

A Low Complexity 2-Power Transform for Video Compression

Shan Lu¹, Keman Yu², Jiang Li² and Shipeng Li²

¹University of Science and Technology of China

²Microsoft Research Asia

Abstract

Video applications in mobile devices call for low complexity video compression algorithms. This paper presents a low complexity 2-power transform for video compression. The elements of the transform matrix are all in 2's low-order power. The transform computation requires only 28 additions and 10 binary shifts. Computational complexity analysis, coding gains and RD curve comparison show that our 2-power transform is simpler than IntDCT and ICT, while its compression efficiency is similar to that of IntDCT, higher than that of ICT and much higher than that of DT and WHT. The low magnitude of the elements also avoids large memory consumption, a problem of many integer transforms.

1. Introduction

Transformation is an essential component of image and video coding. Among many transformation kernels, Discrete Cosine Transform (DCT) is the optimal image-independent one, and adopted by most compression standards [1]. For video compression in devices with weak computational power, the complexity of DCT is still high, so integer alternatives for DCT are pursued. Ideally, the alternatives should have low computational complexity and high compression efficiency. To achieve this goal, many researchers have been working on various simplified transforms.

There are two approaches in getting a simplified transform: the factorization based approach and the integer transform kernel redesign approach. The first approach operates similarly to most fast algorithms for DCT. They factorize DCT to the multiple of a Walsh matrix and some sparse matrices, and then approximate the floating-point coefficients of the latter by integer or binary fractions. The C-matrix transform (CT) [2] and the recently popular Integer DCT (IntDCT) with lifting scheme [3] belong to this approach. These transforms resemble the original DCT closely and often provides high compression. The drawback is that they divide the original transform into several steps: most commonly, a Walsh-Hadamard transform and some sparse matrices. Although sparse matrices can be computed by fast algorithms, the total complexity is always a multiple of WHT, which is hard to reduce. The second approach directly designs a simple matrix to replace the floating-point DCT. This approach

gives researchers much freedom except for the only constraint of matrix orthogonality. The challenge of this approach is that high compression efficiency is hard to maintain.

There are two methods in the integer transform kernel redesign approach. One starts from scaling the original DCT by a large integer and then searches for integer coefficients with respect to orthogonality restrictions [4] [5]. Its drawback is that the elements of the matrix elements are often large integers. The other designs a new symmetric and orthogonal matrix template and then produces a transform family. Representative results of the second method are Cham's integer cosine transform (ICT) [6] and the dyadic transform (DT) [7] family. These transforms can achieve quite low complexity but their compression efficiency is still not satisfactory. The matrix of DT implements a 1-D 8-point transform with only 28 additions plus 10 binary shifts, but it is completely incompatible with DCT. The ICT is better. Its representative ICT (5,3,2,1) is a little more complex than DT but its compression efficiency is listed between that of IntDCT with lifting scheme and CT.

This paper designs a transform matrix with all its elements composed of 2's low-order power. Thus the computational complexity is fairly low (as low as DT). Meanwhile, many properties of DCT such as orthogonality are reserved. Its compression efficiency is similar to that of IntDCT and its theoretical coding gain is only about 0.11dB less than DCT.

The rest of the paper is organized as follows. In section 2, we review the concept of orthogonality to clarify the basic restrictions we must obey in the transform matrix design, and then we start from the matrix template of DCT, make some modifications and search for 2-power solutions. In Section 3, we compare the computational complexity and compression efficiency of the method with those of several typical transforms. We conclude our work in Section 4.

2. Transform Design

2.1 Analyzing Orthogonality Restrictions

Orthogonality is a basic property of a transform kernel. Suppose a matrix T can be written as $T = [s_0V_0, s_1V_1, K, s_7V_7]^T$, where $V_0 \wedge V_7$ are row vectors or

basis with unity magnitude, and $s_0 \wedge s_7$ are scaling factors. The orthogonality of a matrix means: 1) V_i is orthogonal to each other; 2) $s_0 = s_1 = \Lambda = s_7 = 1$. The problem is that it is impossible for an integer transform matrix to satisfy the second constraint.

