LAYERED-GIVENS TRANSFORMS: TUNABLE COMPLEXITY, HIGH-PERFORMANCE APPROXIMATION OF OPTIMAL NON-SEPARABLE TRANSFORMS

Bohan Li[†], *Onur G. Guleryuz*[*], *Jana Ehmann*[‡] *and Arash Vosoughi*[+]

[†]U. C. Santa Barbara, Santa Barbara, CA, USA, [*]LG Electronics, San Jose, CA, USA,
[‡]Google Inc., Mountain View, CA, USA, [+]Sony Electronics, San Jose, CA, USA

ABSTRACT

We introduce layered-Givens transforms (LGTs) which are arbitrary-dimensional, tunable-complexity, orthonormal transforms of data. LGTs are formed by layers of data permutations and Givens rotations with the number of layers controlling the overall transform's computational complexity. We propose a novel method for the design of layered-Givens transforms that approximate desired complex transforms (such as KLTs, SOTs, etc.) so that most of the performance of the approximated transform is retained at significantly reduced complexity. Key to the LGT performance is the choice of the permutations that arrange the data prior to the Givens rotations. Despite the very highly combinatorial nature of the LGT design problem, we provide an algorithm that finds the optimal parameters (permutation and Givens rotations) for each layer. Our results show that designed LGTs closely approximate desired non-separable transforms at significantly reduced complexity.

Index Terms— layered-Givens transforms, row-column transforms, video compression, Blossom algorithm, Hungarian method.

1. INTRODUCTION

Linear transforms are the most important tools in the processing and compression of images and video. Researchers have developed a vast array of transforms ranging from low complexity separable designs such as DCTs, DSTs, wavelets, [12, 9, 16, 8] etc., to computationally complex non-separable designs like KLTs, curvelets, contourlets, and SOTs [16, 15, 3, 14]. For an $N \times N$ signal block, a separable transform requires $2N^3$ multiply-adds ($2N^2 \log N$ if fast transforms are used) whereas a general non-separable transform has N^4 computational complexity. While non-separable transforms provide significantly improved performance in many applications [18, 14], low-complexity-minded design work has mostly concentrated on separable approximations to elaborate designs [17, 9, 13].

In recent work Egilmez et al. introduced row-column transforms (RCTs) [5], which are non-separable transforms that have the same computational complexity as separable transforms. RCTs are formed by applying a set of $2N$, $1D$ transforms to the rows and columns of an $N \times N$ signal block followed by a permutation of the resulting coefficients. The authors of [5] propose a design algorithm that not only yields optimal RCTs but can also produce improved separable designs in approximating a given complex transform. It is shown that RCTs significantly outperform even the improved separable designs while retaining separable complexity.

In this paper we propose a new, unconventional, tunable-complexity transform structure that is termed the *layered-Givens transform* (LGT, Figure 1). LGTs are formed by layers of data permutations and Givens rotations with the number of layers controlling the overall transform's computational complexity. An LGT layer is

Authors' work on LGT was performed while they were with LGE.

$$\begin{tikzcd} x \arrow[r] & \boxed{G_M{}^T} \arrow[r] & \boxed{G_{M-1}{}^T} \arrow[r] & \cdots \arrow[r] & \boxed{G_1{}^T} \arrow[r] & \boxed{P_0{}^T} \arrow[r] & c \end{tikzcd}$$

Fig. 1. The forward layered-Givens transform. The data vector $\boldsymbol{x}$ is transformed using M layers of orthonormal transformations followed by a permutation. Layers M down to 1 involve permutations and Givens rotations. Reconstruction is via $\boldsymbol{x} = \mathbf{G}_M \dots \mathbf{G}_1 \mathbf{P}_0 \boldsymbol{c}$.

illustrated in Figure 2. The unconventional nature of the LGT is due to the data shuffling permutations which re-arrange the data prior to the application of Givens rotations (butterflies). In essence, an LGT couples a very large number of butterflies with $M + 1$ permutations in a way that forms dramatically different data-flows compared to other transformations (RCTs have a fixed data-flow except for a final $\mathbf{P}_0$-like permutation). Interestingly, the algorithm we propose in this paper can optimize the permutations within each layer jointly with the Givens rotation parameters of that layer. Our results show that LGTs obtain high performance approximation even below separable transform complexity levels.

The rest of the paper is organized as follows. In Section 2 we carefully set up and parameterize the LGT design problem so that powerful graph-theory-based techniques can be utilized in our algorithm derived in Section 3. Experimental results are discussed in Section 4 followed by Section 5 of concluding remarks.

