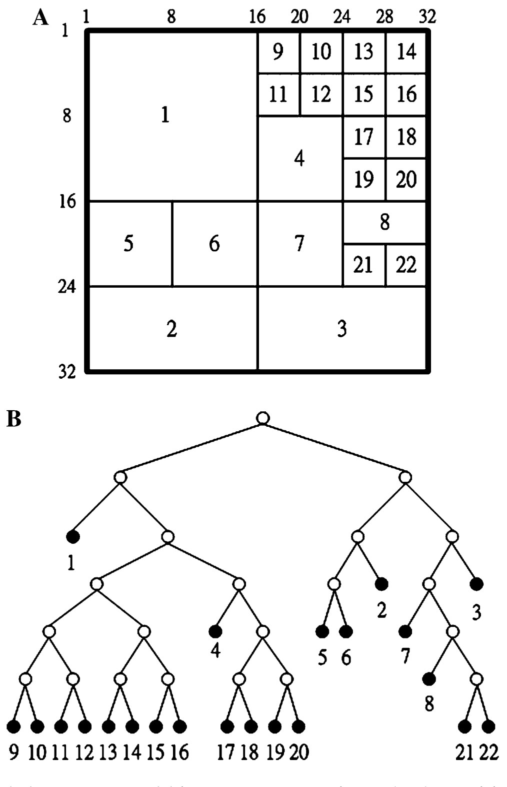


Fig. 2. A partitioned image example and the constructed bintree representation. (A) A partitioned image example. (B) The constructed bintree structure of (A).

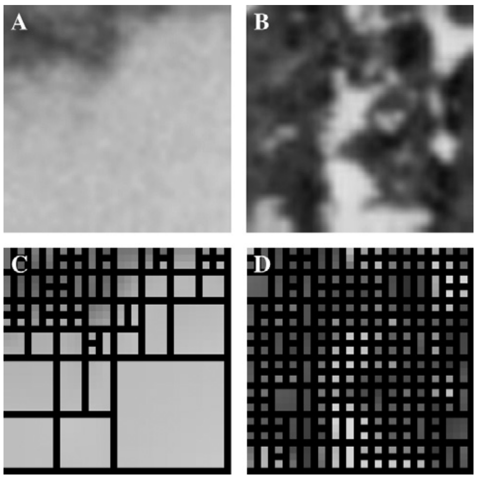


Fig. 3. Memory-redundant problem in the STC-based SDS. (A) A low frequency subimage: sky part of F16 image. (B) A high frequency subimage: mountain part of F16 image. (C) The partitioned homogeneous blocks of (A). (D) The partitioned homogeneous blocks of (B).

Suppose one gray image has been divided into a set of homogeneous blocks as shown in Fig. 2A. The bintree structure of Fig. 2A is shown in Fig. 2B, where the indices of partitioned homogeneous blocks in Fig. 2A are arranged by the partitioning orders. Based on the breadth first search (BFS) scanning order, the indices of the scanned leaves form a strictly increasing sequence.

Following the constructed bintree structure, the S-tree representation consisting of one linear-tree table and one color table is used to represent the partitioned image corresponding to the input gray image. The linear-tree table records the bintree structure by traversing the bintree in a BFS manner and ‘0’s are used to keep the relationship of these partitioned homogeneous blocks; ‘1’s in the linear-tree table are used to represent the partitioned homogeneous blocks. The color table records the related gray levels of four corner pixels of these homogeneous blocks which are denoted by ‘1’s in the linear table. Upon scanning the bit ‘1’ in the linear-tree table, the information of four corners with respect to the homogeneous block can thus be retrieved from the corresponding entry in the color table. The S-tree representation of Fig. 2B is expressed by the following two tables:

Linear-Tree Table	:	0001000000101001011100000001011111111111111
Color Table	:	$(g_{1_{ul}}g_{1_{ur}}g_{1_{bl}}g_{1_{br}})$
		$(g_{2_{ul}}g_{2_{ur}}g_{2_{bl}}g_{2_{br}})$
		$(g_{3_{ul}}g_{3_{ur}}g_{3_{bl}}g_{3_{br}})$
		$(g_{4_{ul}}g_{4_{ur}}g_{4_{bl}}g_{4_{br}})$
		...
		$(g_{22_{ul}}g_{22_{ur}}g_{22_{bl}}g_{22_{br}})$

As mentioned before, the previous STC-based SDS and the BTTC-based SDS for representing gray images has the memory-redundant problem in the nonhomogeneous block with high frequency content. For highlighting this memory-redundant problem, the following example is given. Given two subimages as shown in Figs. 3A and B, Fig. 3A is the subimage extracted from the sky part of gray image F16 (see Fig. 7B) and Fig. 3B is the one extracted from the mountain part of the same image. According to the bintree decomposition rule under the error tolerance $\epsilon = 10$, Figs. 3C and D denote the partitioned homogeneous blocks of Figs. 3A and B, respectively. It can be counted that Fig. 3C has 64 homogeneous blocks and Fig. 3D has 210 homogeneous blocks. The ratio of the number of blocks required in Fig. 3D over that of Fig. 3C is about 3.28. This ratio indicates that the subimage shown in Fig. 3B (Fig. 3A) with high (low) frequency content needs more (less) memory to save these partitioned homogeneous blocks when using the STC-based SDS.

3. The proposed SDCT-based gray image representation

In this section, our proposed SDCT-based gray image representation is presented. For resolving the memory-redundant problem mentioned above for saving high frequency content, we apply the DCT approach to represent the nonhomogeneous block with high frequency content and it can lead to a significant memory-saving effect when compared to the SDS approach for representing the same kind of blocks. For saving low frequency content, we still utilize the previous STC-based SDS.

Given a gray $b \times b$ subimage $\{f(x, y)|0 \leqslant x \leqslant b-1, 0 \leqslant y \leqslant b-1\}$, suppose each pixel value has been subtracted from 128. Applying the DCT to the subimage, it can be expressed by

$$F(u,v) = \frac{1}{\sqrt{2b}} C(u)C(v) \sum_{x=0}^{b-1} \sum_{y=0}^{b-1} f(x,y) \cos\frac{(2x+1)u\pi}{2b} \cos\frac{(2y+1)v\pi}{2b}, \tag{3}$$

where $C(0) = \frac{1}{\sqrt{2}}$ and $C(i) = 1$ for $i = 1, 2, \cdots, b-1$. Let us cut off the left-upper quadrant of Fig. 3B and the cut nonhomogeneous subimage is depicted in Fig. 4A with high frequency content. Fig. 4A is used as the example to demonstrate the memory-saving advantage when combining the DCT approach and the STC-based approach. Under the error tolerance $\epsilon = 10$, Fig. 4B demonstrates the 50 partitioned homogeneous blocks of Fig. 4A and the number of corner pixels of Fig. 4B is 200. Fig. 4C illustrates the DCT coefficient matrix of Fig. 4A. Based on the quantization table as shown in Figs. 4D and E demonstrates the quantized DCT coefficient matrix of Fig. 4C. It can be counted that the number of nonzero quantized DCT coefficients in Fig. 4E is 43 and it is much less than the number of corner pixels in Fig. 4B. The reduction of number of coefficients leads to a memory-saving effect.