To solve the problem, we can perform the scaling factors together with the quantization process, so the second constraint can be loosened.

Suppose s_0, \dots, s_7 are of arbitrary values and V_0, \dots, V_7 are orthogonal to each other. If we denote $\text{diag}(s_0, \dots, s_7)$ by S and $\text{diag}(s_0^{-1}, \dots, s_7^{-1})$ by S^{-1} , then $S^{-1}T$ is an orthogonal matrix, and an orthogonal transform between a matrix X and a matrix Y can be written as $Y = T'S^{-1}XS^{-1}T$. Considering that S^{-1} is a diagonal matrix, the value of $S^{-1}XS^{-1}$ for arbitrary matrix X can be written as $X \otimes C$, where $c_{i,j} = s_i^{-1} \cdot s_j^{-1}$ and \otimes denotes element-by-element multiplication instead of normal matrix multiplication. The non-unity scaling factors' influence is reflected in this \otimes operation, which can be integrated into the quantization process. In the quantization process, for each (i, j) position of a matrix, we use $(c_{i,j}, 2c_{i,j}, \dots, 31c_{i,j})$ to replace the original $(1, \dots, 31)$ quantization factor table. The tables for the inverse transform can be similarly designed. The computational cost of quantization would not increase, while only more memory is needed to store the additional tables. In our method, six additional arrays are needed for the transform.

In summary, in the design of the transform matrix, the second constraint of orthogonality is unnecessary, and can be met in the quantization process. Thus we only need to make the row vectors of a matrix orthogonal to each other.

2.2 Finding Appropriate Matrix Templates

Our target is to design a transform with low computational complexity and high compression efficiency. To meet the low complexity criterion, we consider expressing all the elements of the matrix with powers of 2, thus each multiplication can be replaced by a binary shift. If the kernel further possesses the symmetric property, the complexity is expected to be very low. Regarding compression efficiency, we assume that the more a matrix approximates DCT, the higher compression efficiency it possesses. Thus we will start from a DCT template. If the original template is proved not to have any 2-power solution, we will make some modifications to the template.

The following matrix T_1 is a commonly used DCT template, which preserves all the signs, equality relations and inequality relations among its elements:

$$T_1 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a & b & c & d & -d & -c & -b & -a \\ e & f & -f & -e & -e & -f & f & e \\ b & -d & -a & -c & c & a & d & -b \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ c & -a & d & b & -b & -d & a & -c \\ f & -e & e & -f & -f & e & -e & f \\ d & -c & b & -a & a & -b & c & -d \end{pmatrix} \quad (1)$$

where

$$a \geq b \geq c \geq d \text{ and } e \geq f. \quad (2)$$

The above matrix meets the orthogonality constraints of Section 2.1 if a, b, c and d satisfy Eq. (3):

$$ab = ac + cd + bd. \quad (3)$$

We intend to design a matrix, which satisfies conditions (2), (3) and (4)

$$a, b, c, d, e, f \text{ are powers of 2} \quad (4)$$

Eq. (3) can be written as:

$$\frac{a}{d} + 1 + \frac{b}{c} = \frac{ab}{cd} \quad (5)$$

In Eq. (5), if $b \neq c$, then since $a > d$ and $b > c$, $\frac{a}{d}$ and $\frac{b}{c}$ must be even. The left part of equation is odd and the right part of the equation is even, giving an equation impossible to satisfy. If $b = c$, the left part of the equation equals $\frac{a}{d} + 2$, while the right part of the equation equals $\frac{a}{d}$. This equation also cannot hold. So there are no 2-power solutions for the above DCT template.

