DESIGN OF NON-SEPARABLE TRANSFORMS FOR DIRECTIONAL 2-D SOURCES

Haoming Chen, Shuyuan Zhu, and Bing Zeng

Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology
Clearwater Bay, Kowloon, Hong Kong, China
{eehmchen, eezhshy, eezeng@ust.hk}

ABSTRACT

Traditionally, a 2-D block-based transform is always implemented through two separate 1-D transforms along each block's vertical and horizontal dimensions. Such a framework is however not highly suitable for a directional 2-D source in which the dominant directional information is neither horizontal nor vertical. On the other hand, the R-D performance upper bound for all block-based transform coding schemes applied on such 2-D directional sources can be obtained by the non-separable Karhunen-Loève transform (KLT) – which is unfortunately very expensive computationally. In this paper, we present a new framework for designing some non-separable transforms that offer an R-D performance closer to that of the KLT, but can be implemented with nearly the same complexity as that of the discrete cosine transform (DCT).

1. INTRODUCTION

For an N-point random vector $\boldsymbol{x}=[x_0,\ldots,x_{N-1}]^t$ with covariance matrix $\boldsymbol{R}_x=[r_x(i,j)]_{N\times N}$, we know that the so-called optimal Karhunen-Loève transform (KLT) can be derived from the eigen vectors of $\boldsymbol{R}_x$. However, the KLT has some serious drawbacks. First, it is time-consuming to do the eigenvector decomposition on $\boldsymbol{R}_x$. Second, no fast implementation can be derived for the KLT so that its implementation complexity (measured by the number of multiplications) is $O(N^2)$.

In many practical cases, $\boldsymbol{R}_x$ can be modeled by a Toeplitz-type matrix with

$$r_x(i,j)=\rho^{|i-j|}, i,j=0,1,\cdots,N-1, |\rho|<1. \quad (1)$$

When $\rho\to 1$ (indicating a very strong inter-pixel correlation), it has been proven that the discrete cosine transform (DCT) can approximate the KLT closely [1], [2]. This DCT can be implemented in a fast way with complexity $O(2N\cdot\log_2 N)$. Moreover, its complexity can be further reduced greatly, e.g., only 13 multiplications are needed when $N=8$ [3].

In the video coding scenario, we know that the DCT is often applied on some residual signals that are obtained after the motion-compensation (in a P-frame) or intra-prediction (in an I-frame). In either case, the inter-pixel correlation become much weaker or even negative so that the DCT surely will not work as efficiently as one expects.

In the 2-D case with $\boldsymbol{x}=[x_{i,j}]_{N\times N}$, the correlation between two pixels A and B at locations (p_A, q_A) and (p_B, q_B) can be modeled as

$$r_x(A,B)=\rho^{\sqrt{(p_A-p_B)^2+(q_A-q_B)^2}}, 0\le p_A, q_A, p_B, q_B < N. \quad (2)$$

More generally, the 2-D vector $\boldsymbol{x}$ may have a dominating orientation. To model this scenario, we need a new covariance matrix with

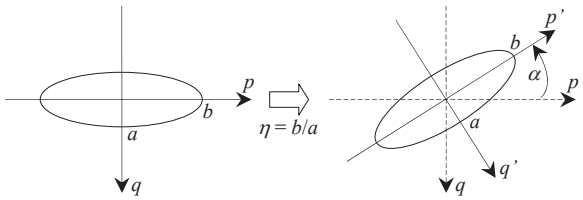


Fig. 1. Elliptical function and its rotated version used to model 2-D directional sources

$$r_x(A,B)=\rho^{\sqrt{d_1^2(\alpha)+\eta^2\cdot d_2^2(\alpha)}}, \quad (3)$$

where $\eta = b/a \ge 1$ represents the ratio of long and short radius of an elliptical function and

$$\begin{cases} d_1(\alpha)=(p_A-p_B)\cos\alpha+(q_A-q_B)\sin\alpha \\ d_2(\alpha)=(q_A-q_B)\cos\alpha-(p_A-p_B)\sin\alpha \end{cases} \quad (4)$$

Compared with Eq. (2) that assumes a circular function, Eq. (3) uses an elliptical function rotated by an angle α, as shown in Fig. 1. Clearly, a bigger η indicates a stronger directionality along the angle θ, and two models become identical if $\eta=1$.

