# New Transforms Tightly Bounded by DCT and KLT

Haoming Chen and Bing Zeng, *Member, IEEE*

***Abstract*—It is well known that the discrete cosine transform (DCT) and Karhunen–Loève transform (KLT) are two good representatives in image and video coding: the first can be implemented very efficiently while the second offers the best R-D coding performance. In this work, we attempt to design some new transforms with two goals: i) approaching to the KLT's R-D performance and ii) maintaining the implementation cost no bigger than that of DCT. To this end, we follow a cascade structure of multiple butterflies to develop an iterative algorithm: two out of N nodes are selected at each stage to form a Givens rotation (which is equivalent to a butterfly); and the best rotation angle is then determined by maximizing the resulted coding gain. We give the closed-form solutions for the node-selection as well as the angle-determination, together with some design examples to demonstrate their superiority.**

***Index Terms*—Butterfly structure, discrete cosine transform, Givens rotation, Karhunen–Loève transform, R-D performance.**

## I. Introduction

THE discrete cosine transform (DCT) has been dominating the area of image and video coding for more than three decades, as evidenced by a number of coding standards, such as JPEG, MPEG-1/2/4, and H.264, that are widely used in today's applications. Only very recently, a few research works have re-visited the much older-aged Karhunen–Loève transform (KLT) and reported some interesting results [1]–[3]. Generally speaking, both KLT and DCT are very good representatives: KLT produces the best R-D performance; whereas DCT is a robust approximation to KLT under some conditions [4]. However, the DCT's performance would drop drastically when these conditions are not met, especially in the 2-D case. To justify this statement, Fig. 1 shows the coding performances of two KLTs of $4 \times 4$ (one based on the statistics archived from the image portion shown at the top-left corner and another based on an analytical model as discussed later on) and the 2-D separable DCT (all using H.264 without intraprediction). Notice that the gap between these two transforms is quite big ($2^{+}$ dB), meaning that we potentially have a big room to improve.

Another advantage of DCT is that it can be implemented very efficiently; whereas no fast algorithms have so ever been developed for KLT. Very naturally, one would ask the following question: can we design new transforms with R-D performance asymptotically approaching to that of KLT while maintaining the implementation complexity similar to or even below that of DCT?

Manuscript received February 15, 2012, revised March 29, 2012; accepted April 02, 2012. Date of publication April 18, 2012; date of current version April 23, 2012. This work was supported in part by an RGC research grant of the HKSAR government. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Lisimachos Paul Kondi.

The authors are with the ECE Department, Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong SAR, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LSP.2012.2195172

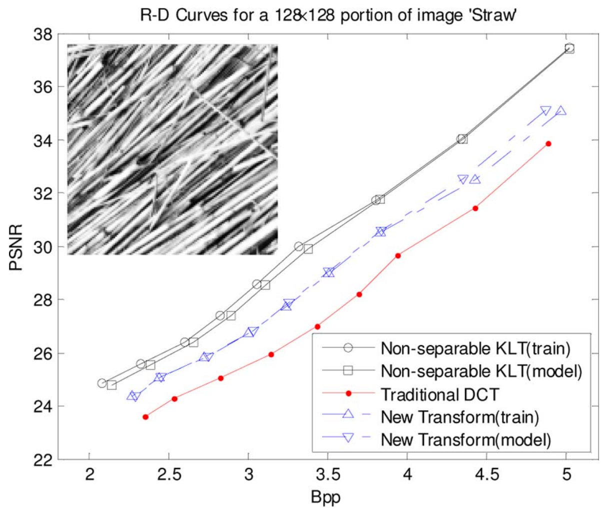

Fig. 1. Gap between KLT and DCT can be huge. Two other curves are the results obtained by our new transforms.

In this work, we present a very positive answer so that such new transforms indeed can be designed for some important scenarios.

## II. Some Basics of KLT and DCT

Given the covariance matrix $\boldsymbol{\Sigma} = [\sigma_{i,j}]_{N\times N}$ of a random vector $\boldsymbol{x} = [x_0, \ldots, x_{N-1}]^T$, it is known that KLT can be derived from the eigenvectors of $\boldsymbol{\Sigma}$. In many cases, $\boldsymbol{\Sigma}$ is modeled by a Toeplitz-type matrix whose elements are defined as

$$\sigma_{i,j} = \rho^{|i-j|}, i, j = 0, 1, \ldots, N-1, |\rho| < 1. \tag{1}$$

When $\rho \to 1$ (indicating a very strong interpixel correlation), it has been proven that DCT can approximate KLT very closely [4].

