Designing Sparse Graphs via Structure Tensor for Block Transform Coding of Images

Ivano Rotondo*, Gene Cheung*, Antonio Ortega† and Hilmi Enes Egilmez†
* National Institute of Informatics, Tokyo, Japan
E-mail: ivano.rotondo@gmail.com, cheung@nii.ac.jp Tel/Fax: +81-3-4212-2567
† University of Southern California, CA, USA
E-mail: antonio.ortega@sipi.usc.edu, hegilmez@usc.edu Tel/Fax: +1-213-740-2320

***Abstract*—The graph Fourier transform (GFT)—adaptive to the signal structures of local pixel blocks—has recently been shown to be a good alternative to fixed transforms, e.g., the Discrete Cosine Transform (DCT), for image coding. However, the majority of proposed GFTs assume an underlying 4-connected graph structure with vertical and horizontal edges only. In this paper, we propose a design methodology to select more general sparse graph structures and edge weights, on which GFTs are defined for block-based coding. Specifically, we first cluster blocks via the Lloyd-Max algorithm based on their principal gradients, which are eigenvectors of the computed structure tensors. For each cluster a graph template with edges orthogonal to the principal gradient is designed. Finally, optimal edge weights are computed assuming each template is a graph describing the inter-pixel correlation in a Gaussian Markov Random Field (GMRF). Experimental results show that GFTs derived from our graph templates lead to sparser signal representations and fewer encoding bits than DCT for a set of natural test images.**

I. Introduction

In conventional block-based image codecs such as JPEG, a fixed transform such as *discrete cosine transform* (DCT) is used for transform coding of a pixel block's intensity values. However, distinct signal structures (*e.g.*, gradients, discontinuities, etc) of local pixel patches vary widely throughout an image, and it is more desirable to adapt a transform to the target signal's characteristics during coding.

Recently it has been shown [1–3] that using different *graph Fourier transforms* (GFT) for transform coding, where each GFT is computed from a weighted undirected graph that reflects the target block's discontinuities (*e.g.*, an edge is assigned weight 0 if the connecting pixels straddle a discontinuity and 1 otherwise), one can obtain significant compression gain over DCT for *piecewise smooth* (PWS) images. Most of these recent approaches use a simple 4-connected graph, where each pixel can be connected to its nearest vertical and horizontal pixel neighbors only. For natural images with discontinuities and/or strong gradients in local pixel patches, this approach is not sufficiently adaptive, however, since a pixel may be most similar to its nearest diagonal neighbors, and a graph structure with diagonal edges that exploit this correlation during filtering would be more appropriate.

In this paper, we propose a design methodology to select more general sparse graph structures and edge weights, on which GFTs will be defined for block-based image coding. (Sparse graphs are preferable, since fast implementation of GFTs becomes possible via lifting [4].) Specifically, we first cluster blocks via the Lloyd-Max algorithm based on their *principal gradients*; for each code block its structure tensor is first computed, and the eigenvector corresponding to the larger eigenvalue is the principal gradient. Then, for each cluster a graph template with edges orthogonal to the principal gradient is designed, connecting most similar pixels in the cluster. Finally, optimal edge weights are computed assuming each template is a graph describing the inter-pixel correlation in a *Gaussian Markov Random Field* (GMRF). Experimental results show that GFTs derived from our designed graph templates lead to sparser signal representation and fewer encoding bits than DCT for a set of natural test images.

The outline of the paper is as follows. We first discuss related work in Section II. We review necessary mathematical tools in Section III and discuss our graph template design methodology in Section IV. Experiment results and conclusion are presented in Section V and VI, respectively.

II. Related Work

Commonly used transforms like DCT, *discrete sine transform* (DST) and *asymmetric discrete sine transform* (ADST) [5] are fixed and do not flexibly adapt to heterogeneous image structures. Directional transforms [6] offer one alternative where directionality can be incorporated into the transform design, but cannot easily adapt to arbitrarily shaped discontinuities such as "L" and "V".