After concatenating all columns of $\boldsymbol{x}$ into a tall vector of length N^2, a big $N^2\times N^2$ covariance matrix $\boldsymbol{R}_x$ can be derived from Eq. (3). Then, one can derive the 2-D KLT through the eigenvector decomposition. Obviously, this KLT is non-separable and sets up the upper bound for the R-D performance for all block-based transform coding schemes. This work has been done by us recently, see [5]. As we said in the beginning, such an optimal KLT is very expensive computationally. For instance, for a 4×4 block, it needs 16×16=256 multiplications, whereas the corresponding 2-D separate DCT needs only 32 multiplications (4 in each 4-point DCT). The goal of this paper is to design some non-separable transforms that can be implemented with nearly the same complexity as that of the separate 2-D DCT and, at the same time, will offer a better R-D performance (thus getting closer to that of the KLT).

2. GIVENS ROTATIONS AND BUTTERFLY STRUCTURES

Any transform matrix used for image/video coding is usually a unitary one. In principle, a unitary matrix can be factorized into a product of multiple Givens rotations [4], whereas each such rotation can be implemented as a simple butterfly-type flow graph structure, see Fig. 2. Taking the 8-point DCT for example, its transform matrix can be decomposed into a total number of 13 rotations [3]. Moreover, 10 out of them have angle $\theta=\pi/4$ and the other three angles are $3\pi/8$, $3\pi/16$, and $7\pi/16$, respectively.

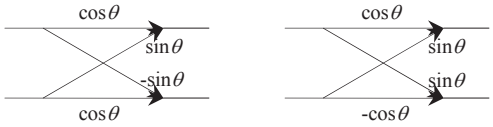


Fig. 2. Two butterfly structures to implement one rotation

Intuitively, one can imagine that, fixed the butterfly structure, other rotation angles can also be adopted so as to obtain different transforms. Our task in this section is to analyze a generic butterfly structure: how to determine the rotation angles so as to maximize the coding gain.

A. Optimal angle for one Givens rotation

Let us start from the simplest case – one Givens rotation standing alone. Referring to Fig. 2(a), the node-variables before and after the rotation is related as

$$\begin{bmatrix} X_0 \\ X_1 \end{bmatrix} = \boldsymbol{C}\begin{bmatrix} x_0 \\ x_1 \end{bmatrix},\ \boldsymbol{C} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \theta \in [0, \pi/2] \tag{5}$$

Let us denote the co-variance matrix of $[x_0\ x_1]^T$ as $\boldsymbol{r}=[r_{i,j}]$, and the co-variance matrix of $[X_0\ X_1]^T$ as $\boldsymbol{R}=[R_{i,j}]$, $i,j=0$ or 1. Then, we have the following result:

$$\boldsymbol{R} = \boldsymbol{C}\cdot\boldsymbol{r}\cdot\boldsymbol{C}^T \tag{6}$$

It is easy to know that the diagonal elements of $\boldsymbol{R}$ represent the variances of transform coefficients. Based on them, we can compute the coding gain as follows:

$$G = -\frac{1}{2\times 2}\sum_{l=0}^{1}\log_2(R_{l,l}). \tag{7}$$

It can be shown that maximizing the coding gain G is equivalent to minimizing the product of $R_{0,0}$ and $R_{1,1}$. Substituting $\boldsymbol{C}$ into Eq. (6), we get

$$\begin{cases} R_{0,0} = r_{0,0}\cos^2\theta + r_{1,1}\sin^2\theta + (r_{0,1}+r_{1,0})\sin\theta\cos\theta \\ R_{1,1} = r_{0,0}\sin^2\theta + r_{1,1}\cos^2\theta - (r_{0,1}+r_{1,0})\sin\theta\cos\theta \end{cases} \tag{8}$$

It is interesting to note from Eq. (8) that $R_{0,0}+R_{1,1}=r_{0,0}+r_{1,1}=$ constant! This is not surprising because the Givens rotation shown in Fig. 2(a) is an energy-preserving one. Because of this property, the minimal product of $R_{0,0}$ and $R_{1,1}$ can be obtained when $R_{0,0}$ or $R_{1,1}$ reaches its extreme value.