Combining the advantages of SDS-based approach for representing homogeneous blocks, each block with low frequency content, and DCT-based approach for representing nonhomogeneous blocks, each block with high frequency content, an example is now taken to justify the compromise between the two approaches. Integrating the two approaches constitutes the main concept of our proposed two-phase SDCT-based approach for representing gray images. Let us return to Fig. 2A. Since these homogeneous blocks, each with size 4×4, denoted by indices from 2 to 14 are some memory-redundant. In what follows, our proposed SDCT-based approach is presented to resolve this memory-redundant problem existed in the STC-based approach.

In the first phase of our proposed SDCT-based approach, Fig. 2A could be partitioned to a specified level, say level 4, according to the bintree decomposition rule. Thus, the constructed bintree structure Fig. 2B becomes the approximate bintree structure as shown in Fig. 5A where the leaves are classified into two types. The nonhomogeneous leaves denoted by triangle nodes represent high frequency contents and the homogeneous leaves denoted by black nodes represent low frequency contents. The corresponding partitioned blocks of Fig. 5A is depicted in Fig. 5B.

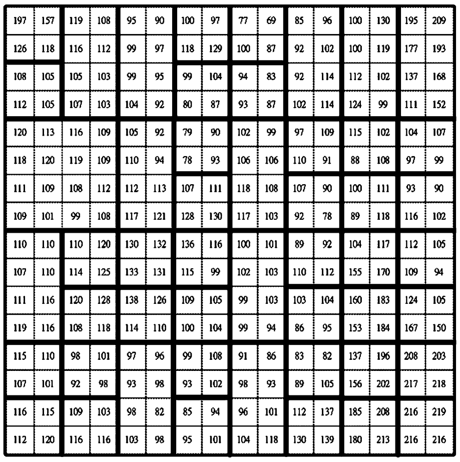
In the second phase of our proposed SDCT-based approach, the nonhomogeneous leaves in the approximate bintree structure are encoded by the DCT-based coding scheme while the homogeneous leaves are encoded by the STC-based representation. For example, performing the DCT on block 4 in Fig. 5B, which is shown in Fig. 6A, the quantized DCT coefficient matrix is shown in Fig. 6B. According to the zig-zag scanning way as shown in Fig. 6C, the scanned quantized DCT coefficient sequence is shown in Fig. 6D. Due to the energy compaction property of DCT, the seven nonzero DCT coefficients appear in the leading subsequence.

According to our proposed SDCT-based approach, our proposed SDCT-based gray image representation consists of the linear-tree table, the leaf-type table, and the color table. The linear-tree table records the approximate bintree structure by traversing the bintree in a BFS manner; '0's are used to preserve the relationship of these leaf nodes and '1's are used to represent these leaf nodes which indicate the relevant partitioned blocks. The leaf-type table is used to discriminate between the leaf nodes encoded by STC-based approach for homogeneous blocks and the leaf nodes encoded by DCT-based approach for nonhomogeneous blocks. The color table keeps the related gray levels of four corner pixels for representing homogeneous blocks and the zig–zag scanned sequences of the quantized DCT coefficients for representing the nonhomogeneous blocks. The contents of three tables for representing Fig. 5A are listed below:

A

197	157	119	108	95	90	100	97	77	69	85	96	100	130	195	209
126	118	116	112	99	97	118	129	100	87	92	102	100	119	177	193
108	105	105	103	99	95	99	104	94	83	92	114	112	102	137	168
112	105	107	103	104	92	80	87	93	87	102	114	124	99	111	152
120	113	116	109	105	92	79	90	102	99	97	109	115	102	104	107
118	120	119	109	110	94	78	93	106	106	110	91	88	108	97	99
111	109	108	112	112	113	107	111	118	108	107	90	100	111	93	90
109	101	99	108	117	121	128	130	117	103	92	78	89	118	116	102
110	110	110	120	130	132	136	116	100	101	89	92	104	117	112	105
107	110	114	125	133	131	115	99	102	103	110	112	155	170	109	94
111	116	120	128	138	126	109	105	99	103	103	104	160	183	124	105
119	116	108	118	114	110	100	104	99	94	86	95	153	184	167	150
115	110	98	101	97	96	99	108	91	86	83	82	137	196	208	203
107	101	92	98	93	98	93	102	98	93	89	105	156	202	217	218
116	115	109	103	98	82	85	94	96	101	112	137	185	208	216	219
112	120	116	116	103	98	95	101	104	118	130	139	180	213	216	216

B



197	157	119	108	95	90	100	97	77	69	85	96	100	130	195	209
126	118	116	112	99	97	118	129	100	87	92	102	100	119	177	193
108	105	105	103	99	95	99	104	94	83	92	114	112	102	137	168
112	105	107	103	104	92	80	87	93	87	102	114	124	99	111	152
120	113	116	109	105	92	79	90	102	99	97	109	115	102	104	107
118	120	119	109	110	94	78	93	106	106	110	91	88	108	97	99
111	109	108	112	112	113	107	111	118	108	107	90	100	111	93	90
109	101	99	108	117	121	128	130	117	103	92	78	89	118	116	102
110	110	110	120	130	132	136	116	100	101	89	92	104	117	112	105
107	110	114	125	133	131	115	99	102	103	110	112	155	170	109	94
111	116	120	128	138	126	109	105	99	103	103	104	160	183	124	105
119	116	108	118	114	110	100	104	99	94	86	95	153	184	167	150
115	110	98	101	97	96	99	108	91	86	83	82	137	196	208	203
107	101	92	98	93	98	93	102	98	93	89	105	156	202	217	218
116	115	109	103	98	82	85	94	96	101	112	137	185	208	216	219
112	120	116	116	103	98	95	101	104	118	130	139	180	213	216	216

C

1850	-106	84	35	47	-13	28	-3	-3	2	-2	-8	-9	-3	-2	0
-150	148	-156	14	18	2	32	3	12	-1	-6	8	10	-4	0	12
220	-49	174	37	-10	41	20	34	25	9	10	21	4	11	-5	-2
-93	34	-29	-49	-12	-25	52	13	-9	15	8	12	-6	7	6	-1
30	45	69	10	23	76	-33	-7	27	-13	-7	-5	8	1	-6	-3
32	-40	-20	43	9	-3	1	7	2	14	2	-2	-4	6	9	3
-36	55	20	-8	10	-1	-10	10	10	4	8	14	6	-3	-2	2
36	-51	-37	9	-7	14	21	5	0	6	4	-5	4	4	3	-5
4	31	19	-6	-14	-12	-10	2	1	12	-2	5	-4	-2	4	1
5	9	12	13	19	23	-2	-4	7	-10	-4	-7	3	2	-4	-1
6	-15	-10	7	-2	-11	1	0	0	4	5	-1	-3	-3	2	2
-19	19	13	-5	4	4	-8	4	4	0	-7	1	-1	-1	9	2
11	-7	-13	-6	-8	-1	9	0	-8	3	8	-2	-1	2	-4	-5
1	7	10	-1	-4	-1	-3	0	3	1	-7	-1	-1	0	-2	3
-7	4	-8	-1	3	-3	0	3	-2	-3	-3	0	3	1	-3	1
-1	6	-2	-2	0	-3	1	1	0	3	3	2	-2	1	-2	0