2. PROBLEM FORMULATION

Let $\boldsymbol{x}$ $(K \times 1)$ denote the signal vector to be transformed. In this paper we are interested in the transformation of the $2D$, $N \times N$ block $\mathbf{X}$ lexicographically ordered into $\boldsymbol{x}$, i.e., $K = N^2$. Note however that $\boldsymbol{x}$ can in general contain arbitrary dimensional data.

Layered Givens Transforms: Assume K is even and let $\mathcal{M} = \{1, ..., M\}$. The M-layer LGT of $\boldsymbol{x}$ is obtained as

$$\boldsymbol{c} = \mathbf{P}_0^T \mathbf{G}_1^T \dots \mathbf{G}_M^T \boldsymbol{x}, \tag{1}$$

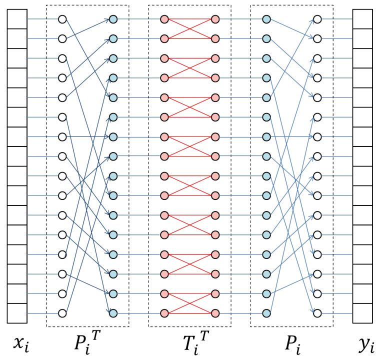


Fig. 2. The i^{th} forward LGT layer. The input vector $\boldsymbol{x}_i$ is permuted using the permutation $\mathbf{P}_i^T$, pairwise rotations applied, and the result inverse-permuted with $\mathbf{P}_i$ to obtain $\boldsymbol{y}_i = \mathbf{G}_i^T \boldsymbol{x}_i = \mathbf{P}_i \mathbf{T}_i^T \mathbf{P}_i^T \boldsymbol{x}_i$.

where the $K \times K$ orthonormal transformation

$$\mathbf{G}_i = \mathbf{P}_i \mathbf{T}_i \mathbf{P}_i^T, \;\; i \in \mathcal{M}, \tag{2}$$

is formed using the $K \times K$ permutation matrix $\mathbf{P}_i$ and the block-diagonal Givens rotation matrix $\mathbf{T}_i$. We have

$$\mathbf{T}_i = \begin{bmatrix} \mathbf{T}_{i,1} & 0 & \dots & 0 \\ 0 & \mathbf{T}_{i,2} & & \\ \vdots & & \ddots & \\ 0 & & & \mathbf{T}_{i,K/2} \end{bmatrix}, \tag{3}$$

with $\mathbf{T}_{i,j} = \begin{bmatrix} \cos(\theta_{i,j}) & \sin(\theta_{i,j}) \\ -\sin(\theta_{i,j}) & \cos(\theta_{i,j}) \end{bmatrix}$ and $0 \leq \theta_{i,j} < 2\pi$.

Let $\mathbf{G}$ be the matrix with the LGT basis in its columns, i.e.,

$$\begin{aligned} \mathbf{G} &= \mathbf{G}_M \dots \mathbf{G}_1 \mathbf{P}_0, & (4)\\ &= \mathbf{P}_M \mathbf{T}_M \mathbf{P}_M^T \dots \mathbf{P}_1 \mathbf{T}_1 \mathbf{P}_1^T \mathbf{P}_0. & (5) \end{aligned}$$

The transform coefficients of $\mathbf{X}$ are thus obtained as $\boldsymbol{c} = \mathbf{G}^T \boldsymbol{x}$. This particular form for $\mathbf{G}$ will be convenient in our optimization setup below. It is nevertheless important to note the following result.

Proposition 2.1 *Given permutation matrices $\mathbf{Q}_0, \dots, \mathbf{Q}_M$ the transformation $\mathbf{Q}_M \mathbf{T}_M \dots \mathbf{Q}_1 \mathbf{T}_1 \mathbf{Q}_0$ can be written as an LGT using $\mathbf{P}_M = \mathbf{Q}_M$, $\mathbf{P}_k = \mathbf{P}_{k+1} \mathbf{Q}_k$, $k = 0, \dots, M-1$ and the same Givens rotation matrices.*

LGT Design Problem: Given a desired $K \times K$ orthonormal transform matrix $\mathbf{H}$, we pose its LGT approximation via

$$\min_{\mathbf{G}} ||\mathbf{H} - \mathbf{G}||_F, \tag{6}$$

where $||.||_F$ denotes the Frobenius norm. Note that

$$\begin{aligned} ||\mathbf{H} - \mathbf{G}||_F^2 &= Tr\{\mathbf{H}^T\mathbf{H}\} - 2Tr\{\mathbf{H}^T\mathbf{G}\} + Tr\{\mathbf{G}^T\mathbf{G}\} \\ &= 2K - 2Tr\{\mathbf{H}^T\mathbf{G}\} & (7) \end{aligned}$$

where $Tr\{.\}$ denotes trace. Using (4) the optimization becomes

$$\max_{\mathbf{P}_0, \mathbf{G}_1, \dots, \mathbf{G}_M} Tr\{\mathbf{H}^T \mathbf{G}_M \dots \mathbf{G}_1 \mathbf{P}_0\}. \tag{8}$$

Observe that the optimization in (8) is nonlinear and, because of the permutations, highly combinatorial.

