GTT: Graph Template Transforms with Applications to Image Coding

Eduardo Pavez, Hilmi E. Egilmez, Yongzhe Wang and Antonio Ortega
Signal and Image Processing Institute, University of Southern California
{pavezcar, hegilmez, yongzhew}@usc.edu, antonio.ortega@sipi.usc.edu

***Abstract*—The Karhunen-Loeve transform (KLT) is known to be optimal for decorrelating stationary Gaussian processes, and it provides effective transform coding of images. Although the KLT allows efficient representations for such signals, the transform itself is completely data-driven and computationally complex. This paper proposes a new class of transforms called graph template transforms (GTTs) that approximate the KLT by exploiting a priori information known about signals represented by a graph-template. In order to construct a GTT (i) a design matrix leading to a class of transforms is defined, then (ii) a constrained optimization framework is employed to learn graphs based on given graph templates structuring a priori known information. Our experimental results show that some instances of the proposed GTTs can closely achieve the rate-distortion performance of KLT with significantly less complexity.**

***Index Terms*—Graph-based transform, learning graphs, KLT, image coding, video coding**

I. Introduction

Transform coding is an essential component for image/video compression standards in which the discrete-cosine transform (DCT) is undoubtedly the most popular transform. This is mainly due to practical reasons given that a reasonable energy compaction can be attained with low-complexity implementations. However, the DCT is known to be optimal only for the class of signals characterized by a highly correlated first-order Gaussian Markov model. To support a broader class of signals, DCT/DST-based hybrid transforms [1] have been proposed and recently adopted in state-of-the-art video coding standards such as HEVC [2]. This transforms are more efficient for prediction residual signals, but are not well adapted to other signals, for example, images including textures. As an alternative, the Karhunen-Loeve transform (KLT) provides optimal energy compaction for the data, but it is completely data-driven (i.e., obtained from an empirical covariance matrix) and does not have a fast implementation. In general, the empirical covariance is a dense matrix which requires signaling of a full matrix for different classes of image blocks, introducing a considerable amount of overhead; therefore these transforms may not be competitive in terms of rate-distortion performance. This naturally leads to the problem of designing a transform that approximates KLT but with less complexity and overhead.

This paper proposes a new class of transforms called graph template transforms (GTTs) that approximates inverse covariance matrices under sparsity constraints. For this purpose, first a set of graph templates are constructed to define the sparsity pattern of the approximate inverse covariance matrices, or equivalently, to define connectivity pattern of a graph that reveal inter-sample similarities among pixels. Then, a constrained optimization problem is solved to learn a graph by optimizing the entries of the matrix of interest. Finally, the resulting matrix is used to derive GTTs.

In the literature, several estimation methods have been proposed to construct graphs from data for various applications. Friedman *et.al.* [3] propose graphical lasso to estimate sparse inverse covariance matrix. More recently in [4] and [5], a sparse combinatorial Laplacian matrix is estimated from the data samples by including a smoothness functional. In these works, a regularization parameter is used to control sparsity, and this usually leads to arbitrarily connected graphs. However, in our work, we propose using a graph template to impose a sparsity pattern, then the empirical inverse covariance is approximated based on that template. There are several advantages of introducing a graph template for image/video coding applications. Firstly, it can include prior information about different classes of block signals. For example, samples with high vertical correlation may benefit from a template graph with less or no horizontal links. Secondly, the parameter estimation is more robust to lack of sufficient data or to noise, given the reduced number of total parameters to estimate. Finally, the number of graph parameters to transmit can be reduced with respect to an arbitrary graph transform or the KLT because the graph template is fixed and known a priori, and can be pre-stored at the decoder side.

Recently, graph signal processing [6] has drawn a lot of interest for its ability to embed a priori signal information in a graph. For block-based compression applications, it has been demonstrated that graph-based transforms can provide significant coding gains for piecewise-smooth images such as depth maps [7], [8], [9] which exploit sharp boundaries in depth images to create graphs. In this paper, we show that the proposed GTTs can be also used for natural image compression, focusing in particular on texture images, for which a piecewise smooth image model would not be suitable.

The rest of the paper is organized as follows. Section II discusses some preliminaries. The proposed GTTs and various design methods are presented in Section III. Experimental results are described in Section IV. Section V draws some

conclusions.

II. Preliminaries

A. Karhunen-Loeve transform (KLT)

Consider a set of N image blocks formed as column vectors $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ of size K, with empirical covariance matrix $\mathbf{S}$. We assume that these vectors have been chosen from a set of block signals sharing some properties/characteristics of interest (e.g., they might have a structure such as edges with similar orientation). The empirical KLT is given by the eigenvector matrix $\mathbf{\Phi}$ of $\mathbf{S} = \mathbf{\Phi}\Lambda\mathbf{\Phi}^t$. This is the optimal decorrelating transform for such signals.

B. Graph-based Transform (GBT)

Given an undirected graph $G = (V, E, \mathbf{Q})$ consisting of a collection of nodes $V = \{1, 2, ..., K\}$ connected by a set of edges E and matrix representation $\mathbf{Q}$, where for $i \neq j$ the link $(i, j) \notin E$ if and only if the weight $\mathbf{Q}_{ij} = 0$. We assume $\mathbf{Q}$ is a symmetric positive semi-definite matrix. Note that our graph definition is different from the ones used in recent graph signal processing literature, where $\mathbf{Q}$ is usually restricted to be a graph Laplacian derived from an adjacency matrix [6]. After deciding on a $\mathbf{Q}$ matrix, we first perform an orthonormal eigen-decomposition,

$$\mathbf{Q} = \mathbf{U}\Lambda\mathbf{U}^t. \tag{1}$$

and the associated graph-based transform (GBT) is defined as the orthonormal matrix $\mathbf{U} = \{\mathbf{u}_1, ..., \mathbf{u}_K\}$, with corresponding forward transform given by

$$\tilde{\mathbf{x}} = \mathbf{U}^t\mathbf{x}. \tag{2}$$

Note that, if we choose $\mathbf{Q} = \mathbf{S}$ or $\mathbf{Q} = \mathbf{S}^{-1}$, then we simply obtain the KLT basis. This is because any non-singular diagonalizable matrix and its inverse have the same set of eigenvectors.