Now, let's focus on $R_{0,0}$ (the same result can be obtained by focusing on $R_{1,1}$). By taking the partial derivative of $R_{0,0}$ (with respect to θ) and setting it to zero, we can determine the "best" angle θ as follows:

$$\theta = \begin{cases} \phi/2 & \text{if } (r_{0,0}-r_{1,1})\cdot(r_{0,1}+r_{1,0}) \ge 0 \\ (\pi-\phi)/2 & \text{if } (r_{0,0}-r_{1,1})\cdot(r_{0,1}+r_{1,0}) < 0 \end{cases} \tag{9}$$

where

$$\phi = \cos^{-1}\frac{|r_{0,0}-r_{1,1}|}{\sqrt{(r_{0,0}-r_{1,1})^2+(r_{0,1}+r_{1,0})^2}} \tag{10}$$

In practice, one may find that some modified rotation structures actually appeared in the butterfly implementation of a transform matrix. For instance, the structure shown in Fig. 2(b) is often encountered. Fortunately, our analysis shows that the "best" angle θ is exactly the same as determined above in Eqs. (9) and (10).

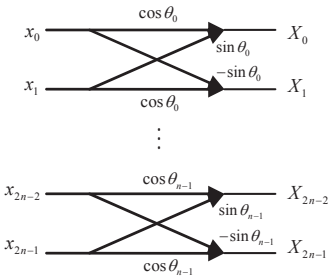


Fig. 3. One stage consisting of N butterflies in parallel

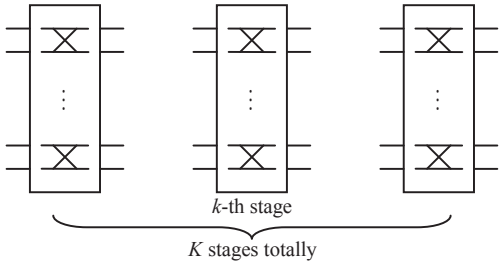


Fig. 4. Cascade of multiple stages to form a meaningful implementation for a practical transform matrix

B. Parallel of multiple butterflies

In practice, a number of rotations are often put into a parallel in order to construct one stage within any useful transform. Figure 3 shows such a stage consisting of N butterflies in parallel. Because of the parallel nature, it is easy to understand that all butterflies in this parallel structure can be handled independently. That is, given $\boldsymbol{r}=[r_{i,j}]_{(2N-1)\times(2N-1)}$ - the co-variance matrix of $[x_0,\ldots,x_{2N-1}]^T$, each "best" rotation angle θ_i, $i=0,\ldots,N-1$, can be determined independently by Eq. (9), with

$$\phi_i = \cos^{-1}\frac{|r_{2i-2,2i-2}-r_{2i-1,2i-1}|}{\sqrt{(r_{2i-2,2i-2}-r_{2i-1,2i-1})^2+(r_{2i-2,2i-1}+r_{2i-1,2i-2})^2}} \tag{11}$$

Notice that this is an analytical and closed-form solution. Thus, once the input node-variables at any stage are paired to form a parallel of multiple butterflies, we can follow the closed-form Eq. (11) to determine all rotation angles analytically to achieve the maximum coding gain.

C. The generic case - cascade of multiple stages

In reality, it is always the case that multiple stages are cascaded to construct a useful implementation for any practical transform matrix, see Fig. 4 for one example. In this case, one can find that it becomes extremely difficult to solve the corresponding problem analytically, even when $K=2$. Nevertheless, we can still get an analytical solution through the stage-by-stage strategy. That is, we can analytically determine all rotation angles involved in the first stage and compute the co-variance matrix of the output node-variable; based on this new co-variance matrix, we can determine all rotation angles involved in the second stage and compute another new co-variance matrix; and so on. However, this stage-by-stage strategy does not guarantee the optimality (global or even local) of the derived solution.