3. PROPOSED METHOD

Our technique proceeds with the optimization by separately optimizing each term ($\mathbf{G}_i$, $i \in \mathcal{M}$, and $\mathbf{P}_0$) in an *alternating maximization approach*.

Optimization of $\mathbf{P}_0$: The trace in (8) can be maximized in terms of the permutation $\mathbf{P}_0$ by solving

$$\max_{\mathbf{P}_0} Tr\{\mathbf{H}^T \mathbf{G}_M \dots \mathbf{G}_1 \mathbf{P}_0\}. \tag{9}$$

Remark: The optimization in (9) is an *assignment problem* and can be solved using the *Hungarian method* [11] in polynomial time.

Optimization of $\mathbf{G}_i$: Using the trace in (8) the optimization for $\mathbf{G}_i$ can be written as

$$\max_{\mathbf{G}_i} Tr\{\mathbf{L}_i \mathbf{G}_i \mathbf{E}_i\}, \tag{10}$$

where $\mathbf{L}_i = \mathbf{H}^T \mathbf{G}_M \dots \mathbf{G}_{i+1}$ and $\mathbf{E}_i = \mathbf{G}_{i-1} \dots \mathbf{G}_1 \mathbf{P}_0$. Note that $\mathbf{G}_i$ can be seen as selecting pairs of rows from $\mathbf{E}_i$ and rotating these pairs to match co-located pairs of rows in $\mathbf{L}_i^T$ (columns in $\mathbf{L}_i$). Of course, the optimal $\mathbf{G}_i$ must choose the best pairs and the best rotation angles jointly to maximize (10).

(I) Optimizing a single rotation: With the help of (2), (3), and Figure 2, let us concentrate on the j^{th} butterfly in $\mathbf{T}_i$. Suppose the permutation $\mathbf{P}_i$ selects rows p and q as the j^{th} pair to be rotated by $\mathbf{T}_{i,j}$. Let the row vectors $\boldsymbol{e}_{i,p}$, $\boldsymbol{e}_{i,q}$ and $\boldsymbol{l}_{i,p}$, $\boldsymbol{l}_{i,q}$ determine the rows p and q from $\mathbf{E}_i$ and $\mathbf{L}_i^T$ respectively. Using $Tr\{\mathbf{AB}\} = Tr\{\mathbf{BA}\}$ the relevant portion of (10) becomes

$$Tr\left\{ \begin{bmatrix} \cos(\theta_{i,j}) & \sin(\theta_{i,j}) \\ -\sin(\theta_{i,j}) & \cos(\theta_{i,j}) \end{bmatrix} \begin{bmatrix} \boldsymbol{e}_{i,p} \\ \boldsymbol{e}_{i,q} \end{bmatrix} \begin{bmatrix} \boldsymbol{l}_{i,p}^T & \boldsymbol{l}_{i,q}^T \end{bmatrix} \right\}. \tag{11}$$

Let $\alpha_{i,p,q} = \boldsymbol{e}_{i,p}\boldsymbol{l}_{i,p}^T + \boldsymbol{e}_{i,q}\boldsymbol{l}_{i,q}^T$, $\beta_{i,p,q} = \boldsymbol{e}_{i,q}\boldsymbol{l}_{i,p}^T - \boldsymbol{e}_{i,p}\boldsymbol{l}_{i,q}^T$, and $\gamma_{i,p,q} = (\alpha_{i,p,q}^2 + \beta_{i,p,q}^2)^{1/2}$. Equation (11) can be written as

$$\cos(\theta_{i,j})\alpha_{i,p,q} + \sin(\theta_{i,j})\beta_{i,p,q}, \tag{12}$$

with the maximizing $\theta_{i,j}$ satisfying

$$\cos(\theta_{i,j}) = \alpha_{i,p,q}/\gamma_{i,p,q}, \; \sin(\theta_{i,j}) = \beta_{i,p,q}/\gamma_{i,p,q}. \tag{13}$$

The resulting maximal trace is hence

$$\gamma_{i,p,q} = (\alpha_{i,p,q}^2 + \beta_{i,p,q}^2)^{1/2}. \tag{14}$$

(II) Optimal pairings for rotations: It is now clear that the optimization of $\mathbf{G}_i$ in (10) can be accomplished by maximizing the sum of $\gamma_{i,p,q}$ over all possible nonintersecting pairs. This can be posed as a problem of maximum matching in graph theory.