To be more adaptive to local image statistics, a classical approach is to first cluster similar pixel patches into separate classes, and then for each class derive the optimal decorrelating *Karhunen-Loève Transform* (KLT) based on class statistics [7]. This purely data-driven approach leads to unstructured inverse covariance matrices, however, and thus the resulting KLTs tend to be computationally expensive to implement.

With the recent development in *graph signal processing* (GSP) [8], transforms can now be defined on an irregular data kernel described by a weighted undirected graph [1–3]. Unlike [7], GFTs derived from structured sparse graphs lead to efficient implementation via lifting [4]. However, while edge weights can be selected to reflect the similarities of connecting pixels, using an underlying 4-connected graph structure with

only vertical and horizontal edges restricts the degree of freedom when adapting to the target block's signal structure. Our proposal generalizes these schemes by considering diagonal edges as well when designing sparse graphs.

The most related work to our proposal is *graph template transform* (GTT) [9], which, assuming a probabilistic interpretation of signal in a code block, proposed to find sparse inverse covariance matrices given a set of training blocks, similar to graphical lasso proposed in [10]. Our proposal differs in the following regards. First, while [9] considered up to 24-connected graphs (each pixel is connected to its 24 nearest pixels), all our graph templates remain 4-connected and thus are sparser, which can lead to faster implementation via lifting [4]. Second, we cluster blocks based solely on principal gradients derived from computed structure tensor, which leads to fast identification of blocks to appropriate clusters. Finally, we assume that all edges with the same orientation in a given template have the same weight, leading to more robust edge weight estimates than [9] where each edge in a graph template can take on a different weight.

III. Preliminaries

A. Graph Fourier Transform

A graph $\mathcal{G}$ is defined by a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$ that connect them. Each edge $k \in \mathcal{E}$ connects nodes $i = e_1(k)$ and $j = e_2(k)$, $i, j \in \mathcal{V}$, with weight $w_{i,j}$. Denote by $\mathbf{W}$ the *adjacency matrix*, where $W_{i,j} = w_{i,j}$. Denote by $\mathbf{D}$ the *degree matrix*, where $d_{i,i} = \sum_j W_{i,j}$. The combinatorial Laplacian $\mathbf{L}$ is defined as:

$$\mathbf{L} = \mathbf{D} - \mathbf{W} \tag{1}$$

Given $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$, we define the GFT for $\mathcal{G}$ as the eigen-matrix of the graph Laplacian $\mathbf{L}$, *i.e*, Φ such that

$$\Phi \Lambda \Phi^T = \mathbf{L} \tag{2}$$

where Λ is a diagonal matrix with eigenvalues of $\mathbf{L}$ as its diagonal entries. Given a graph-signal $\mathbf{x}$ on $\mathcal{G}$, its GFT coefficients α can be computed as $\alpha = \Phi^T \mathbf{x}$. While there exist other definitions of graph Laplacian and corresponding transforms, we prefer the combinatorial graph Laplacian in (1), since its corresponding GFT guarantees a DC component, which is important for natural images that tend to be smooth. See [8] for more details.

B. Structure Tensor

The 2D structure tensor $S_w(\mathbf{p})$ of a pixel patch centered at pixel location $\mathbf{p}$ is computed as follows[1]:

$$= \begin{bmatrix} \sum_{\mathbf{r}} w(\mathbf{r})(I_x(\mathbf{p}-\mathbf{r}))^2 & \sum_{\mathbf{r}} w(\mathbf{r}) I_x(\mathbf{p}-\mathbf{r}) I_y(\mathbf{p}-\mathbf{r}) \\ \sum_{\mathbf{r}} w(\mathbf{r}) I_x(\mathbf{p}-\mathbf{r}) I_y(\mathbf{p}-\mathbf{r}) & \sum_{\mathbf{r}} w(\mathbf{r})(I_y(\mathbf{p}-\mathbf{r}))^2 \end{bmatrix} \tag{3}$$

where $w(\mathbf{r})$ is a weight parameter for displacement vector $\mathbf{r}$ such that $\sum_{\mathbf{r}} w(\mathbf{r}) = 1$. I_x and I_y are the partial derivatives with respect to the x- and y- axis, respectively.