In practice, one common scenario is that $\boldsymbol{x}$ contains a sharp edge. In this case, two segments (before and after the edge) are much less correlated or even uncorrelated. Then, we need to modify $\boldsymbol{\Sigma}$ accordingly to make it to be block-diagonal.

In the 2-D case with $\boldsymbol{x}_{2D} = \{x_{p,q}\}_{N\times N}$, the correlation between two pixels $A$ and $B$ at locations $(p_A, q_A)$ and $(p_B, q_B)$ can be modeled as [5]

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

After concatenating all columns of $\boldsymbol{x}_{2D}$ into a "tall" vector of height $N^2$, a big covariance matrix $\boldsymbol{\Sigma}$ (of size $N^2 \times N^2$) is obtained and the corresponding KLT can be derived. This KLT is often non-separable and sets the performance upper-bound for all block-based transform coding schemes [2].

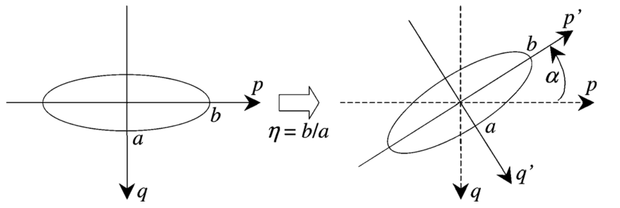

Fig. 2. Elliptical function and its rotated version used to model 2-D directional sources.

More generally, the 2-D block $\boldsymbol{x}_{2D}$ may have a dominating direction—which can be horizontal, vertical, diagonal, or any other as long as it can be defined in an $N \times N$ block. To model this scenario, we need a new covariance matrix with

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

where $\eta = b/a > 1$ represents the ratio of long and short radius of a rotated elliptical function (see Fig. 2) and

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

Notice that the model defined by (3) becomes identical to (2) when $\eta = 1$. Now, let's use $\boldsymbol{C}$ to denote the matrix of a transform applied on $\boldsymbol{x}$. The transformed coefficients are $\boldsymbol{X} = [X_0,\ldots,X_{N-1}]^T$ and the energy of each individual coefficient is denoted as $\sigma^2_{X_n}$. The coding gain achieved by applying $\boldsymbol{C}$ is

$$G_C = -\frac{1}{N}\sum_{n=0}^{N-1}\log_2\left(\sigma^2_{X_n}\right). \tag{5}$$

Three examples are presented below to show the difference between DCT and KLT.

### A. 1-D Signal Containing a Sharp Edge

Suppose that a 16-point signal contains a sharp edge in middle so that the first eight samples and the last 8 samples are uncorrelated. Let's further assume that each of two segments is modeled by the same $8 \times 8$ Toeplitz matrix with $\rho = 0.95$. Then, we derive the corresponding KLT. By applying it and the standard 16-point DCT, we obtain $G_{\text{DCT}} \approx 2.3196$ and $G_{\text{KLT}} \approx 2.9386$, respectively.

### B. Directional 2-D Source (No Intra-Prediction)

Suppose that a $4 \times 4$ image block $\{x_{i,j}\}_{4\times 4}, i,j = 0,\ldots,3$, has the diagonal down-left(DDL) direction, i.e., $\alpha = 45^\circ$. Assuming $\eta = 5$ and $\rho = 0.95$, we use (3) to calculate the $16 \times 16$ covariance matrix and derive the corresponding KLT. Applying it and the 2-D separable DCT, we obtain $G_{\text{DCT}} \approx 2.0404$ and $G_{\text{KLT}} \approx 2.4112$, respectively.

### C. Residual 1-D Source (After Intra-Prediction)

Consider the $4 \times 4$ image block again, but now with $\alpha = 90^\circ$, $\eta = 5$, and $\rho = 0.95$. Clearly, the dominating direction is vertical so that we apply Mode-0 (in H.264) intraprediction. After the intraprediction, we calculate the covariance matrix for each column (all columns have the same result) and then derive the corresponding KLT. The coding gain incurred by DCT is $G_{\text{DCT}} \approx 3.1169$; whereas applying KLT results in $G_{\text{KLT}} \approx 3.3232$.