Proposition 3.1 *Let the K row indices determine vertices $\mathcal{V}$ of a graph. Let $\gamma_{i,\epsilon} = \gamma_{i,p,q}$ be the weight of the edge $\epsilon = \{p, q\}$ in the edge set $\mathcal{E}$ of the graph. Consider a matching $\Omega \subset \mathcal{E}$ such that for all distinct ϵ_k, $\epsilon_l \in \Omega$, $\epsilon_k \cap \epsilon_l = \emptyset$.Then,*

$$\max_{\mathbf{G}_i} Tr\{\mathbf{L}_i \mathbf{G}_i \mathbf{E}_i\} = \max_{\Omega} \sum_{\epsilon \in \Omega} \gamma_{i,\epsilon}, \tag{15}$$

with the optimal matching Ω^* determining the optimal pairings, i.e., the optimal $\mathbf{P}_i$, after which (13) can be used to solve for the optimal rotations, i.e., optimal $\mathbf{T}_i$.

Remark: The maximal matching problem in Proposition 3.1 can be solved using the *Blossom algorithm* [4] in polynomial time.

Algorithm 1 LGT_D($\mathbf{H}$,$\mathbf{G}_M^0$, ..., $\mathbf{G}_1^0$,$\mathbf{P}_0^0$,tol)

```
 1: Initialize $k \leftarrow 0$, $\tau \leftarrow \infty$, $\varepsilon^0 \leftarrow ||\mathbf{H} - \mathbf{G}_M^0 ... \mathbf{G}_1^0 \mathbf{P}_0^0||_F$
 2: while $\tau > tol$ do
 3:     $k \leftarrow k + 1$
 4:     $\mathbf{P}_0 \leftarrow$ solve (9) given $\mathbf{G}_i^{k-1}$, $i \in \mathcal{M}$ (Hungarian method)
 5:     $\mathcal{G}_0 \leftarrow \mathbf{G}_M^{k-1} ... \mathbf{G}_1^{k-1} \mathbf{P}_0$, $err(0) \leftarrow ||\mathbf{H} - \mathcal{G}_0||_F$
 6:     for $l \in \mathcal{M}$ do
 7:         $\mathbf{G}_l \leftarrow$ solve (10) given $\mathbf{G}_i^{k-1}$, $i \in \mathcal{M} \backslash l$, and $\mathbf{P}_0^{k-1}$
                (Blossom algorithm)
 8:         $\mathcal{G}_l \leftarrow \mathbf{G}_M^{k-1} ... \mathbf{G}_{l+1}^{k-1} \mathbf{G}_l \mathbf{G}_{l-1}^{k-1} ... \mathbf{G}_1^{k-1} \mathbf{P}_0^{k-1}$
 9:         $err(l) \leftarrow ||\mathbf{H} - \mathcal{G}_l||_F$
10:     end for
11:     $l^* \leftarrow \arg\min_l err(l)$, $\varepsilon^k \leftarrow err(l^*)$, $\tau \leftarrow \varepsilon^{k-1} - \varepsilon^k$
12:     if $l^* > 0$ then $\mathbf{G}_{l^*}^k \leftarrow \mathbf{G}_{l^*}$ else $\mathbf{P}_0^k \leftarrow \mathbf{P}_0$
13: end while
14: Return $\mathbf{G}_M^k$, ..., $\mathbf{G}_1^k$, $\mathbf{P}_0^k$, $\varepsilon^k$
```

Proposed Algorithm: Our algorithm for LGT optimization consists of two parts. LGT_D (Algorithm 1) is a greedy descent algorithm which iteratively calculates the best layer to optimize and updates that layer. LGT_A (Algorithm 2) is a stochastic annealing procedure [10] which makes $\mathcal{A}$ annealing jumps using LGT_D. A gradually cooled temperature variable modulates the jump probability [10]. As is typical with annealing techniques the best approximating LGT can