[1] https://en.wikipedia.org/wiki/Structure_tensor

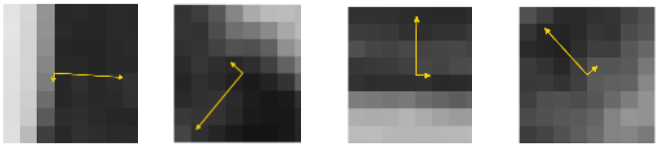
By performing eigen-decomposition on the 2D structure tensor $S_w(\mathbf{p})$, one can obtain eigenvalues λ_1 and λ_2, where $\lambda_1 \geq \lambda_2 \geq 0$, and corresponding eigenvectors v_1 and v_2 that describe the gradient $\nabla I = (I_x, I_y)$ of the patch centered at $\mathbf{p}$. v_1 corresponding to the larger λ_1 is the *principal gradient*. For example, $\lambda_1 \approx \lambda_2 \approx 0$ would imply the patch is mostly flat with little or no detectable gradient in any direction. In contrast, $\lambda_1 \gg \lambda_2 \approx 0$ would indicate a dominant principal gradient in the patch in direction v_1. Finally, if λ_1 and λ_2 are both large, then the patch has complex structure and there is no one dominant gradient direction.

IV. Computing Optimal Graph

Given a training set of N pixel blocks, we now describe four steps in our proposed block-based coding system. First, M of N training blocks with dominant principal gradients are identified for GFT coding; blocks without dominant principal gradients are encoded using conventional DCT. Second, the M selected blocks are clustered into K groups via the Lloyd-Max algorithm. Third, for each cluster a graph template is designed. Finally, appropriate edge weights are computed for each designed template. The designed graph for each cluster is used to derive a GFT as previously described.

During actual encoding, a code block is first assigned to one of K clusters ($K+1$ clusters including default cluster that uses DCT) based on its computed principal gradient. It is then encoded using the GFT corresponding to that cluster. The assigned cluster index is encoded as side information for correct decoding at the decoder.

A. Identifying Blocks for Clustering



(a) block 4 (b) block 55 (c) block 88 (d) block 130

Fig. 1. Example 8×8 code blocks with corresponding eigenvectors of their computed structure tensor drawn in yellow.

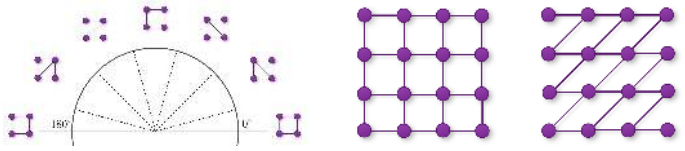
The goal of the first step is to identify blocks with dominant principal gradients. We accomplish this by examining the two eigenvalues of the computed structure tensor matrix. Specifically, given eigenvalues λ_1^n and λ_2^n of block n, where $\lambda_1^n \geq \lambda_2^n \geq 0$, we declare the block to have a dominant principal gradient iff $\lambda_1^n - \lambda_2^n > \delta$, where δ is a predefined threshold. Recall that the sizes of the structure tensor eigenvalues reflect the strength of the gradients in the block. If there is a significant difference between them, then the patch has a dominant principal gradient that an adaptive GFT can exploit for coding gain. See Fig. 1 for examples of blocks with dominant principal gradients.

B. Lloyd-Max Clustering

Given M blocks in the training set with strong principal gradients, the next step is to cluster them into K groups via

the Lloyd-Max (LM) algorithm [11]. Specifically, each block n is associated with a principal gradient of angle θ_n, where $0^o \leq \theta_n < 180^o$. Using θ_n as the variable of interest, we are essentially seeking K non-uniform quantization bins for θ_n, so that: i) each bin $\mathcal{B}_k$ has a centroid ϕ_k that minimizes the mean squared error (MSE) $\sum_{n\in\mathcal{B}_k} |\phi_k - \theta_n|^2$ with respect to blocks n assigned to bin $\mathcal{B}_k$, and ii) each block n is assigned to a bin $\mathcal{B}_k$ with the closest centroid: $k = \arg\min_k |\phi_k - \theta_n|^2$. The LM algorithm finds these K bins that satisfy both conditions.