On the other hand, once the overall implementation butterfly

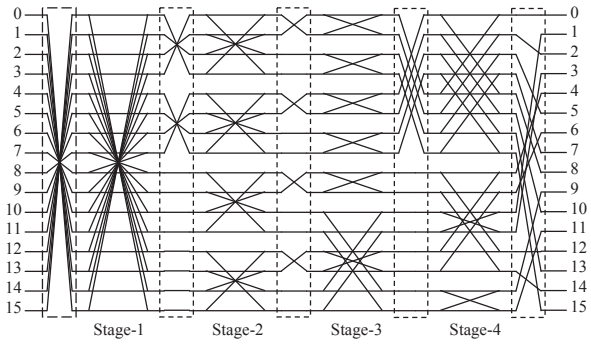


Fig. 5. Flow-graph of the equivalent16-point DCT

structure is fixed, one can run an exhaustive search for all involved rotation angles to get the maximum coding gain. Typically, each rotation angle θ_i ranges over [0° 90°]. Therefore, the search would soon become unaffordable when the total number of butterflies increases.

In the following section, we will focus on the design of 4×4 non-separable 2-D transform. A hybrid design strategy (a combination of the analytical solution and the full-search) will be developed for this very popular block size.

3. Design of 4×4 Non-Separable 2-D Transform

We first convert the 4×4 2-D separate DCT into the equivalent 16-point 1-D transform by concatenating four columns. This 1-D transform can be implemented using the flow-graph as shown in Fig. 5. There are totally 4 stages and each stage consists of 8 butterflies in parallel. Notice that the extra structure before each stage (in a dashed-line box) is a permutation – which has no impact on the coding gain. Table I lists all pairs and the corresponding rotation angles appeared in Fig. 5.

Next, we focus on how to design some new transforms based on the flow-graph shown in Fig. 5, i.e., to determine some rotation angles so that the resulting coding gain is maximized. Before we pursue this design, two remarks are necessary:

- This design is meaningful only when a strong directionality (other than the horizontal or vertical) is assumed in the 2-D source. Otherwise, the separate 2-D DCT will be very close to the optimal transform (which is the KLT) [5].
- Referring to Fig. 5, there are totally 4 stages, whereas the total number of butterflies is 32. Thus, it is obvious that a completely analytical solution is not possible, neither is the full-search on all involved 32 angles.

Here, we follow the directional model shown in Fig. 1 and consider only one diagonal direction with $\alpha = \pi/4$. [1] Since we cannot handle all four stages (with 32 butterflies) jointly, let's handle two stages while fixing the other two. Obviously, it would be very beneficial to fix Stages-1 and 2 as all rotation angles involved are 45° so that no real multiplications are needed.

Then, for the remaining two stages, we propose to perform a search on Stage-3 and make use of the analytical solution on Stage-4. The complexity of this hybrid fully depends on the search process. Fortunately, the search can be carried out hierarchically: we do a coarse search (e.g., every 15° over [0° 90°]) and then refine the search on a smaller scale.

Given α, η, and ρ, the co-variance matrix can be derived from the elliptical model in Eq. (3). By setting $\alpha = \pi/4$ and $\rho = 0.95$, we designed two new transforms for $\eta = 5$ and 7, respectively. The results are presented in Table II. Then, we compare them with the optimal (non-separable) KLT, the traditional separate 2-D DCT, and the separate 2-D KLT, with the corresponding coding gains presented in Table III.

Table I. Rotation angles appeared in Fig. 5

Stage 1	(0,15), 45°	(1,14), 45°	(2,13), 45°	(3,12), 45°
	(4,11), 45°	(5,10), 45°	(6,9), 45°	(7,8), 45°
Stage 2	(0,3), 45°	(1,2), 45°	(4,7), 45°	(5,6), 45°
	(8,11), 45°	(9,10), 45°	(12,15), 45°	(13,14), 45°
Stage 3	(0,1), 45°	(2,3), 67.5°	(4,5), 45°	(6,7), 67.5°
	(8,9), 45°	(10,14), 45°	(12,13), 45°	(11,15), 45°
Stage 4	(0,4), 45°	(1,5), 45°	(2,6), 67.5°	(3,7), 67.5°
	(8,12), 67.5°	(9,13), 67.5°	(10,11), 67.5°	(14,15), 67.5°