## III. The Problem and its Solution

In all three scenarios presented above, we observed that KLT performs much better than DCT. However, we know that DCT can be implemented very efficiently, which is not true for KLT. Now, the question we would like to solve is: how can we design new transforms with performance asymptotically approaching to that of KLT, while maintaining the implementation complexity similar to or even below that of DCT.

### A. Formulation of the Problem

In principle, any unitary transform matrix (of size $N \times N$) can be factorized into a product of Givens rotations [6], i.e.,

$$\boldsymbol{C} = \prod_{k=1}^{K}\Omega(i_k,j_k,\theta_k), i_k \neq j_k \in \{0,\ldots,N-1\} \tag{6}$$

where the node-variables at each stage are numbered from 0 to $N-1$, and $\Omega(i_k,j_k,\theta_k)$ is an $N \times N$ matrix representing the Givens rotation (by $\theta_k$) between two node-variables $(i_k,j_k)$. However, the difficulty in applying this result onto an obtained KLT matrix is that there often exist many factorizations with different $K$ (the total number of Givens rotations). In most cases, $K$ is much bigger than $L$—the total number of butterflies (equivalent to Givens rotations) used in the fast implementation of the $N \times N$ DCT matrix. To maintain the DCT's complexity, one naive idea is to choose $L$ Givens rotations from $K$ candidates, assuming that the factorization has been accomplished. However, such an idea will not be working in practice because: (1) it implies a huge combinational number (e.g., $L$ is 32 in the $4\times\ 4$ 2-D DCT) and (2) $L$ selected rotations (with their rotation angles fixed after the factorization) together cannot guarantee a transform that is better than DCT.

Nevertheless, the above discussion does imply a feasible solution: instead of selecting $L$ out of $K$ Givens rotations, we can design $L_0(\leq L)$ Givens rotations (that are different from those obtained in the factorization) in such a way that the resulted coding gain is optimized:

$$\begin{aligned} &\underset{i_l,j_l,\theta_l}{\text{maximize}} && G_{\boldsymbol{C}} \\ &\text{subject to} && \boldsymbol{C} = \prod_{l=1}^{L_0}\Omega(i_l,j_l,\theta_l) \end{aligned} \tag{7}$$

This is to say that we need to determine, at each stage $l$, a "best" pair of two nodes $(i_l,j_l)$ and the "best" angle $\theta_l$ so as to maximize the coding gain.

A simplified version of this problem has been studied in our previous work [7] where we follow the DCT's butterfly graph exactly so that the pairing of nodes is avoided completely. The optimization formulated in (7) has relaxed all variables, including the indices. Clearly, the current problem becomes harder to handle. In the following, we first present some results for a standing-alone Givens rotation (already obtained in [7]) and then construct an iterative algorithm for designing transforms with performance asymptotically approaching the best coding gain (that is achievable only by KLT).

### B. Optimal Angle for a Standing-Alone Givens Rotation

Two nodes before and after a Givens rotation are related as

$$\begin{bmatrix} X_0 \\ X_1 \end{bmatrix} = \Omega \cdot \begin{bmatrix} x_0 \\ x_1 \end{bmatrix}, \ \Omega = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \ \theta \in \left[0, \frac{\pi}{2}\right]. \tag{8}$$

Let's denote the covariance matrix of $[x_0 x_1]^T$ as $\boldsymbol{r} = [r(i,j)]$. Then, the covariance matrix of $[X_0 X_1]^T$ is $\boldsymbol{R} = \Omega \boldsymbol{r} \Omega T$. According to [7], the coding gain after applying the transform is maximized by taking the rotation angle $\theta$ as follows:

$$\theta = \begin{cases} \frac{\phi}{2} & \text{if } (r(0,0) - r(1,1)) \cdot (r(0,1) + r(1,0)) \geq 0 \\ \frac{(\pi-\phi)}{2} & \text{otherwise} \end{cases} \tag{9}$$

where

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

After applying the Givens rotation with angle given in (9), it can be verified that (i) $R(0,0)+R(1,1) = r(0,0)+r(1,1)$—the variance-preserving property of a unitary transform and (ii) $R(0,1) = R(1,0) = 0$—meaning that two nodes are de-correlated completely after the transform. The second result is exactly the same as what can be achieved by KLT. Subsequently, we have reached the optimal solution for a standing-alone butterfly structure.