C. Graph Template Design



(a) gradient-based clustering (b) template 4 (c) template 5

Fig. 2. (a) examples of six clusters; (b) cluster 4 with vertical and horizontal edges; (b) cluster 5 with 45^o and horizontal edges.

Given the clustering described previously, we now discuss how a graph template can be designed for each cluster. We have two design criteria for each template: i) it contains edges that connect pairs of most similar neighboring pixels in the cluster, and ii) it is a fully connected graph. The second criterion is important; a disconnected graph would result in a GFT with multiple DC components (one for each connected component), which is not efficient for coding. To fulfill these criteria, we propose a three-step construction procedure for a $h \times h$ template for each cluster as follows:

1) Select an edge orientation ψ_1 from the set $\{0^o, 45^o, 90^o, 135^o\}$ that is most orthogonal to the cluster's centroid ϕ_k.
2) Select another edge orientation ψ_2 from the set $\{0^o, 90^o\}\backslash\{\psi_1\}$ that is most orthogonal to the cluster's centroid ϕ_k.
3) Draw all edges in the $h \times h$ template with orientation ψ_1 or ψ_2 to connect neighboring nodes in the graph.

One can see that this procedure always returns a 4-connected graph template. Fig. 2(a) shows an example clustering with six clusters. Examples of templates generated from our procedure is shown in Fig. 2(b) and (c).

D. Edge Weight Computation

For each graph template with two sets of edges with orientation ψ_1 and ψ_2, we now compute the edge weights, where we assume edges of the same orientation have the same weight. We first write the graph Laplacian $\mathbf{L}$ as a weighted sum of "mini-Laplacians" $\mathbf{L}_k$ with weights w_1 and w_2:

$$\mathbf{L} = w_1 \sum_{k\in l_1} \mathbf{L}_k + w_2 \sum_{k\in l_2} \mathbf{L}_k \tag{4}$$

where l_1 and l_2 are the two set of edges, and each mini-Laplacian $\mathbf{L}_k$ accounts only for one edge k with end nodes $e_1(k)$ and $e_2(k)$:

$$\mathbf{L}_k(i,j) = \begin{cases} -1 & \text{if } i \neq j \text{ and } i,j \in \{e_1(k), e_2(k)\} \\ 1 & \text{if } i = j \text{ and } i \in \{e_1(k), e_2(k)\} \\ 0 & \text{o.w.} \end{cases} \tag{5}$$

As an example, the mini-Laplacian $\mathbf{L}_1$ that accounts for an edge $e_1 = (1,2)$ that connects node 1 and 2 is:

$$\mathbf{L}_1 = \begin{bmatrix} 1 & -1 & 0 & \dots \\ -1 & 1 & 0 & \dots \\ 0 & 0 & 0 & \dots \\ \vdots & \vdots & \vdots & \end{bmatrix} \tag{6}$$

Writing Laplacian $\mathbf{L}$ in the form (4) shows that the eigenvectors of $\mathbf{L}$ depend only on the relative size of w_1 to w_2 rather than the absolute values of w_1 and w_2.

From [12] we know that if $\mathbf{L}$ is a *precision matrix* for a GMRF signal model, then graph template $\mathcal{G}$ describes the conditional independence of variables on $\mathcal{G}$; *i.e.*, $x_i \perp x_j \iff L_{i,j} = 0$. Further, we know that:

$$corr(x_i, x_j \,|\, \mathbf{x}_{-ij}) = -\frac{L_{i,j}}{\sqrt{L_{i,i}L_{j,j}}} \tag{7}$$

where $\mathbf{x}_{-ij}$ denotes all variables in $\mathbf{x}$ except x_i and x_j, and $corr(x_i, x_j \,|\, \mathbf{x}_{-ij})$ is the conditional correlation between x_i and x_j given $\mathbf{x}_{-ij}$. To compute the relative size w_1 to w_2, we can write:

$$\frac{w_1}{w_2} = \frac{corr(x_i, x_j \,|\, \mathbf{x}_{-ij})\sqrt{L_{i,i}L_{j,j}}}{corr(x_s, x_t \,|\, \mathbf{x}_{-st})\sqrt{L_{s,s}L_{t,t}}} \tag{8}$$