Algorithm 2 LGT_A($\mathbf{H}$,$\mathbf{G}_M^0, ..., \mathbf{G}_1^0$,$\mathbf{P}_0^0, \mathcal{A}, tol$)

```
1: $\varepsilon^0 \leftarrow ||\mathbf{H} - \mathbf{G}_M^0 ... \mathbf{G}_1^0 \mathbf{P}_0^0||_F$
2: for $k = 1 : \mathcal{A}$ do
3:     $\mathbf{G}_M, ..., \mathbf{G}_1, \mathbf{P}_0 \leftarrow \mathbf{G}_M^{k-1}, ..., \mathbf{G}_1^{k-1}, \mathbf{P}_0^{k-1}$
4:     Randomly reset $M/2 + 1$ of $\mathbf{G}_M, ..., \mathbf{G}_1, \mathbf{P}_0$ to identity.
5:     $\mathbf{G}_M, ..., \mathbf{G}_1, \mathbf{P}_0, \varepsilon \leftarrow$LGT_D( $\mathbf{H}$,$\mathbf{G}_M, ..., \mathbf{G}_1, \mathbf{P}_0, tol$)
6:     $t \leftarrow \log((\mathcal{A}+1)/k)$, $\rho \leftarrow \max(\exp((\varepsilon^{k-1} - \varepsilon)/t), 1)$
7:     $b \leftarrow Bernoulli(\rho)$ ($b = 1$ with probability $\rho$)
8:     if $b = 1$ then
9:         $\mathbf{G}_M^k, ..., \mathbf{G}_1^k, \mathbf{P}_0^k, \varepsilon^k \leftarrow \mathbf{G}_M, ..., \mathbf{G}_1, \mathbf{P}_0, \varepsilon$
10:    else
11:        $\mathbf{G}_M^k, ..., \mathbf{G}_1^k, \mathbf{P}_0^k, \varepsilon^k \leftarrow \mathbf{G}_M^{k-1}, ..., \mathbf{G}_1^{k-1}, \mathbf{P}_0^{k-1}, \varepsilon^{k-1}$
12:    end if
13: end for
14: Return $\mathbf{G}_M^k, ..., \mathbf{G}_1^k, \mathbf{P}_0^k$
```

be tracked and returned as the algorihtm result. Assuming 8×8 blocks, i.e., $K = 64$, and $M = 11$, LGT_D takes a fraction of a second ($\sim$ .1 sec.) on commodity hardware. Setting $\mathcal{A} \sim 1000$ is typically sufficient resulting in an 8×8 LGT design in a few minutes.
LGT Complexity: An LGT layer contains $K/2$ butterflies. Suppose each butterfly is carried out with computations equivalent to β multiply-adds. Note that $\beta \sim 3$ is straightforward and $\beta \sim 2$ can be accomplished [7, 6, 1], resulting in

$$\mathcal{C}_{\mathcal{LGT}}(M, K) = (\beta/2)MK \geq \mathcal{C}_{\mathcal{FLGT}}(M, K) \sim MK, \qquad (16)$$

where $\mathcal{C}_{\mathcal{FLGT}}(M, K)$ indicates the complexity for a fast implementation. For moderate K we will assume that the permutations add marginal complexity *in comparison,* since (a) modern cache sizes easily allow cached access to length-K vectors in software[1], (b) programmable multiplexers that can shuffle vectors in a single cycle can be used in hardware. Note also that the LGT structure is highly amenable to parallelization.

4. EXPERIMENTAL RESULTS

We used Algorithm 2 to approximate a set of Sparse Orthonormal Transforms [14] derived from a training set of images for 8×8 blocks, i.e., $N = 8$, $K = 64$. The SOT is a generalization of the KLT as it reduces to the KLT on Gaussian processes but provides significant improvements over the KLT on non-Gaussian data. SOTs trained on typical images and videos tend to have directional structure which is considered to be the main reason why they outperform simpler designs in applications [14]. Directionality is well-known to be difficult to approximate with computationally simple transforms.

Figure 3 illustrates the approximation results for the eight SOTs from [14] using improved separable[2] [5], row-column [5], and LGT designs[3]. The SOTs used in our experiments are primarily aligned with directions from 0 to $7\pi/8$. If a separable design incurs $\mathcal{C}_{\mathcal{S}}$ complexity then LGTs with $M = 12, 11, 10, 9$ incur $\mathcal{C}_{\mathcal{FLGT}} \sim 0.75 \times \mathcal{C}_{\mathcal{S}}, 0.69 \times \mathcal{C}_{\mathcal{S}}, 0.63 \times \mathcal{C}_{\mathcal{S}}$, and $0.56 \times \mathcal{C}_{\mathcal{S}}$ respectively.