In practice, one may find that some modified rotation can appear in the butterfly implementation of a transform matrix. For instance, the minus sign may appear on other element of $\Omega$ in (8). Fortunately, our analysis tells that the "best" angle $\theta$ will be exactly the same as determined above.

### C. Solution to a Cascade of Multiple Givens Rotations

The results for a standing-alone Givens rotation derived above suggests an iterative approach to the problem formulated in (7) when the number of nodes $N$ is greater than 2: in each stage $l$, we choose a pair of two nodes in such a way that its best angle $\theta_l$ (analytically determined from (9) and (10) for any selection) leads to the *maximum* coding gain after performing the Givens rotation (with angle $\theta_l$) on two selected nodes. Following our earlier notations, we also use $\boldsymbol{r}$ and $\boldsymbol{R}$ to represent the covariance matrices of the node-variables before and after the $l$th Givens rotation, respectively. Notice that there are $N$ nodes so that both $\boldsymbol{r}$ and $\boldsymbol{R}$ have size $N \times N$. The coding gain achieved after the $l$th Givens rotation is

$$G_C = -\frac{1}{N} \sum_{n=0}^{N-1} \log_2 R(n,n). \tag{11}$$

As the $l$th Givens rotation takes place on two nodes $(i_l, j_l)$, only two diagonal elements $R(n,n)$ would change after the transform, i.e., $n = i_l$ and $n = j_l$. Upon this change, the coding gain becomes

$$\begin{aligned} G_C &= -\frac{1}{N} \log_2 \left( R(i_l,i_l) \times R(j_l,j_l) \times \prod_{\substack{i=0 \\ n\neq j_l \text{ and } n\neq k_l}}^{N-1} R(n,n) \right) \\ &= -\frac{1}{N} \log_2 \left( \left( 1 - \frac{r(i_l,j_l) \times r(j_l,i_l)}{r(i_l,i_l) \times r(j_l,j_l)} \right) \times \prod_{n=0}^{N-1} r(n,n) \right). \end{aligned} \tag{12}$$

Based on (12), we now present our iterative algorithm as follows:

*1) The Algorithm:* Given the initial covariance matrix $\boldsymbol{\Sigma}$ of the input random vector $\boldsymbol{x} = [x_0, \ldots, x_{N-1}]^T$, set $l = 0$, $\boldsymbol{r} = \boldsymbol{\Sigma}$, and initialize the transform matrix with $\boldsymbol{C} = \boldsymbol{I}$ ($N \times N$ identity matrix).

1) For $l = 1 : L_0$, search over $\Pi = \{0, 1, \ldots, N-1\}$ to identify two integers $(i_l, j_l)$ that lead to the largest ratio $\gamma$, where

$$\gamma = \frac{r(i_l,j_l) \times r(j_l,i_l)}{r(i_l,i_l) \times r(j_l,j_l)}.$$

2) Calculate the best angle $\theta_l$ for the selected pair using (9) and (10). Stop if the best angle derived is equal to 0; otherwise, update the transform matrix $\boldsymbol{C}$ by $\Omega(i_l, j_l, \theta_l)$ (with all parameter determined): $\boldsymbol{C} \leftarrow \Omega(i_l, j_l, \theta_l) \cdot \boldsymbol{C}$.
3) Calculate the covariance matrix of the output nodes as follow:

$$\boldsymbol{R} = \Omega(i_l, j_l, \theta_l) \cdot \boldsymbol{r} \cdot \Omega^T(i_l, j_l, \theta_l)$$

and use it to replace $\boldsymbol{r}$, i.e., $\boldsymbol{r} \leftarrow \boldsymbol{R}$. Go to the next iteration.

*2) Convergence:* Denoting the coding gain after the $l$th iteration as $G^{(l)}$, we have

$$G^{(l)} \geq G^{(l-1)} \text{ and}$$
$$\lim_{l\to\infty} G^{(l)} = G_{\text{KLT}} = -\frac{1}{N} \sum_{i=0}^{N-1} \log_2 \lambda_i$$

where $G_{\text{KLT}}$ is the coding gain achieved by KLT and $\lambda_i$'s are the eigenvalues of $\boldsymbol{\Sigma}$.