Table II. Rotation angles involved in the design transforms

$\eta = 5$				
Stage 3	(0,1),43°	(2,3), 64°	(4,5), 43°	(6,7), 60°
	(8,9), 35°	(10,14), 35°	(12,13), 34°	(11,15), 34°
Stage 4	(0,4), 47°	(1,5), 27°	(2,6), 26°	(3,7), 31°
	(8,12), 68°	(9,13), 52°	(10,11), 68°	(14,15), 52°
$\eta = 7$				
Stage 3	(0,1), 43°	(2,3), 65°	(4,5), 42°	(6,7), 56°
	(8,9), 33°	(10,14), 33°	(12,13), 31°	(11,15), 31°
Stage 4	(0,4), 47°	(1,5), 38°	(2,6), 27°	(3,7), 33°
	(8,12), 68°	(9,13), 48°	(10,11), 68°	(14,15), 48°

Table III. Coding gains of various transforms

	Non-separable 2-D KLT	New Transform	Separate 2-D KLT	Traditional 2-D DCT
$\eta = 5$	1.2056	1.0441	1.0375	1.0202
$\eta = 7$	1.1082	0.8939	0.8835	0.8585

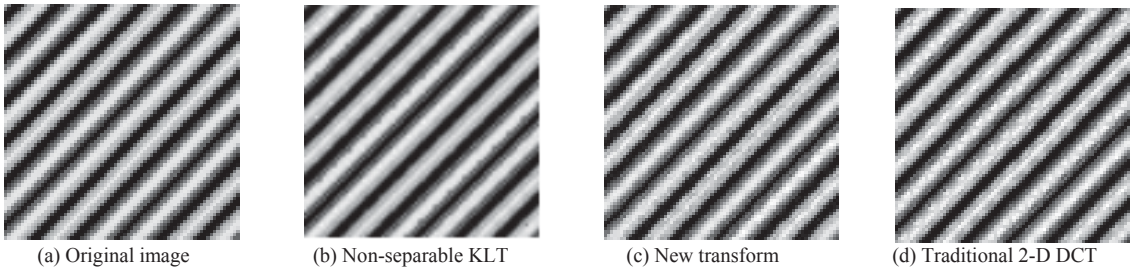
It can be seen from Table III that the separate 2-D KLT is only slightly better than the traditional 2-D DCT. This is because that a much stronger correlation within this 2-D source is along the diagonal direction so that it becomes very inefficient to design the optimal KLT along the horizontal or vertical direction. On the other hand, the non-separable 2-D KLT offers a huge gain over the traditional 2-D DCT (by about 20%), indicating a big room for us to design new transforms to improve the performance. Finally, although our new transforms are still far from the optimal KLT, they are found to be slightly better than the separate KLT and therefore the traditional 2-D DCT. Some experimental results will be presented below to show that the "little" gain in our new transforms (as demonstrated above) will become more significant in the practical cases.

Notice that there are several different flow-graphs equivalent to the one implemented in Fig. 5. We choose the structure of Fig. 5 just as one example to illustrate our design procedure. The same procedure is applicable to other (equivalent) flow-graph structures, but perhaps leading to different (non-separable) transforms.

4. Experimental Results

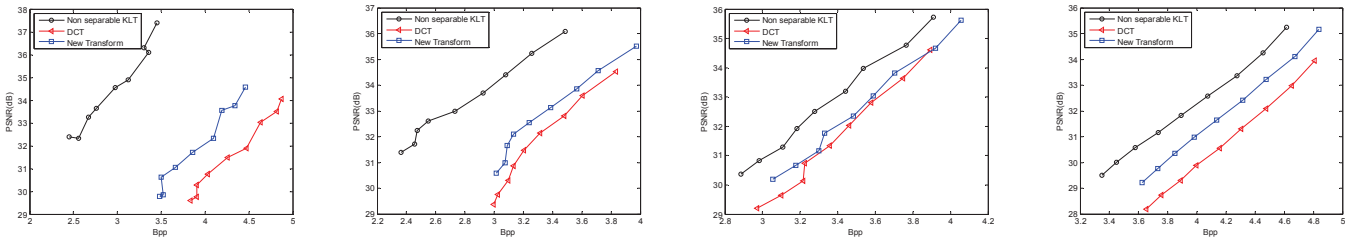
In this section, we present some simulation results of testing our new transforms. In the first test, we synthesize an ideal sinusoidal

[1] Another diagonal direction with $\alpha = -\pi/4$ (or $3\pi/4$) can be designed symmetrically, and these two are clearly the most important directions.