RCTs can be seen to improve over separable approximations in all cases, peaking when approximating the near horizontal and near vertical SOT basis at 0 and $\pi/2$. As the alignment moves toward the diagonals at $\pi/4$ and $3\pi/4$ the RCT performance is reduced. Observe that the LGT with $M = 12$ provides significant improvements

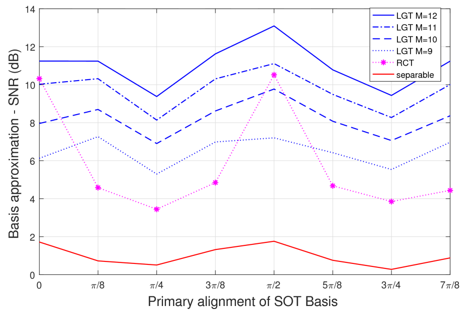


Fig. 3. LGT, RCT, and improved-separable approximations to the eight non-separable basis, SOT1-SOT8 [14].

over RCT and improved separable designs with a noticeably less performance drop at the diagonals. The LGT with $M = 11$ is similar, outperforming RCT in all cases but one. On the lower complexity end the LGT with $M = 9$ has $\mathcal{C}_{\mathcal{FLGT}} \sim 0.56 \times \mathcal{C}_{\mathcal{S}}$ while outperforming the separable design on all directions and the RCT on all but two directions.

Figures 5 and 6 provide a detailed look at the basis generated by RCT, separable, and LGT designs when approximating the SOT aligned at $3\pi/4$. As illustrated, the LGT is visually almost indistinguishable from the original at $M = 12$ and still manages to retain the directional structure at the lowest complexity level. Table 1 includes compression results using a SOT-based transform coder where the SOT is interchanged with separable, RCT, and LGT designs (refer to [14] for codec details and to Figure 4 for the test images). The LGT outperforms the other simplified designs while saving a significant amount of computation especially with a fast implementation.

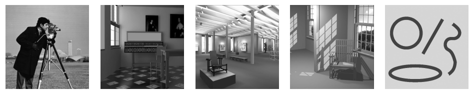


Fig. 4. Test images Camera, Vermeer, Museum, Chair, and Graphics.

5. CONCLUSIONS

We proposed layered-Givens transforms which are tunable-complexity transforms composed of layers of data permutations and Givens rotations. The proposed formulation and parametrization of LGTs allow methods from graph theory to be utilized in optimizing the permutations jointly with the Givens rotations. Thus optimized permutations are the key to LGT performance in enabling low-complexity and high-fidelity approximations of desired complex transforms. Our results clearly indicate that LGTs outperform other well-designed transform simplifications. LGTs can be extended to use non-orthogonal/nonlinear two-terminal compute units in applications like neural-networks, to incorporate weighting functions that emphasize more important basis functions in applications like compression, and to include other approximation metrics. We leave such extensions and other design improvements to another article.

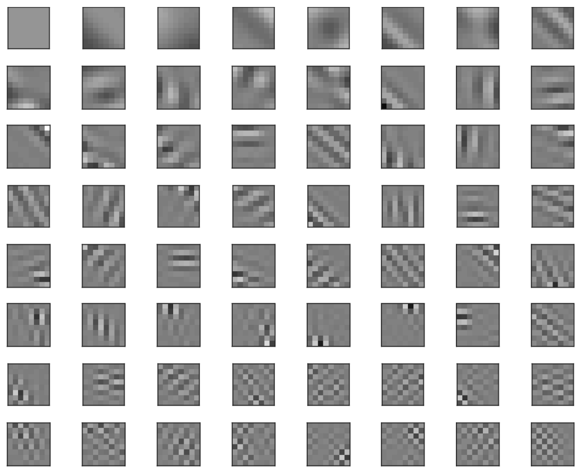
Image	Separable	RCT	LGT (M=11)	SOT
Camera	3.18%	5.32%	5.41%	8.49%
Vermeer	2.96%	5.35%	6.12%	7.42%
Museum	1.81%	5.01%	6.23%	10.47%
Chair	2.32%	4.71%	5.55%	10.34%
Graphics	9.41%	19.94%	20.32%	28.83%