where $i,j \in \{e_1(l_1), e_2(l_1)\}$ and $s,t \in \{e_1(l_2), e_2(l_2)\}$. Conditional $corr(x_i, x_j|\mathbf{x}_{-ij})$ is hard to compute, so we apply the approximation:

$$\frac{corr(x_i, x_j \,|\, \mathbf{x}_{-ij})}{corr(x_s, x_t \,|\, \mathbf{x}_{-st})} \approx \frac{corr(x_i, x_j)}{corr(x_s, x_t)} \tag{9}$$

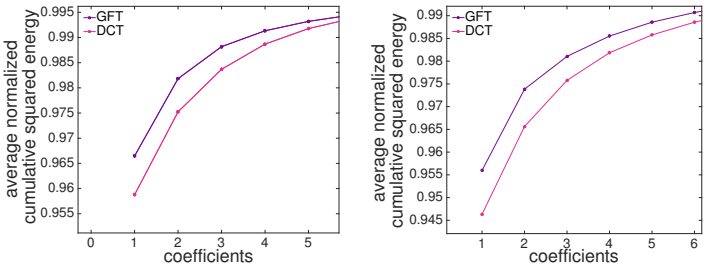
so that $corr(x_i, x_j)$ can be computed from data in cluster $\mathcal{B}$. Having computed the relative weight w_1/w_2 in (8), the resulting GFT is then the KLT that optimally decorrelates the input signal $\mathbf{x}$.

V. Experimentation

To test the effectiveness of our proposal, we use three standard natural test images `lena`, `barbara` and `elaine` and a fingerprint image as our test set. Each image is a grayscale images of 512×512 resolution with 8 bits bit-depth. We use non-overlapping 8×8 pixel blocks in these images and set $\delta = 200$ for our experiments.

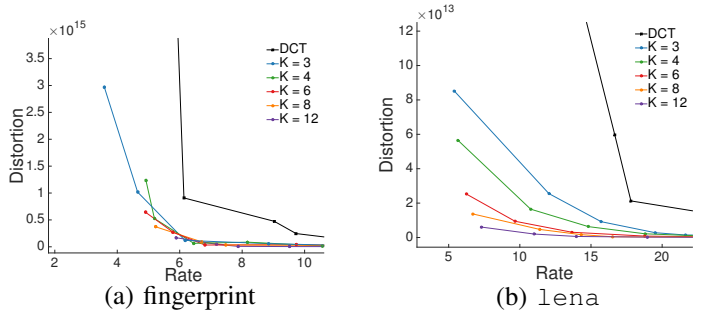
Fig. 3 show the average normalized cumulative squared energy for coded blocks in the fingerprint image and `lena`, when the number of clusters is $K = 6$. We observe that our proposed GFT can accumulate more energy with fewer coefficients than DCT; *i.e.,* GFT can achieve more compact signal representations.

While more clusters in general can lead to sparser signal representations, the side information coding cost of cluster indices also increases. Given blocks classified into K clusters



(a) fingerprint (b) lena

Fig. 3. Total energy versus number of non-zero transform coefficients used for for signal reconstruction when $K = 6$.



(a) fingerprint (b) lena

Fig. 4. Total distortion versus rate for different combination of cluster number K and QP Q.

$\{\mathcal{B}_1, \ldots, \mathcal{B}_K\}$ and a quantization parameter Q, we estimate the distortion and rate as follows. Let $d_Q(n)$ be the distortion of a transform-coded block n given QP Q. The total distortion for the training set is:

$$D(Q, K) = \sum_{k=1}^{K} \sum_{n \in \mathcal{B}_k} d_Q(n) \tag{10}$$

Let the number of non-zero transform coefficients of block n be $\xi_Q(n)$. It is expected that as Q increases, more high-frequency coefficients are rounded to zero, resulting in a smaller $\xi_Q(n)$. We estimate the cost of encoding a cluster index k to be $\sum_k -p_k \log(p_k)$, where p_k is the probability of cluster k. The total rate for the training set is then:

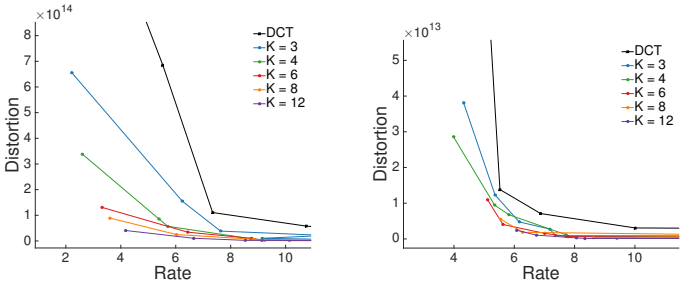
$$R(Q, K) = \sum_{k=1}^{K} \left(-p_k \log(p_k) + \mu \sum_{n \in \mathcal{B}_k} \xi_Q(n) \right) \tag{11}$$

where μ reflects the cost of coding transform coefficients relative to transform indices.

Given distortion (10) and rate (11) defined above, we try different combinations of K and Q, and then trace the convex hull as the RD performance of our proposal. Fig. 4 and 5 show the RD points for different combinations of K and Q for the four test images. We see that for low rate, a small K is preferable, while for large rate, a larger K is more appropriate. In any case, the convex hull of GFT is lower than DCT, showing that our proposal is better than DCT at any rate region.

VI. Conclusion

To adapt to varying signal structures in code blocks, in this paper we generalize previous work in graph Fourier transforms



(a) barbara (b) elaine

Fig. 5. Total distortion versus rate for different combination of cluster number K and QP Q.

(GFT) defined on undirected graphs by considering more general sparse graph structures that capture principal gradients in code blocks, which are derived from computed structure tensor. Edge weights of a fixed graph template are computed simply assuming a Gaussian Markov Random Field (GMRF) image model. Experimental results show that GFTs derived from our designed graph templates lead to sparser signal representations than DCT for a set of natural images.

References

[1] G. Shen, W.-S. Kim, S.K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *IEEE Picture Coding Symposium*, Nagoya, Japan, December 2010.

[2] W. Hu, G. Cheung, X. Li, and O. Au, "Depth map compression using multi-resolution graph-based transform for depth-image-based rendering," in *IEEE International Conference on Image Processing*, Orlando, FL, September 2012.

[3] W. Hu, G. Cheung, A. Ortega, and O. Au, "Multi-resolution graph Fourier transform for compression of piecewise smooth images," in *IEEE Transactions on Image Processing*, January 2015, vol. 24, no.1, pp. 419–433.

[4] Y.-H. Chao, A. Ortega, W. Hu, and G. Cheung, "Edge-adaptive depth map coding with lifting transform on graphs," in *31st Picture Coding Symposium*, Cairns, Australia, May 2015.

[5] J. Han, A. Saxena, V. Melkote, and K. Rose, "Jointly optimized spatial prediction and block transform for video and image coding," in *IEEE Transactions on Image Processing*, April 2012, vol. 21, no.4, pp. 1874–1884.

[6] B. Zeng and J. Fu, "Directional discrete cosine transforms–a new framework for image coding," in *IEEE Transactions on Circuits and Systems for Video Technology*, March 2008, vol. 18, no.3, pp. 305–313.

[7] G. Martinelli, L. P. Ricotti, and G. Marcone, "Neural clustering for optimal KLT image compression," in *IEEE Transactions on Signal Processing*, April 1993, vol. 41, no.4, pp. 1737–1739.

[8] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," in *IEEE Signal Processing Magazine*, May 2013, vol. 30, no.3, pp. 83–98.

[9] E. Pavez, H. Egilmez, Y. Wang, and A. Ortega, "GTT: Graph template transforms with applications to image coding," in *31st Picture Coding Symposium*, Cairns, Australia, May 2015.

[10] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," in *Biostatistics*, 2008, vol. 9, no.3, pp. 432–441.

[11] Allen Gersho and Robert M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[12] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*, vol. 104 of *Monographs on Statistics and Applied Probability*, Chapman & Hall / CRC, 2005.