D

16	14	11	11	10	13	16	20	24	32	40	46	51	56	61	66
14	13	12	12	12	15	18	22	25	37	49	53	56	57	58	59
12	12	12	13	14	17	19	23	26	42	58	59	60	58	55	52
13	13	13	14	15	19	22	28	33	46	58	61	65	61	56	51
14	14	13	15	16	20	24	32	40	49	57	63	69	63	56	49
14	15	15	18	19	23	27	36	46	59	72	74	75	67	59	81
14	16	17	20	22	26	29	40	51	69	87	84	80	71	62	83
16	18	20	25	30	37	43	51	60	79	98	95	92	81	70	89
18	20	22	30	37	47	56	62	68	89	109	106	103	90	77	64
21	25	29	38	46	54	60	68	75	91	107	108	108	97	85	73
24	30	35	45	55	60	64	73	81	93	104	109	113	103	92	81
37	44	50	58	67	72	76	84	92	103	113	115	117	107	97	86
49	57	64	71	78	83	87	95	103	112	121	121	120	111	101	91
61	70	78	83	87	90	93	100	108	109	111	112	112	106	100	94
72	82	92	94	95	97	98	105	112	106	100	102	131	101	99	97
83	94	106	105	103	104	103	110	116	103	89	92	94	96	98	100

E

115	-7	7	3	4	-1	1	0	0	0	0	0	0	0	0	0
-10	11	-12	1	1	0	1	0	0	0	0	0	0	0	0	0
18	-4	14	2	0	2	1	1	0	0	0	0	0	0	0	0
-7	2	-2	-3	0	-1	2	0	0	0	0	0	0	0	0	0
2	3	5	0	1	3	-1	0	0	0	0	0	0	0	0	0
2	-2	-1	2	0	0	0	0	0	0	0	0	0	0	0	0
-2	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	-2	-1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 4. An example to demonstrate the memory-saving advantage of DCT approach. (A) The upper-top one-fourth subimage of Fig. 3B. (B) The partitioned homogeneous blocks of (A). (C) The DCT coefficient matrix of (A). (D) The used quantization table. (E) The quantized DCT coefficient matrix of (C).

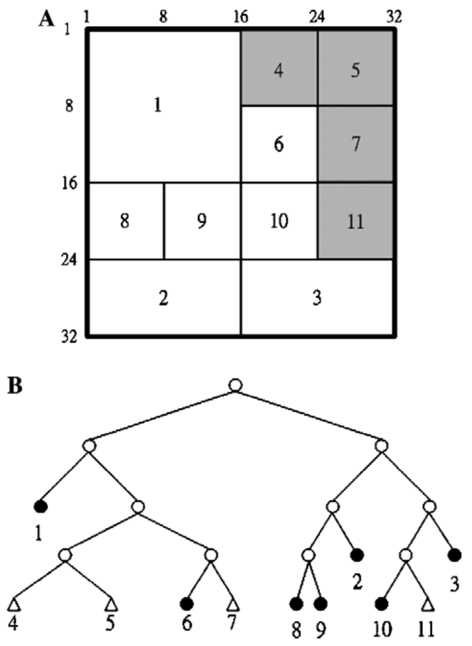
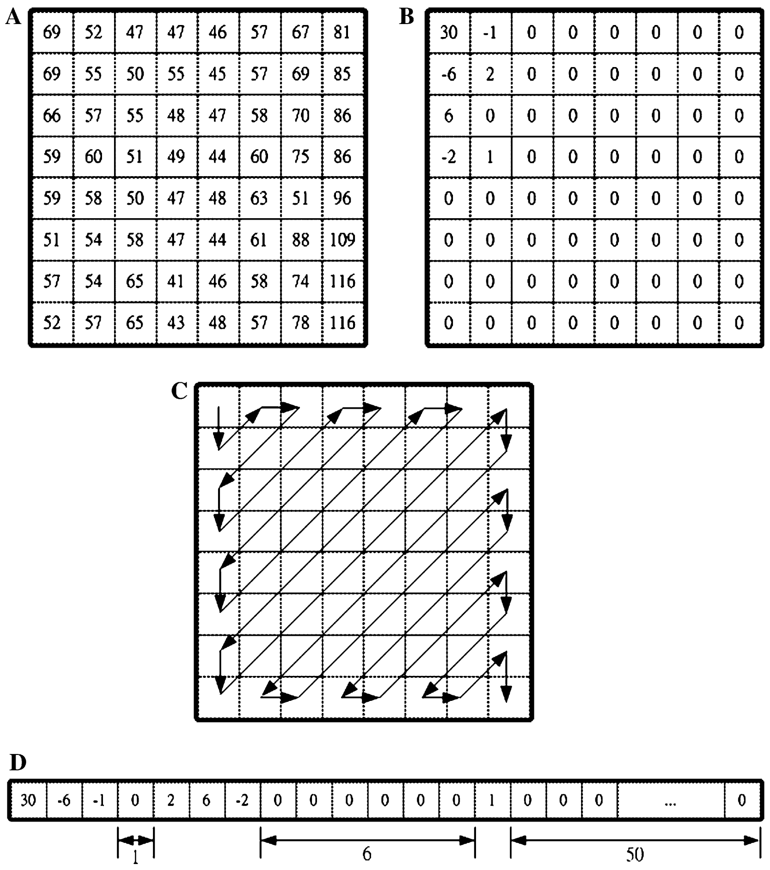


Fig. 5. The approximate bintree structure and partitioned blocks of Fig. 2. (A) The approximate bintree structure of Fig. 2A. (B) The corresponding partitioned blocks of (A).



A

69	52	47	47	46	57	67	81
69	55	50	55	45	57	69	85
66	57	55	48	47	58	70	86
59	60	51	49	44	60	75	86
59	58	50	47	48	63	51	96
51	54	58	47	44	61	88	109
57	54	65	41	46	58	74	116
52	57	65	43	48	57	78	116

B

30	-1	0	0	0	0	0	0
-6	2	0	0	0	0	0	0
6	0	0	0	0	0	0	0
-2	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

D

30	-6	-1	0	2	6	-2	0	0	0	0	0	0	1	0	0	...	0

Fig. 6. An example used in DCT-based coding scheme. (A) The corresponding subimage with respect to node 2 of Fig. 5B. (B) The quantized DCT coefficient matrix of (A). (C) The zig–zag scanning order. (D) The zig–zag scanned sequence of (B).

Linear-Tree Table	:	000100000010111111111
Leaf-Type Table	:	00011010001
Color Table	:	$(1_{ul}1_{ur}1_{bl}1_{br})$
		$(2_{ul}2_{ur}2_{bl}2_{br})$
		$(3_{ul}3_{ur}3_{bl}3_{br})$
		$(4_{d1}4_{d2}4_{d3}\ldots4_{dn_4})$
		...
		$(9_{ul}9_{ur}9_{bl}9_{br})$
		$(10_{ul}10_{ur}10_{bl}10_{br})$
		$(11_{d1}11_{d2}11_{d3}\ldots11_{dn_{11}})$

According to the above description, our proposed SDCT-based image representation is described below:

Step 1. Given a gray image I, we create a node p to represent I. The node p contains following variables.
W denotes the width of the subimage I.
H denotes the height of the subimage I.
(X, Y) denotes the coordination of the subimage I.
isLeaf is a boolean variable to denote whether the node is a leaf or not. The initial value of *isLeaf* is set to **true**.
Step 2. Push the node p into a Queue $Q1$.
Step 3. If $Q1$ is not empty, then $Q1$ pops an element called q and go to Step 4. Otherwise, go to Step 7.
Step 4. According to the Gouraud shading method, Eq. (2) is applied to check the block type of subimage indexed by node q. If the block type of q is nonhomogeneous, then go to Step 5. Otherwise, store the gray levels of its own four corner pixels and push the node q into the queue $Q2$. Subsequently, go to Step 3.
Step 5. If the node q satisfies $W = H$ and $W = 16$, go to Step 6. Otherwise, change the value of isLeaf to **false** and partition the node q into two subimages with equal size in y- and x-axes by turns; the two created subimages are denoted by nodes m and n, respectively. Finally, push the two nodes into the queue $Q1$; push node q into the queue $Q2$ and go to Step 3.
Step 6. For the concerned node, Eq. (3) is applied to generate the DCT coefficient matrix. Based on the quantization table, the quantized DCT coefficient matrix is obtained. According to the zig–zag scanning way, the quantized DCT coefficient matrix is represented as a sequence. The node q is pushed into the queue $Q2$.
Step 7. The content of $Q2$ is indeed the approximate bintree of the gray image I. Popping $Q2$, we can obtain the linear-tree table, the leaf-type table, and the color table.