In order to find a template which is mostly like that of DCT and possesses 2-power number coefficients, we have to make some modifications. We hope the modified template possesses the property that it naturally meets the orthogonality constraint. The following matrix T_2 is obtained by only changing some orders and signs in row 3 and row 5 of the DCT template.

$$T_2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a & b & c & d & -d & -c & -b & -a \\ e & f & -f & -e & -e & -f & f & e \\ c & d & -a & -b & b & a & -d & -c \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ b & -a & -d & c & -c & d & a & -b \\ f & -e & e & -f & -f & e & -e & f \\ d & -c & b & -a & a & -b & c & -d \end{pmatrix} \quad (6)$$

It should be noted that no matter what values the coefficients of $a, b, c, d, e,$ and f of T_2 are chosen, the row vectors of the matrix are always orthogonal.

2.3 Fill Template with Power of 2 numbers

The following criteria are set when we search for 2 power numbers for the coefficients of $a, b, c, d, e,$ and f :

- 1) keeping the conditions of $a \geq b \geq c \geq d$ and $e \geq f$;
- 2) making the ratios of the coefficients as approximate as possible to those of the DCT.
- 3) low order powers are preferred.

Although the complexity of binary shifts would not vary much between high 2-power number and low 2-power numbers, larger numbers would produce large intermediate results and require more bits in storage. In addition, although large integer pairs may approximate the ratio of two floating-point coefficients better, for 2-power numbers, its accuracy is the same as that of small ones.

The commonly used DCT matrix is constructed using the following set of floating-point values of coefficients $a, b, c, d, e,$ and f :

$$0.9808, 0.8315, 0.5556, 0.1951, 0.9239, 0.3827$$

i.e. if we insert the above values into matrix T_1 , T_1 is exactly the commonly used DCT. As we stated in the above criteria, we want to find a matrix T_2 with 2-power coefficients that are as close to those of DCT as possible. We choose

$$a = b = 2, c = 1, d = \frac{1}{4}, e = 2, f = 1 \quad (7)$$

Note that here d is not an integer, but its multiplication can be implemented by two-bit right shift. In fact, $(2, 2, 1, \frac{1}{2}, 2, 1)$ is also a good set. Its corresponding matrix possesses a similar computational complexity as that of (7), but its compression efficiency is a little worse. Finally we obtain our 2-power transform matrix:

$$T_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 1 & 2^{-2} & -2^{-2} & -1 & -2 & -2 \\ 2 & 1 & -1 & -2 & -2 & -1 & 1 & 2 \\ 1 & 2^{-2} & -2 & -2 & 2 & 2 & -2^{-2} & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 2 & -2 & -2^{-2} & 1 & -1 & 2^{-2} & 2 & -2 \\ 1 & -2 & 2 & -1 & -1 & 2 & -2 & 1 \\ 2^{-2} & -1 & 2 & -2 & 2 & -2 & 1 & -2^{-2} \end{pmatrix} \quad (8)$$

2.4 Analyzing Computational Complexity

The 2-power matrix is symmetric like that of DCT. Its simple coefficients greatly simplify the multiplication

operations. It requires only 28 addition and 10 binary shift operations to compute a 1-D order-8 transform.

Table 1. Computational Complexity Comparison

	IntDCT	ICT	DT	2-Pow	WHT
Addition	45	34	28	28	24
Binary Shift	18	10	10	10	0

We list some representative integer transforms in Table 1. Our 2-Power transform is among the simplest. The complexity of WHT listed in the last column represents the lower limit of integer transforms. Considering that using purely '1' can avoid binary shifts, the complexity of 2-Power is very close to the lower limit.

The flow diagram of the 2-power transform is shown in Fig.1.