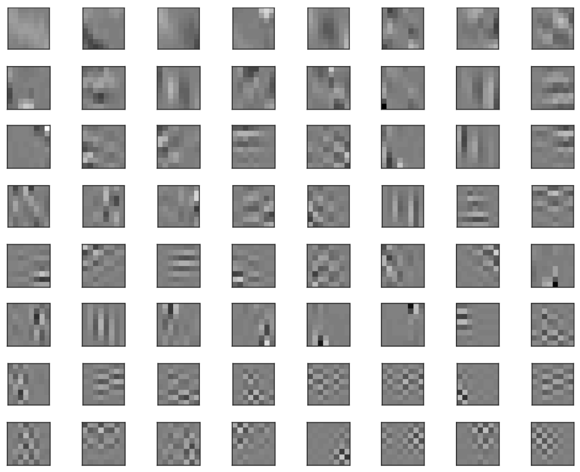
Table 1. Percentage rate gains (at constant distortion) improved Separable ($\mathcal{C}_{\mathcal{S}}$), RCT ($\mathcal{C}_{\mathcal{S}}$), LGT ($M = 11, \mathcal{C}_{\mathcal{FLGT}} \sim 0.69 \times \mathcal{C}_{\mathcal{S}}$), and SOT ($4 \times \mathcal{C}_{\mathcal{S}}$) obtain over the baseline DCT (computed using [2]).

[1]Our software implementation of the LGT on commodity hardware confirms the marginal impact of the permutations.

[2]It is worth mentioning that a naive separable approximation to the SOT basis obtains $\sim -1dB$.

[3]Initialization in Algorithm 2 was done such that the initial LGT matched the DCT.


(a) Non-separable target basis


(b) RCT ($\mathcal{C}_{\mathcal{S}}$, SNR $3.85dB$)

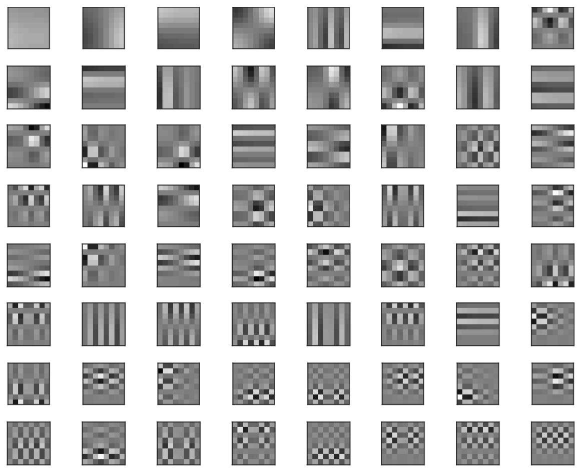
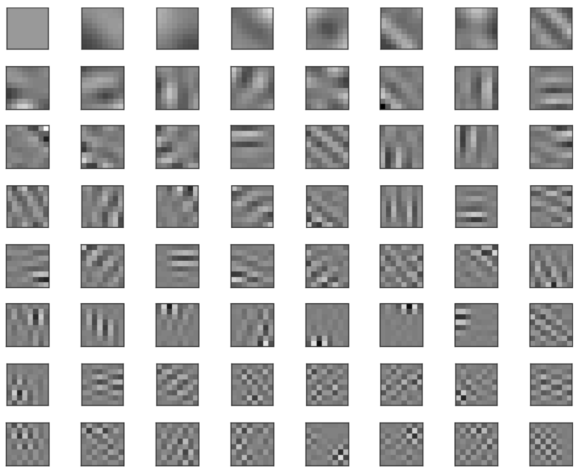
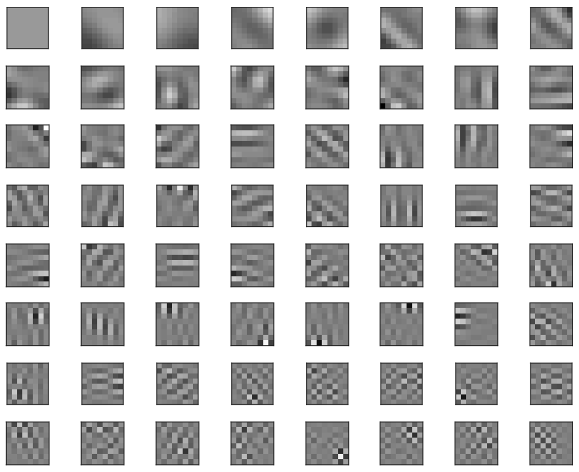

(c) Improved Separable ($\mathcal{C}_{\mathcal{S}}$, SNR $0.28dB$)

Fig. 5. RCT and improved-separable approximation performance to SOT7 (directional at $3\pi/4$). Both simplifications have been reordered to match the target basis ordering. While the separable design loses the directionality of the SOT, RCT mostly retains it.