4. Application: SDCT-based moment computation

In last section, we have presented the proposed SDCT-based image representation. In this section, as a representative of applications, we want to present an efficient algorithm for computing lower order moments on the SDCT-based representation directly. Computing moments on the image is important in image processing field [8,11,25,13,29,27,21,22,26]. The lower order moments are especially useful in several applications such as computing the area, the centroid, and the major axis of a object [11,25], recognizing patterns by moment invariants [13], color image compression [29], moment-preserving thresholding [27], acquiring the motion parameters [21], deskewing rotationally symmetric shapes [22], and color image retrieval [20]. Due to these applications, for convenience, the lower order moments are still called to moments in order to avoid any confusion in this paper. In next section, experimental results will reveal that our proposed SDCT-based moment computation is faster than the previous STC-based moment computation [6] and the DCT-based moment computation.

Quite different from the previous STC-based moment computation algorithm [6], our proposed SDCT-based moment computation algorithm accommodates the moment computations of the STC-based block for homogeneous block and the DCT-based block for nonhomogeneous block.

Next Lemma [6] is used to compute the moments of each STC-based block in $O(1)$ time.

Lemma 1. *For each STC-based block $B_{STC,i}$, the estimated moments can be computed in $O(1)$ time.*

Proof. See Appendix A

After introducing how to compute the moments for each STC-based block in $O(1)$ time, we now describe how to compute the moments on each DCT-based block. According to the IDCT technique, the DCT-based block $\{F(u,v)|0 \leqslant u, v \leqslant b-1\}$ can be transformed into the gray subimage, $\{f(x,y)|0 \leqslant x, y \leqslant b-1\}$:

$$f(x,y) = \frac{1}{\sqrt{2b}} \sum_{u=0}^{b-1} \sum_{v=0}^{b-1} C(u)C(v)F(u,v) \cos\frac{(2x+1)u\pi}{2b} \cos\frac{(2y+1)v\pi}{2b},$$

where $b \times b$ denotes the size of DCT-based block; $C(0) = \frac{1}{\sqrt{2}}$ and $C(i) = 1$, $i = 1, 2, \ldots, b-1$. Instead of running the IDCT on the DCT-based block first and then computing moments on the decompressed subimage, we derive some novel formulas to compute moments on the DCT-based block directly, and the time complexity of our proposed approach is dependent on the number of nonzero terms in that block. For exposition, let b be equal to 16 and the number of DCT-based blocks in the partitioned gray image be k. For each DCT-based block $B_{\mathrm{DCT},i}$ at position (m_i, n_i) for $0 \leqslant i \leqslant k-1$, where we have

$$f(x+m_i, y+n_i) = \frac{1}{\sqrt{32}} \sum_{u=0}^{15} \sum_{v=0}^{15} C(u)C(v)F(u+m_i, v+n_i) \times \cos\frac{(2x+1)u\pi}{32} \cos\frac{(2y+1)v\pi}{32}. \tag{10}$$

From Eq. (10), the computation of m_{pq} for 16×16 DCT-based block at position (m_i, n_i) can be obtained by

$$\begin{aligned}\sum_{x=0}^{15} \sum_{y=0}^{15} (x+m_i)^p (y+n_i)^q f(x+m_i, y+n_i) = {} & \sum_{x=0}^{15} \sum_{y=0}^{15} (x+m_i)^p (y+n_i)^q \frac{1}{\sqrt{32}} \sum_{u=0}^{15} \sum_{v=0}^{15} C(u)C(v) \\ & \times F(u+m_i, v+n_i) \times \cos\frac{(2x+1)u\pi}{32} \cos\frac{(2y+1)v\pi}{32}.\end{aligned}$$

Adjusting the summation notations, the following equation is got

$$\begin{aligned}&\sum_{x=0}^{15} \sum_{y=0}^{15} (x+m_i)^p (y+n_i)^q \frac{1}{\sqrt{32}} \sum_{u=0}^{15} \sum_{v=0}^{15} C(u)C(v)F(u+m_i, v+n_i) \cos\frac{(2x+1)u\pi}{32} \cos\frac{(2y+1)v\pi}{32} \\ &= \frac{1}{\sqrt{32}} \sum_{u=0}^{15} \sum_{v=0}^{15} F(u+m_i, v+n_i) \left[C(u) \sum_{x=0}^{15} (x+m_i)^p \cos\frac{(2x+1)u\pi}{32}\right] \left[C(v) \sum_{y=0}^{15} (y+n_i)^q \cos\frac{(2y+1)v\pi}{32}\right].\end{aligned}$$

Let

$$\begin{aligned}T_p(u+m_i) &= C(u) \sum_{x=0}^{15} (x+m_i)^p \cos\frac{(2x+1)u\pi}{32} \quad \text{and} \\ T_q(v+n_i) &= C(v) \sum_{y=0}^{15} (y+n_i)^q \cos\frac{(2y+1)v\pi}{32},\end{aligned}$$

then the computation of m_{pq} for the 16×16 block at position (i, j) yields

$$\sum_{x=0}^{15} \sum_{y=0}^{15} (x+m_i)^p (y+n_i)^q f(x+m_i, y+n_i) = \frac{1}{\sqrt{32}} \sum_{u=0}^{15} \sum_{v=0}^{15} F(u+m_i, v+n_i) T_p(u+m_i) T_q(v+n_i).$$

From the above derivation, we have the following result.

Lemma 2. *The computational complexity of computing m_{pq} on the DCT-based block depends on the number of nonzero DCT coefficients in the block.*

After describing the proposed methods for computing the moments for the STC-based block and the DCT-based block, respectively, the whole algorithm for computing moments on the SDCT-based image representation is shown below:

Algorithm. SDCT-based Moment Computation.
Input: linear-tree table, leaf-type table, and color table.
Output: approximate moment m_{pq}.

```
Step 1. Initially set $m_{pq} = 0$ and $m_{pq}$ is used to record the approximate moment of SDCT-based image
        representation.
Step 2. If the linear-tree table is not empty, scan it and read one bit, next go to Step 3; otherwise, go to Step 7.
Step 3. If the scanned bit is '0', it means an internal node is scanned and we compute the coordinate and
        the size of the corresponding block for the current scanned internal node according to the BFS
        traversal way; go to Step 2. If the scanned bit is '1', it means that a leaf node is scanned and go
        to Step 4.
Step 4. Scan a bit from the leaf-type table. If the scanned bit is '0', it means that a STC-based block is
        scanned and go to Step 5; otherwise, a DCT-based block is scanned and go to Step 6.
Step 5. According to the scanned entry in the color table, perform the STC-based moment compu-
        tation for that STC-based block. Next, we accumulate the calculated moment to $m_{pq}$ and
        go to Step 2.
Step 6. According to the scanned entry in the color table, perform the DCT-based moment computation
        for that DCT-based block. Next, we accumulate the calculated moment to $m_{pq}$, and go to Step 2.
Step 7. Output the final approximate moment $m_{pq}$.
```

Based on the above description, we have the following result for computing (p, q)-order moment.