Furthermore, our conjecture is that the convergence discussed above will be reached in a finite number of iterations. We are currently working on the proof and will report it in our future work.

## IV. Design Examples

We first focus on the $4 \times 4$ block-size (commonly used in H.264). Here, let us assume that the model (3) with $\alpha = 45°$, $\eta = 5$, and $\rho = 0.95$ is used, leading to Mode-3 or diagonal-down-left(DDL) mode. We can code such a $4 \times 4$ block directly (without any intraprediction) or follow the H.264 standard to do the Mode-3 intraprediction. In either way, a $16 \times 16$ covariance matrix can be obtained so that we can drive the corresponding KLT. Since the separable $4 \times 4$ 2-D DCT needs 32 butterflies totally (each 4-point DCT needs 4 butterflies), we first set $L_0 = 32$ when applying our iterative algorithm to design

TABLE I
COMPARATIVE RESULTS OF DIFFERENT TRANSFORMS UNDER TWO SCENARIOS

| | 4×4 block without intra-prediction | | | 4x4 block with DDL Intra-prediction | | |
|---|---|---|---|---|---|---|
| | KLT | DCT | Ours | KLT | DCT | Ours |
| Coding Gain | 2.4112 | 2.0404 | 2.3852 | 2.8956 | 2.5173 | 2.8748 |

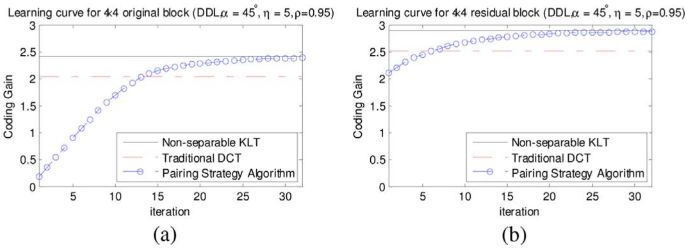

Fig. 3. Asymptotic performances: (a) without intraprediction and (b) with the DDL intraprediction.

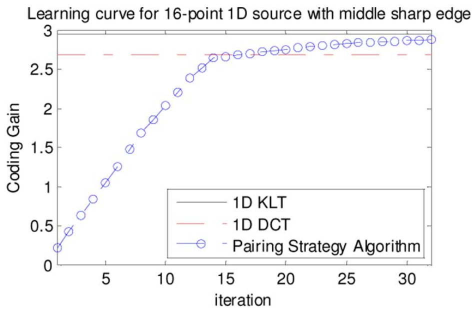

Fig. 4. Asymptotic performances: 16-point 1-D source.

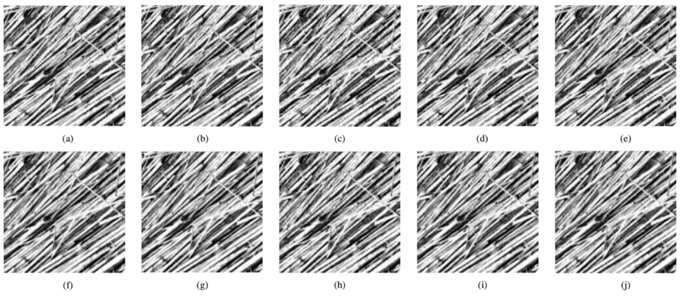

Fig. 5. Visual results coded with different transforms at the same bit-rate. All in the first row are coded at 2.5 bits/pixel and the 2nd row at 3.5 bits/pixel. From left to right, the transforms used are: training-based KLT, model-based KLT, 2-D separable DCT, training-based new transform, and model-based new transform (a) $\mathrm{PSNR} = 26.11$ dB; (b) $\mathrm{PSNR} = 25.92$ dB; (c) $\mathrm{PSNR} = 24.18$ dB; (d) $\mathrm{PSNR} = 25.18$ dB; (e) $\mathrm{PSNR} = 25.21$ dB; (f) $\mathrm{PSNR} = 30.18$ dB; (g) $\mathrm{PSNR} = 30.52$ dB; (h) $\mathrm{PSNR} = 27.36$ dB; (i) $\mathrm{PSNR} = 28.97$ dB; (j) $\mathrm{PSNR} = 29.05$ dB.

new transforms (so that they have the same implementation cost as the separable $4 \times 4$ 2-D DCT). Some comparative results (in terms of the coding gain) among the traditional DCT, the optimal KLT, and our new transforms are presented in Table I.