(a) LGT, $M = 12$ ($\mathcal{C}_{\mathcal{FLGT}} \sim 0.75 \times \mathcal{C}_{\mathcal{S}}$, SNR $9.44dB$)


(b) LGT, $M = 11$ ($\mathcal{C}_{\mathcal{FLGT}} \sim 0.69 \times \mathcal{C}_{\mathcal{S}}$, SNR $8.27dB$)

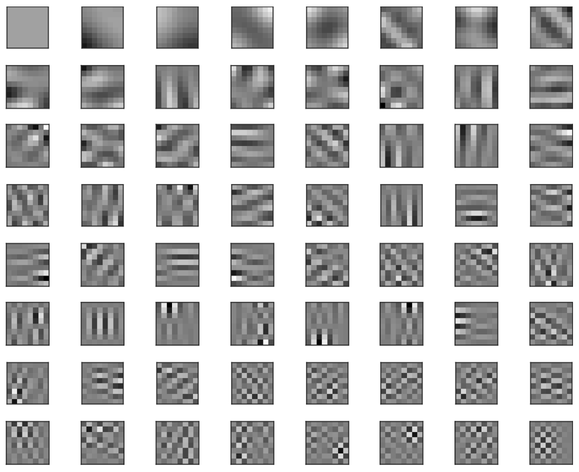

(c) LGT, $M = 9$ ($\mathcal{C}_{\mathcal{FLGT}} \sim 0.56 \times \mathcal{C}_{\mathcal{S}}$, SNR $5.54dB$)

Fig. 6. LGT approximation performance to SOT7 of Figure 5 (a). At $M = 12$, the LGT is hard to distinguish from SOT7. Even at $M = 9$, i.e., well below the separable complexity level $\mathcal{C}_{\mathcal{S}}$, LGT clearly retains the directionality.

6. REFERENCES

[1] A. A. Anda and H. Park. Fast plane rotations with dynamic scaling. *SIAM Journal on Matrix Analysis and Applications*, 15(1):162–174, 1994.

[2] G. Bjontegaard. Calculation of average psnr differences between rd-curves. *ITU-T Q.6/SG16 VCEG-M33 Contribution*, 48, Apr 2001.

[3] M. N. Do and M. Vetterli. The contourlet transform: An efficient directional multiresolution image representation. *Trans. Img. Proc.*, 14(12):2091–2106, Dec. 2005.

[4] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17(3):449–467, 1965.

[5] H. E. Egilmez, O. G. Guleryuz, J. Ehmann, and S. Yea. Row-column transforms: Low-complexity approximation of optimal non-separable transforms. In *2016 IEEE International Conference on Image Processing, ICIP 2016, Phoenix, AZ, USA, September 25-28, 2016*, pages 2385–2389, 2016.

[6] W. M. Gentleman. Least squares computations by givens transformations without square roots. *IMA Journal of Applied Mathematics*, 12(3):329, 1973.

[7] G. H. Golub and C. F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[8] V. K. Goyal. Theoretical foundations of transform coding. *IEEE Signal Processing Magazine*, 18(5):9–21, Sep 2001.

[9] J. Han, A. Saxena, V. Melkote, and K. Rose. Jointly optimized spatial prediction and block transform for video and image coding. *IEEE Trans. Image Process.*, 21(4):1874–1884, Apr. 2012.

[10] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, et al. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[11] H. W. Kuhn and B. Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

[12] K. R. Rao and P. Yip. *Discrete Cosine Transform Algorithms, Advantages, Applications*. Academic Press Professional, Inc., San Diego, CA, USA, 1990.

[13] O. Sezer, R. Cohen, and A. Vetro. Robust learning of 2-d separable transforms for next-generation video coding. In *Data Compression Conference (DCC), 2011*, pages 63–72, March 2011.

[14] O. Sezer, O. Guleryuz, and Y. Altunbasak. Approximation and compression with sparse orthonormal transforms. *Image Processing, IEEE Transactions on*, 24(8):2328–2343, Aug 2015.

[15] J.-L. Starck, E. J. Candes, D. L. Donoho, A. Wavelet, and I. Denoising. The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, 11(6):670–684, 2002.

[16] M. Vetterli and J. Kovačevic. *Wavelets and Subband Coding*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.

[17] Y. Ye and M. Karczewicz. Improved h.264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning. In *IEEE International Conference on Image Processing (ICIP)*, pages 2116–2119, Oct 2008.

[18] F. Zou, O. Au, C. Pang, J. Dai, X. Zhang, and L. Fang. Rate-distortion optimized transforms based on the lloyd-type algorithm for intra block coding. *IEEE Journal of Selected Topics in Signal Processing*, 7(6):1072–1083, Dec 2013.