Theorem 1. *Suppose the given gray image has been partitioned into K blocks and there are K_1 STC-based blocks and K_2 $(= K - K_1)$ DCT-based blocks. Based on the proposed SDCT-based image representation, the approximate moment m_{pq} can be computed in $O(K_1 + K_2cb^2)$ time where c denotes the expected value of the nonzero DCT's coefficient appearing in the DCT-based block and $b \times b$ denotes the block size.*

In our experiments with four testing images, the average value of c is 0.112; the error tolerance is set to $\epsilon = 20$, and b is set to 16.

Based on experimental data, while the error tolerance ϵ is small, the original STC-based approach decomposes a nonhomogeneous block with high frequency content into many smaller homogeneous blocks and it leads to a memory-redundant side effect. However, our proposed SDCT-based approach uses less nonzero DCT coefficients to represent these STC-based blocks with high frequency content. While the error tolerance ϵ is large, our proposed SDCT-based approach utilizes many STC-based blocks, in which each block is of large size and is represented by four corners, to represent these blocks with large size instead of using DCT-based blocks which need more memory to save DCT coefficients. Experimental results reveal that our proposed SDCT-based approach has a good compromise between the STC-based approach and the DCT-based approach.

5. Experimental results

In this section, four popular gray images, each with size 512×512, are used as the benchmark to evaluate the performance among the concerned methods. The four testing gray images are Lena, F16, Pepper, and Barbara as shown in Figs. 7A–D, respectively. All the concerned experiments are performed on the IBM compatible Pentium IV microprocessor with 1.8 GHZ and 256 MB RAM. The operating system is MS-Windows XP and the program developing environment is Borland C++ Builder 6.0.

Fig. 7. Four testing gray images. (A) Lena. (B) F16. (C) Barbara. (D) Baboon.

5.1. Performance comparison between the proposed SDCT-based representation and the previous STC-based one

First some experimental results are illustrated to show the image compression performance in terms of compression improvement ratio (CIR) and the image quality performance in terms of peak signal-to-noise ratio (PSNR) of our SDCT-based image representation and the previous STC-based image representation. The definition of CIR is given by

$$\mathrm{CIR} = \frac{\mathrm{CR(SDCT)} - \mathrm{CR(STC)}}{\mathrm{CR(STC)}},$$

where CR denotes the compression ratio. The PSNR of a gray image with size $N \times N$ is defined by

$$\mathrm{PSNR} = 10\ \log_{10} \frac{255^2}{\frac{1}{N^2}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}(g(x,y) - g'(x,y))^2},$$

where $g'(x, y)$ denotes the gray level at position (x, y) in the decompressed image. Given four error tolerances $\epsilon = 10$, 20, 30, and 40, Table 1 demonstrates that our proposed SDCT-based approach has better CR than the

Table 1
Compression performance comparison

	STC-based representation				SDCT-based representation			
	$\epsilon = 10$	$\epsilon = 20$	$\epsilon = 30$	$\epsilon = 40$	$\epsilon = 10$	$\epsilon = 20$	$\epsilon = 30$	$\epsilon = 40$
Lena	2.89	5.57	8.98	13.53	3.71	10.12	17.46	25.79
F16	3.13	5.03	7.32	10.08	3.46	6.97	11.32	16.41
Barbara	1.80	2.46	3.20	4.12	2.03	3.78	5.82	8.92
Baboon	1.33	1.96	2.78	3.88	2.23	3.46	4.73	6.54
Average	2.29	3.75	5.57	7.90	2.86	6.08	9.83	14.41
	4.88				8.30			

previous STC-based approach under the same PSNR. Here, the measure CR is defined to be the ratio of 8 over the number of average bits required in the proposed SDCT/STC-based image representation for saving one pixel. Based on the four testing images, the average CR of the previous STC-based image representation is 4.88, but the average CR of the proposed SDCT-based image representation is 8.30. Note that in [5], the encoding phase in the STC-based approach discards the consecutive 1's at the bottom level of linear-tree table while using a variable to save the discarded length. In our implementation, we do not consider this special implementation skill for the STC-based approach and the SDCT-based approach. Using the notation CIR, Table 2 reveals that the average CIR of our proposed SDCT-based image representation over the previous STC-based one is 63.08%. It is observed that the higher the error tolerance is, the better the CIR gets.

From Table 1, the choice of error tolerance ϵ depends on what compression ratio we want. Empirically, if we hope the compression ratio of the SDCT-based image representation to be 9.83 (0.814 bits per pixel) in average, the value of ϵ can be selected to 30; if we hope the compression ratio to be 6.08 (1.32 bits per pixel) in average, the ϵ can be selected to 20. However, under the same ϵ value, the compression ratio may be different for different images.

For one input gray image, as shown in Table 3, based on our proposed SDCT-based gray image representation, the percentage of the area of created STC-based blocks are increased when the error tolerance ϵ grows, but the percentage of the area of created DCT-based blocks are decreased when the error tolerance ϵ grows. It indicates that our proposed SDCT-based gray image representation has a good compromise between the STC-based approach and the DCT-based approach.

Table 2
Compression improvement ratio comparison

	$\epsilon = 10$ (%)	$\epsilon = 20$ (%)	$\epsilon = 30$ (%)	$\epsilon = 40$ (%)
Lena	28.53	81.73	94.38	90.70
F16	10.53	38.51	54.64	62.77
Barbara	12.58	53.72	81.91	116.38
Baboon	68.05	76.69	69.78	68.42
Average	29.92	62.66	75.18	84.57
	63.08 %			

Table 3
Percentages of STC-based blocks and DCT-based blocks used in the SDCT-based approach

	STC-based blocks				DCT-based blocks			
	$\epsilon = 10$ (%)	$\epsilon = 20$ (%)	$\epsilon = 30$ (%)	$\epsilon = 40$ (%)	$\epsilon = 10$ (%)	$\epsilon = 20$ (%)	$\epsilon = 30$ (%)	$\epsilon = 40$ (%)
Lena	15.82	34.47	49.90	60.55	84.18	65.53	50.10	39.45
F16	25.39	42.48	48.54	52.64	74.61	57.52	51.46	47.36
Barbara	7.62	19.92	27.93	34.96	92.38	80.08	72.07	65.04
Baboon	0.00	2.25	7.32	14.26	100.00	97.75	92.68	85.74