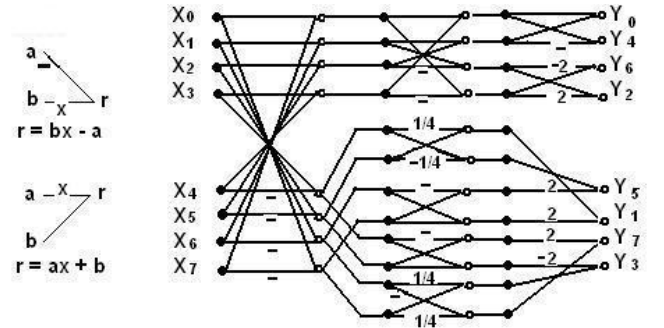


Figure 1: Flow diagram of 2-Power Transform.

3. Performance Evaluation

In this section, both statistical models and real images are used to evaluate the performance of our 2-power transform in video coding applications.

We compare the performance of our 2-power transform with that of four representative integer transforms. The first is IntDCT [3], which is popular in recent years; The second is ICT(5,3,2,1) [6], which is recommended in Cham's ICT family. The third is the DTb8[7], which is in Cham's DT family and has equal complexity as ours. The fourth is WHT. These transforms represent two major simplified transform families and the former two have already been used in some practical systems.

3.1 Coding Gain

Coding gain is a widely used measure of compression efficiency. If we model the input signal by an AR (1) process, the coding gain of a given transform can be calculated analytically. The AR (1) process is characterized by the correlation coefficient ρ . The comparison results are shown in Table 2.

Table 2. Coding Gain (dB) of Different Transforms

ρ	DCT	2-Pow	ICT	DT	WHT
0.95	8.83	8.70	8.65	8.40	7.95
0.90	6.28	6.16	6.11	5.88	5.50
0.85	4.83	4.73	4.67	4.46	4.15
0.80	3.83	3.75	3.69	3.50	3.25

It can be seen that our 2-Power transform produces higher coding gains than those of ICT (5,3,2,1) and much higher coding gains than those of DT, which have the same complexity as our method. Our 2-Power transform is only about 0.11dB lower than that of DCT. Since IntDCT is implemented by several steps rather than in one transform kernel, its coding gains are not compared here.

3.2 RD Curves

We implement the above transform modules in the H.263+ standard codec based on TMN8. We use two video sequences, Foreman and Silent. The Rate-Distortion (RD) curves are shown in Fig. 2 and 3.

In Fig. 2, The RD curve of the 2-Power transform (dashed line with circles) almost overlap with that of IntDCT (dashed line with crosses). Both of them are better than ICT (solid line with triangles) and worse than DCT. A similar trend appears in Fig. 3. At low bit rates, our 2-Power transform performs equally to DCT while the gap is enlarged at high bit rates. Regarding DTb8, although it is also a 2-power transform, its RD Curves are much lower. Please note that since the compression efficiency of WHT, which possesses the lowest computational complexity, is even much worse than that of DT, we do not list its RD curves here. In general, the PSNR difference between 2-Power and DCT is about less than 0.11dB.

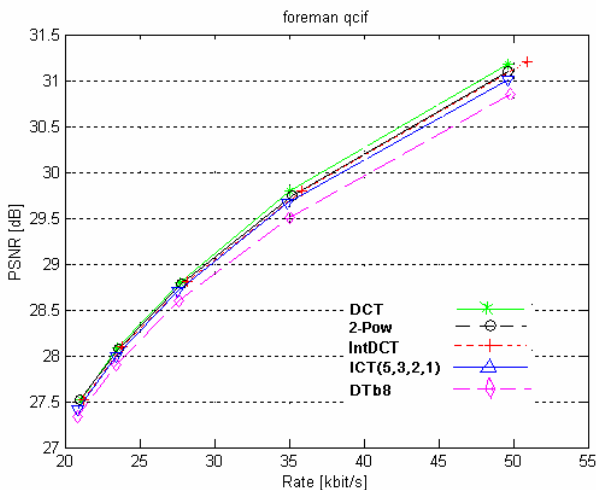


Figure 2: Rate/Distortion Plot of Foreman sequence.