(a) Original image (b) Non-separable KLT (c) New transform (d) Traditional 2-D DCT

Fig. 6. Subjective comparisons of various transforms on the synthetic image (step-size = 50)



(a) (b) (c) (d)

Fig. 7. R-D performance for (a) the synthetic image, (b) the first frame of "Foreman", (c) the image Barbara, and (d) the image Straw.

image patch (of size 64×64) that is oriented exactly along the 45° (diagonal) direction, see Fig. 6(a). Then, we apply the optimal KLT, our new transform (with $\eta = 7$), and the traditional 2-D DCT to code this image patch. The R-D curves of these three transforms are shown in Fig. 7(a). It is no surprise that the optimal KLT offers a huge gain (over 10 dB) over the traditional DCT. It is also obvious that our new transform has filled up about one-fifth of this gap. Some visual results are shown in Fig. 6(b)-(d).

In the second test, we choose a video frame or image from the commonly-used set and follow H.264 to decide the intra-prediction mode for each 4×4 block. We keep all blocks of Mode 3 or 4 (two diagonal modes). We do not do the intra-prediction; alternatively, we apply our new transforms (with $\eta = 5$) on those blocks. We do not consider other blocks (that do not belong to Mode 3 or 4), i.e., the bit-count and the resulted distortion are computed for blocks of Mode 3 or 4. The R-D results for three test images are shown in Fig. 7(b)-(d), from which we can obtain very similar observations: (i) the optimal KLT offers a huge gain over the traditional DCT and (ii) this gap has been filled up partially through using our new transforms.

It is noted from Fig. 7 that the bit-rate is pretty high. This is because that we did not do the intra-prediction so that H.264 becomes inefficient. Our future work will consider the design of non-separable transforms for the predicted residual signals – some preliminary results have been reported for motion-compensated signals in [6].

5. Concluding Remarks

To our best knowledge, this is a very original attempt to design practical non-separable transforms for image and video coding. It becomes particularly meaningful for 2-D sources that have some dominating directions. Although the R-D performance achieved by our current design is still far below that of the optimal KLT, they do offer a remarkable gain over the traditional 2-D DCT. Their importance becomes higher by noticing that they can be implemented at the same complexity as that of the traditional 2-D DCT.

The design mechanism is based on a fixed butterfly structure: to determine all involved rotation angles optimally to achieve the biggest coding gain. In our future work, we will relax this fixed structure so that any two node-variables at each stage can be paired – leading to the problem of how to find the best pairing strategy. We anticipate that a further improved R-D performance would be achieved by accommodating such a pairing strategy.

Another future work is to apply this design framework to the intra-predicted signal in H.264 or the more recent HEVC so that the separate 2-D transforms that are currently adopted will be replaced by some non-separable transforms.

References

[1] N. Ahmed, T. Natarajan, and R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. 23, pp. 90–93, Jan. 1974.

[2] N. S. Jayant and P. Noll, *Digital Coding of Waveforms – Principles and Applications to Speech and Video*, Englewood Cliffs, NJ: Prentice-Hall, 1984.

[3] W. H. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Trans. Commun.*, vol. COM-25, pp. 1004–1009, Sept. 1977.

[4] P. Dita, "Factorization of unitary matrices," *J. Phys. A: Math Gen.*, vol. 36, 2003.

[5] S. Zhu, S. Au-Yang, and B. Zeng, "R-D performance upper bound of transform coding for 2-D directional sources," *IEEE Signal Proc. Lett.*, vol. 16, no. 10, pp. 861–864, Oct. 2009.

[6] S. Zhu, S. Au-Yang, and B. Zeng, "In search of 'better-than-DCT' unitary transforms for encoding of residual signals," *IEEE Signal Proc. Lett.*, vol. 17, no. 11, pp. 961-964, Nov. 2010.