Table 4
PSNR comparison

	STC-based representation				SDCT-based representation			
	$\epsilon = 10$	$\epsilon = 20$	$\epsilon = 30$	$\epsilon = 40$	$\epsilon = 10$	$\epsilon = 20$	$\epsilon = 30$	$\epsilon = 40$
Lena	38.52	33.02	29.49	27.17	39.54	35.01	31.29	28.65
F16	39.13	34.14	30.88	28.39	39.66	35.68	32.89	30.30
Barbara	39.85	33.94	29.91	27.16	40.35	37.27	33.56	30.49
Baboon	40.33	32.79	28.81	26.10	43.27	42.17	35.78	30.53
Average	39.45	33.47	29.77	27.20	40.70	37.53	33.38	29.99
		32.48				35.40		

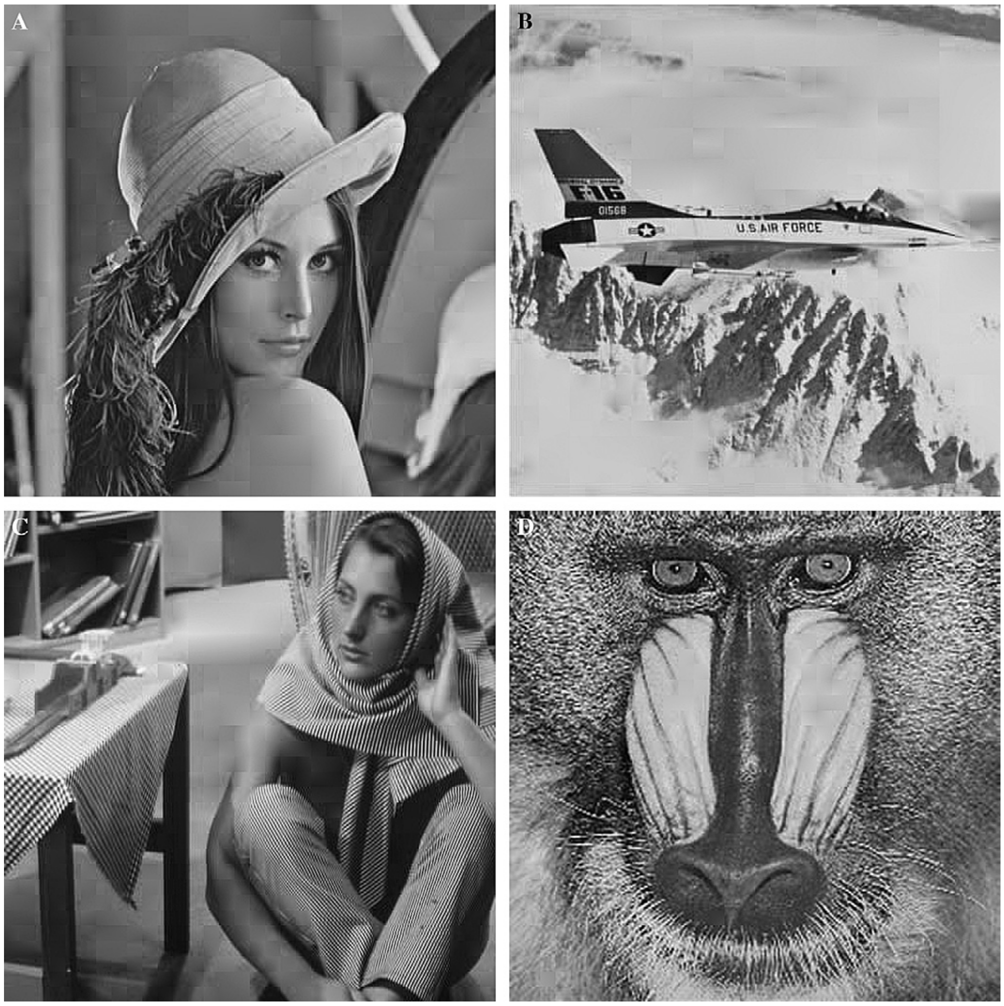


Fig. 8. Four decompressed SDCT-based images for the Fig. 7. (A) Lena. (B) F16. (C) Barbara. (D) Baboon.

Table 4 indicates that our proposed SDCT-based image representation has higher PSNR than the previous STC-based one under the same compression ratio. The average PSNR of the previous STC-based (the proposed SDCT-based) image representation is 32.48 (35.40).

From Table 4, the choice of error tolerance ϵ depends on what PSNR we want. Empirically, if we hope the PSNR of the SDCT-based image representation to be about 37.5 (33.4) in average, the value of ϵ can be selected to 20 (30). However, under the same ϵ value, the PSNR may be different for different images. When the

Table 5
Execution–time performance comparison

	Encoding Time			Decoding Time		
	STC	DCT	SDCT	STC	DCT	SDCT
Lena	62.3	125.0	91.8	18.7	147.0	63.7
F16	57.7	126.5	86.5	20.1	142.3	71.9
Barbara	85.8	129.8	106.5	25.4	145.2	94.9
Baboon	100.8	132.8	121.4	29.4	146.8	120.5
Average	76.7	128.5	101.5	23.4	145.3	87.7

value of ϵ is selected to 20, Fig. 8 depicts the good quality of the four decompressed SDCT-based images from the four testing images of Fig. 7.

The execution–time performance comparison of the three concerned approaches for representing gray images has been carried out in Table 5. Table 5 reveals that our proposed SDCT-based approach has better execution–time performance than the DCT-based approach in both encoding phase and decoding phase. Based on four testing images of Fig. 7, the average encoding time and the average decoding time of the DCT-based approach for representing a gray image are 128.5 ms (milliseconds) and 145.3 ms, respectively; the average encoding time and the average decoding time of SDCT-based approach are 101.5 and 87.7 ms, respectively. However, the SDCT-based approach takes more encoding/decoding time than the STC-based approach whose average encoding time and average decoding time are 76.7 and 23.4 ms, respectively.

5.2. Application: moment computation

In last section, we have presented the algorithm for computing approximate moments on the SDCT-based block directly. Table 6 demonstrates the moment accuracy comparison between the proposed SDCT-based approach and the conventional approach. for $\epsilon = 20$. It is observed that the approximate moments are very close to the exact moments computed by the conventional method running on the original input gray image. In Table 7, the average relative error 0.44% indicates the good accuracy of approximate moments calculated by our proposed SDCT-based approach.

Besides the good accuracy of our proposed SDCT-based approach for computing moments, Table 8 demonstrates that our proposed SDCT-based moment computation has better execution time performance when compared to the original STC-based moment computation and the original DCT-based moment computation. For example, the average execution–time improvement ratio of the proposed SDCT-based moment computation over the STC-based approach is 42.19%. Here, the execution–time improvement ratio is denoted by $\frac{\text{Time(STC)}-\text{Time(SDCT)}}{\text{Time(STC)}}$, where Time(STC) denotes the execution time required by the STC-based moment computation.