From computational complexity analysis, coding gains and RD curves comparison, we can see that our 2-Power transform is simpler than IntDCT and ICT, while its

efficiency is similar to that of IntDCT, higher than that of ICT and much higher than that of DT.

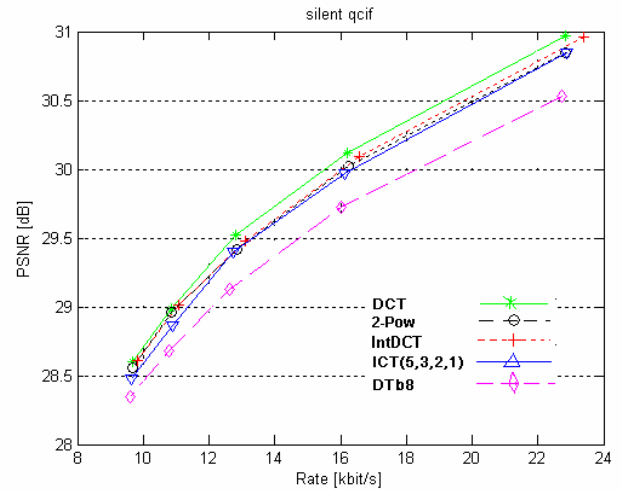


Figure 3. Rate/Distortion Plot of Silent sequence.

4. Conclusions

This paper presents a low complexity 2-power transform for image and video compression. The elements of the transform matrix are all in 2's low-order power. The operations of the transform only require 28 additions and 10 binary shifts.

We analyze the orthogonality condition in the design of a transform matrix and then move normalization to the quantization process. Since there is not a 2-power number solution in the DCT template, we modify some rows of the matrix and get a matrix that is very similar to that of DCT and possesses 2-power number elements. Computational complexity analysis shows that our 2-power transform is simpler than IntDCT and ICT which require 45 additions and 18 shifts, and 34 additions and 10 shifts respectively. Experiments on coding gains and RD curves confirm that our 2-power transform performs similarly to IntDCT, better than ICT, and much better than DT and WHT. The low magnitude of the elements of the matrix also avoids significant memory consumption, which is a problem of many integer transforms. Our 2-power transform may be a good alternative to transforms in H.26L Low-complexity mode [8].

References

- [1] V. Bhaskaran and K. Konstantinides, Image and Video Compression Standards, Algorithms and Architectures, Second Edition, Kluwer Academic Publishers, Boston/Dordrecht/London, 1997.
- [2] H.W. Jones, D. N. Hein, and S. C. Knauer, "The Karhunen-Loeve Discrete Cosine and Related Transforms

Obtained via the Hadamard Transform,” Proc. Intl. Telemetering Conference, Los Angeles, Nov. 1978, 14, pp. 87-98.

[3] Ying-Jui Chen, Soontorn Oraintara, and Truong Nguyen, “Integer Discrete Cosine Transform (IntDCT),” IEEE Trans. Signal Processing, Feb. 2000, pp. 1-5.

[4] Gisle Bjontegaard, “Addition of 8x8 Transform to H.26L”, ITU-T Q15/SG16, Document Q15-I-39, Red Bank, NJ, Oct. 2000.

[5] Mathias Wien, Claudia Mayer, “Integer Transforms for H.26L using Adaptive Block Transforms,” ITU-T

Q15/SG16, Document Q15-K-24, Portland, Oregon, Aug. 2000.

[6] W. -K. Cham, “Development of integer cosine transforms by the principle of dyadic symmetry,” IEE Proceedings, Vol.136, Pt.1, No.4, Aug. 1989.

[7] K. -T. Lo, and W. -K. Cham, “Development of simple orthogonal transforms for image compression,” IEE Proc.-Vis. Image Signal Process., Vol. 142, No.1, February 1995.

[8] Gisle Bjontegaard, “H.26L Test Model Long Term Number 8 (TML-8) draft0”, ITU-T Q.6/SG16, VCEG-N10, Santa Barbara, CA, Sep. 2001.