It is clear that our new transforms do offer a better performance than the separable 2-D DCT in both scenarios (without or with intraprediction). Next, let's choose $L_0 < 32$ so as to study the asymptotic performances over the whole iterative procedure as shown in Fig. 3: the performance of our new transform in either case has actually surpassed that of DCT far before the 32nd iteration. For instance, only 14 or 6 iterations are needed to deliver a performance better than that of DCT, which means a big saving in the implementation cost. Meanwhile, we would like to say that we also tried other sizes, such as $8 \times 8$, $16 \times 16$, and $32 \times 32$, and even bigger improvements (over DCT) have been obtained. We will report those results in our future works.

We then study the 1-D case. Here, we assume that a 16-point signal contains a sharp edge in middle. We use an $8 \times 8$ Toeplitz matrix to model the covariance matrix of both segments (before and after the edge) and assume that two segments are uncorrelated. By choosing $\rho = 0.95$, we run our iterative algorithm for designing a new transform. The learning curve is shown in Fig. 4. Again, we found that the performance of our new transform in this case has surpassed that of DCT after the 15th iteration; whereas the 16-point DCT needs more than 32 butterflies.

On the other hand, we compute the covariance matrix from the image portion shown in Fig. 1 or from the correlation model given in (3) (with $\rho = 0.99$, $\alpha = 45^\circ$, and $\eta = 6$). Two new transforms are designed using our iterative algorithm $(L_0 = 32)$. These two transforms are then tested under the same environment (i.e., using H.264 without intraprediction) and their R-D performances are also shown in Fig. 1 (two dashed curves). It can be seen that these two R-D curves are much closer to the KLTs' performance curves: the improvement over DCT is more than 1 dB. Finally, some visual results are given in Fig. 5 (under the same bit-rate), from which one can see clearly that a much improved visual quality has been obtained by using our new transforms.

## V. CONCLUDING REMARKS

To our best knowledge, this is the first attempt to designing transforms that are asymptotically bounded by KLT (in terms of the R-D coding performance) and DCT (in terms of the implementation cost). To this goal, we have made use of the cascade structure of multiple butterflies and developed an iterative algorithm. We showed that the design at each iteration can be simplified as finding a best pair of two nodes (out of $N$ candidates) and then determining the corresponding rotation angle to get the maximum coding. We gave the closed-form solution to both node-selection and angle-determination.

A few design examples have been presented to demonstrate the effectiveness of these new transforms. Our future work will be focusing on H.264 or HEVC intrapredicted signals. In this scenario, each block belongs to a directional mode so that a well-designed directional transform would be able to yield a significant improvement, while being implemented at a similar cost as that for the traditional DCT.

## REFERENCES

[1] Y. Ye and M. Karczewicz, "Improved H.264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning," in *IEEE ICIP-2008*, San Diego, CA, Oct. 2008.
[2] S.-Y. Zhu, S.-K. A. Yeung, and B. Zeng, "R-D performance upper bound of transform coding for 2-D directional sources," *IEEE Signal Process. Lett.*, vol. 16, no. 10, pp. 861–864, Oct. 2009.
[3] C. Yeo, Y. H. Tan, and Z. Li, "Low-complexity mode-dependent KLT for block-based intra coding," in *IEEE ICIP-2011*, Brussels, Belgium, Sep. 2011.
[4] A. K. Jain, "A sinusoidal family of unitary transforms," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 1, no. 4, pp. 356–365, Oct. 1979.
[5] T. Natarajan and N. Ahmed, "Performance evaluation for transform coding using a nonseparable covariance model," *IEEE Trans. Commun.*, vol. 26, no. 2, pp. 310–312, 1978.
[6] P. Diţă, "Factorization of unitary matrices," *J. Phys. A: Math. Gen.*, vol. 36, pp. 2781–2789, 2003.
[7] H.-M. Chen, S.-Y. Zhu, and B. Zeng, "Design of non-separable transforms for directional 2-D sources," in *IEEE ICIP-2011*, Brussels, Belgium, Sep. 2011.