Table 6
Moments accuracy comparison

	Conventional approach				Proposed approach			
	Lena	F16	Barbara	Baboon	Lena	F16	Barbara	Baboon
m00	3.25E07	4.68E07	3.08E07	3.37E07	3.23E07	4.67E07	3.06E07	3.36E07
m10	8.67E09	1.22E10	7.73E09	8.64E09	8.62E09	1.22E10	7.70E09	8.60E09
m01	8.05E09	1.18E10	7.36E09	8.70E09	8.00E09	1.17E10	7.32E09	8.67E09
m11	2.19E12	3.05E12	1.87E12	2.19E12	2.18E12	3.04E12	1.86E12	2.19E12
m20	3.01E12	4.24E12	2.58E12	2.96E12	2.99E12	4.23E12	2.57E12	2.95E12
m02	2.70E12	4.02E12	2.37E12	2.98E12	2.68E12	4.01E12	2.35E12	2.97E12
m21	7.71E14	1.06E15	6.33E14	7.46E14	7.67E14	1.06E15	6.29E14	7.43E14
m12	7.38E14	1.03E15	6.13E14	7.44E14	7.34E14	1.02E15	6.09E14	7.41E14
m30	1.16E15	1.65E15	9.74E14	1.14E15	1.15E15	1.65E15	9.69E14	1.14E15
m03	1.02E15	1.55E15	8.74E14	1.15E15	1.02E15	1.55E15	8.69E14	1.15E15

Table 7
Relative error

	Lena (%)	F16 (%)	Barbara (%)	Baboon (%)	Average (%)
m00	0.54	0.24	0.46	0.40	0.41
m10	0.57	0.26	0.46	0.41	0.43
m01	0.57	0.29	0.54	0.38	0.45
m11	0.56	0.32	0.51	0.39	0.45
m20	0.62	0.27	0.45	0.40	0.43
m02	0.53	0.32	0.59	0.37	0.45
m21	0.60	0.35	0.49	0.39	0.46
m12	0.54	0.36	0.54	0.38	0.45
m30	0.65	0.27	0.44	0.39	0.44
m03	0.49	0.32	0.62	0.36	0.45
Average	0.57	0.30	0.51	0.39	0.44

Table 8
Execution-time improvement ratios of the proposed SDCT-based moment computation over the other three approaches

	Lena (%)	F16 (%)	Barbara (%)	Baboon (%)	Average (%)
STC-based approach	34.74	36.92	46.61	50.51	42.19
DCT-based approach	18.51	22.09	10.01	0.73	12.84
Conventional approach	34.79	41.45	37.21	16.60	32.51

6. Conclusions

In this paper, we have presented the proposed novel SDCT-based image representation for representing gray images. Our proposed SDCT-based representation can be viewed as a new hybrid gray image representation which extends the binary image domain to gray image domain. The proposed SDCT-based image representation combines the advantages of STC-based approach for representing homogeneous blocks, each block with low frequency content, and DCT-based approach for representing nonhomogeneous blocks, each block with high frequency content. A new data structure consisting of three tables, namely the linear-tree table, the leaf-type table, and the color table, is presented to realize the proposed novel gray image representation. Under four real gray images, experimental results demonstrate that the proposed SDCT-based image representation has better memory-saving effect when compared to the previous STC-based image representation although the proposed SDCT-based approach takes more encoding/decoding time than the STC-based approach.

Further, the SDCT-based algorithm has been presented for computing moments and experimental results reveal that the proposed SDCT-based algorithm for moment computation has better execution–time performance when compared to the other approaches, such as the STC-based moment computation, DCT-based moment computation, and the conventional moment computation on the input gray image. It is an interesting research issue to apply the results of this paper to design efficient algorithms for different applications, such as region segmentation, window query, image retrieval, and watermark.

Appendix A. The Proof of Lemma 1

For convenience, we suppose that the width and the height of the block B_i are w_i $(= x_{2,i} - x_{1,i} + 1)$ and h_i $(= y_{2,i} - y_{1,i} + 1)$, respectively. From Eq. (1), the estimated gray level at position $(x_{1,i} + x, y_{1,i} + y)$, $0 \leqslant x \leqslant w_i - 1$ and $0 \leqslant y \leqslant h_i - 1$, is calculated by

$$\begin{aligned}
g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) &= g_{\text{est}}(x_{1,i}+x, y_{1,i}) + y \times \frac{g_{\text{est}}(x_{1,i}+x, y_{2,i}) - g_{\text{est}}(x_{1,i}+x, y_{1,i})}{h_i - 1} \\
&= g_{1,i} + x \times \frac{g_{2,i} - g_{1,i}}{w_i - 1} + y \times \frac{g_{3,i} - g_{1,i}}{h_i - 1} + xy \times \frac{g_{1,i} - g_{2,i} - g_{3,i} + g_{4,i}}{(w_i - 1)(h_i - 1)} \\
&= g_{1,i} + x\triangle g_{h,i} + y\triangle g_{v,i} + xyF_{1,i},
\end{aligned} \tag{4}$$

where $\triangle g_{h,i} = \frac{g_{2,i}-g_{1,i}}{w_i-1}$, $\triangle g_{v,i} = \frac{g_{3,i}-g_{1,i}}{h_i-1}$, and $F_{1,i} = \frac{g_{1,i}-g_{2,i}-g_{3,i}+g_{4,i}}{(w_i-1)(h_i-1)}$.

From Eq. (1), The moment equation can be rewritten as

$$m_{pq} = \sum_{i=0}^{K-1} m_{pq,i} = \sum_{i=0}^{K-1} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} (x_{1,i}+x)^p (y_{1,i}+y)^q g_{\text{est}}(x_{1,i}+x, y_{1,i}+y). \tag{5}$$

Then, according to the binomial expansion theorem, the term $m_{pq,i}$ in Eq. (5) can be represented by:

$$\begin{aligned}
m_{pq,i} &= \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} [x_{1,i}^p + \binom{p}{1} x_{1,i}^{p-1} x + \cdots + x^p][y_{1,i}^q + \binom{q}{1} y_{1,i}^{q-1} y + \cdots + y^q] g_{est}(x_{1,i}+x, y_{1,i}+y) \\
&= x_{1,i}^p y_{1,i}^q \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \binom{q}{1} x_{1,i}^p y_{1,i}^{q-1} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} y g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \cdots + x_{1,i}^p \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} y^q g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \binom{p}{1} x_{1,i}^{p-1} y_{1,i}^q \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \binom{p}{1}\binom{q}{1} x_{1,i}^{p-1} y_{1,i}^{q-1} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} xy g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \cdots + \binom{p}{1} x_{1,i}^{p-1} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} xy^q g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \cdots + \cdots + y_{1,i}^q \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^p g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \binom{q}{1} y_{1,i}^{q-1} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^p y g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&\quad + \cdots + \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^p y^q g_{\text{est}}(x_{1,i}+x, y_{1,i}+y).
\end{aligned} \tag{6}$$

By Eq. (4), the kernel computation of each double summation term in Eq. (6) is

$$\begin{aligned}
C_{pq,i} &= \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^p y^q g_{\text{est}}(x_{1,i}+x, y_{1,i}+y) \\
&= g_{1,i} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^p y^q + \triangle g_{h,i} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^{p+1} y^q \\
&\quad + \triangle g_{v,i} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^p y^{q+1} + F_{1,i} \sum_{x=0}^{w_i-1} \sum_{y=0}^{h_i-1} x^{p+1} y^{q+1}.
\end{aligned} \tag{7}$$

Using the following five closed forms, the summation term $c^p_{0,w_i-1} = \sum_{x=0}^{w_i-1} x^p$ for a specific p, $0 \leqslant p \leqslant 4$, can be computed in $O(1)$ time. The summation term $c^q_{0,h_i-1} = \sum_{y=0}^{h_i-1} y^q$ is the same.

$$\begin{aligned}
c^0_{0,w_i-1} &= \sum_{x=0}^{w_i-1} x^0 = w_i,\\
c^1_{0,w_i-1} &= \sum_{x=0}^{w_i-1} x^1 = \frac{w_i(w_i-1)}{2},\\
c^2_{0,w_i-1} &= \sum_{x=0}^{w_i-1} x^2 = \frac{w_i(w_i-1)(2w_i-1)}{6},\\
c^3_{0,w_i-1} &= \sum_{x=0}^{w_i-1} x^3 = \frac{w_i^2(w_i-1)^2}{4},\\
c^4_{0,w_i-1} &= \sum_{x=0}^{w_i-1} x^4 = \frac{w_i(w_i-1)(2w_i-1)(3w_i^2-3w_i-1)}{30}.
\end{aligned} \tag{8}$$

So, Eq. (7) can be rewritten as

$$C_{pq,i} = g_{1,i} c^p_{0,w_i-1} c^q_{0,h_i-1} + \triangle g_{h,i} c^{p+1}_{0,w_i-1} c^q_{0,h_i-1} + \triangle g_{v,i} c^p_{0,w_i-1} c^{q+1}_{0,h_i-1} + F_{1,i} c^{p+1}_{0,w_i-1} c^{q+1}_{0,h_i-1}. \tag{9}$$

Eq. (9) implies that the computation of the term $C_{pq,i}$ can be finished in $O(1)$ time, and the estimated moments $m_{pq,i}$, $0 \leqslant p+q \leqslant 3$, can also be calculated in $O(1)$ time. This completes the proof.

References

[1] Y.K. Chan, C.C. Chang, Block image retrieval based on a compressed linear quadtree, Image Vision Comput. 22 (5) (2003) 391–397.
[2] C.C. Chang, J.C. Chuang, C.Y. Chung, Quadtree-segmented image compression method using vector quantization and cubic B-spline interpolation, Imaging Sci. J. 52 (2) (2004) 106–116.
[3] Z. Chen, I.P. Chen, A simple recursive method for converting a chain code into a quadtree with a lookup table, Image Vision Comput. 19 (7) (2001) 413–426.
[4] P.M. Chen, Variant code transformations for linear quadtrees, Pattern Recogn. Lett. 23 (11) (2002) 1253–1262.
[5] K.L. Chung, J.G. Wu, Improved image compression using S-tree and shading approach, IEEE Trans. Commun. 48 (5) (2000) 748–751.
[6] K.L. Chung, P.C. Chen, An efficient algorithm for computing moments on a block representation of a grey-scale image, Pattern Recogn. 38 (12) (2005) 2578–2586.
[7] R. Distasi, M. Nappi, S. Vitulano, Image compression by B-tree triangular coding, IEEE Trans. Commun. 45 (9) (1997) 1095–1100.
[8] J. Flusser, Refined moment calculation using image block representation, IEEE Trans. Image Processing 9 (11) (2000) 1977–1978.
[9] J.D. Foley, A.V. Dam, S.K. Feiner, J.F. Hughes, Computer Graphics, Principle, and Practice, second ed., Addison-Wesley, Reading, MA, 1990.
[10] I. Gargantini, An effective way to represent quadtrees, Commun. ACM 25 (12) (1982) 905–910.
[11] R.C. Gonzalcz, R.E. Woods, Digital Image Processing, second ed., Prentice Hall, New York, 2002.
[12] P.G. Howard, F. Kossentini, B. Martins, S. Forchhammer, W.J. Rucklidge, The emerging JBIG2 standard, IEEE Trans. Circ. Syst. Video Technol. 8 (7) (1998) 838–848.
[13] M.K. Hu, Visual pattern recognition by moment invariants, IRE Trans. Inform. Theor. 8 (2) (1962) 179–187.
[14] W.D. Jonge, P. Scheuermann, A. Schijf, S+-trees: an structure for the representation of large pictures, Comput. Vision Image Understanding 59 (3) (1994) 265–280.
[15] E. Kawaguchi, T. Endo, On a method of binary picture representation and its application to data compression, IEEE Trans. Pattern Anal. Mach. Intell. 2 (1) (1980) 27–35.
[16] C. Kachris, N. Bourbakis, A. Dollas, A reconfigurable logic-based processor for the SCAN image and video encryption algorithm, Int. J. Parallel Prog. 31 (6) (2003) 489–506.
[17] X. Wan, C.-C.J. Kuo, A new approach to image retrieval with hierarchical color clustering, IEEE Trans. Circ. Syst. Video Technol. 8 (5) (1998) 628–643.
[18] T.W. Lin, Compressed linear quadtree representations for storing similar images, Image Vision Comput. 15 (11) (1997) 833–843.
[19] T.W. Lin, Set operations on the constant bit-length linear quadtree, Pattern Recogn. 30 (7) (1997) 1239–1249.
[20] G. Paschos, I. Radev, N. Prabakar, Image content-based retrieval using chromaticity moments, IEEE Trans. Knowl. Data Eng. 15 (5) (2003) 1069–1072.
[21] S.C. Pei, L.G. Liou, Using moments to acquire the motion parameters of a deformable object without correspondences, Image Vision Comput. 12 (8) (1994) 475–485.

[22] S.C. Pei, J.H. Horng, A moment-based approach for deskewing rotationally symmetric shapes, IEEE Trans. Image Processing 8 (12) (1999) 1831–1834.
[23] H. Samet, The Design and Analysis of Spatial Data Structures, Addison-Wesley, New York, 1990.
[24] H. Samet, Application of Spatial Data Structure, Addison-Wesley, New York, 1990.
[25] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, second ed., PWS, New York, 1998.
[26] I.M. Spiliotis, B.G. Mertzios, Real time computation of two-dimensional moments on binary images using image block representation, IEEE Trans. Image Processing 7 (11) (1998) 1609–1615.
[27] W.H. Tsai, Moment-preserving thresholding: a new approach, Comput. Vision Graph. Image Processing 29 (3) (1985) 377–393.
[28] G.K. Wallace, The JPEG still picture compression standard, Commun. ACM 34 (4) (1991) 30–44.
[29] C.K. Yang, J.C. Lin, W.H. Tsai, Color image compression by moment-preserving and block truncation coding techniques, IEEE Trans. Commun. 45 (12) (1997) 1513–1516.

Kuo-Liang Chung received the B.S., M.S., and Ph.D. degrees in Computer Science and Information Engineering from National Taiwan University, Taipei, Taiwan, in 1982, 1984, and 1990, respectively. From 1984 to 1986, he completed the military service. From 1986 to 1987, he was a research assistant in the Institute of Information Science, Academic Sinica. He has been a Professor in the Department of Computer Science and Information Engineering at National Taiwan University of Science and Technology since 1995. Now he is the chair in the same department. Prof. Chung received the distinguished research award (2004–2007) from the National Science Council, Taiwan. He is also an IEEE senior member. His research interests include image compression, image processing, video compression, pattern recognition, coding theory, algorithms, and multimedia applications.

Yau-Wen Liu received the B.S. degree in Computer Science and Information Engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 2004. He is currently a master student in Computer Science and Information Engineering at National Taiwan University, Taipei, Taiwan. His research interests include image/video compression, image processing, pattern recognition, computer vision, and algorithms.

Wen-Ming Yan received the B.S. and M.S. degrees in Mathematics from National Taiwan University, Taipei, Taiwan. Now he is an Associate Professor in Computer Science and Information Engineering at National Taiwan University. Prof. Yan's research interests include scientific computation, image compression, computer vision, numerical linear algebra, image processing, coding theory, and algorithms.