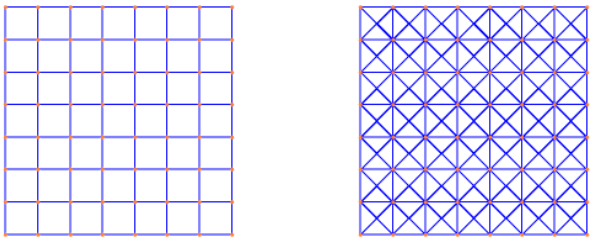
III. Graph Template Transforms

In this section, we design three different kinds of $\mathbf{Q}$ matrices that provide sparse approximations for the inverse covariance matrix, $\mathbf{S}^{-1}$. Since the inverse covariance matrix is dense in general, we introduce graph templates to set a sparse nonzero pattern for $\mathbf{Q}$ that approximates $\mathbf{S}^{-1}$. We also present two optimization problems where the graph templates impose constraints on the optimization. The resulting sparse matrix $\mathbf{Q}$ is used to construct the graph template transforms (GTT).

A. Matrix Types

The empirical covariance matrix is modeled as $\mathbf{S}^{-1} = \mathbf{Q} + \mathbf{N}$, where $\mathbf{Q}$ is a sparse positive semi-definite matrix and $\mathbf{N}$ an error term. Assuming the link set E is fixed, we consider three types of matrices:

1) Sparse inverse covariance (SIC): $\mathbf{Q}$ is a positive semi-definite matrix, where for every $i \neq j$ $\mathbf{Q}_{ij} = 0$ when $(i, j) \notin E$.



(a) (b)

Fig. 1: Graphs (a) connecting each pixel with its four nearest neighboring pixels (4-connected) and (b) connecting each pixel with pixels that are 1-hop away (8-connected).

2) Generalized Laplacian (GL): $\mathbf{Q}$ is a GL matrix if it is a SIC matrix with non-positive off-diagonal terms, i.e., $\mathbf{Q}_{ij} \leq 0$ if $i \neq j$.
3) Combinatorial Laplacian (L): $\mathbf{Q}$ is a combinatorial Laplacian, if it is a singular generalized Laplacian with an all ones eigenvector, i.e., $\mathbf{Q}_{ii} = -\sum_{j \neq i} \mathbf{Q}_{ij}$.

Given a fixed edge set E, the SIC, GL and L matrices have a decreasing number of parameters. A sparse inverse covariance will provide a better approximation for the inverse covariance, but given the arbitrary sign of the off diagonal entries, it uses more overhead. Also it is not straightforward to give an interpretation as a weighted graph, where the edge weights have positive and negative values. If the data were to follow a Gaussian distribution and $\mathbf{N} = \mathbf{0}$, $\mathbf{Q}$ is called the precision matrix, and $\mathbf{Q}_{ij} \neq 0$ if and only if the random variables corresponding to nodes i and j are conditionally independent, given all the other nodes, leading to a Gaussian Markov Random Field (GMRF) process. The generalized Laplacian has a fixed sign pattern, thus reducing the overhead. Furthermore, the off diagonal elements can be interpreted as the edge similarities between pixels. The combinatorial Laplacian further reduces the number of parameters, because its diagonal entries can be directly computed. Also, the first eigenvector associated with zero eigenvalue corresponds to DC component which is often desirable in image compression applications.

B. Template selection

Many image processing applications have adopted the nearest-neighbor image model for natural image compression [10], and texture analysis [11]. In terms of our graph-based notation, this model is equivalent to unweighted 4-connected grid graph as shown in Fig.1(a) which derives the traditional 2D-DCT [12]. Different transforms can be obtained by choosing different graphs like the one shown in Fig.1(b) which allows diagonal connections.

Assuming the pixels are localized in an uniform grid, we use three different graph templates (edge sets E), whose corresponding weights are optimized

1) 4-*connected graph template (E_4)*: Each pixel is connected to its 4 nearest neighbors which corresponds to horizontal and vertical edges as shown in Fig.1(a).

2) 8-*connected graph template (E_8)*: This template extends the 4-connected graph template by allowing diagonal connections so that each pixel is connected to its 8 nearest neighbors as shown in Fig.1(b).
3) 24-*connected graph template (E_{24})*: This template connects each pixel to its 24 closest neighbors. This allows connections with farther pixels and in additional diagonal directions.

One can show that, if the image blocks have size $\sqrt{K} \times \sqrt{K}$. The 8-connected graph template has $|E| = 2(\sqrt{K}-1)(2\sqrt{K}-1)$ edges. Hence the SIC and GL matrices have $|E| + K$ nonzero entries, while the L matrix has $|E|$ nonzero entries. Thus the parameter complexity of the GTT is $\mathcal{O}(K)$, scaling linearly on the number of pixels per block. This is smaller than the KLT, which requires $K(K+1)/2$ parameters to store a symmetric matrix (i.e., fully connected graph template).

Different graph templates can be also used to define sparsity patterns capturing other signal characteristics. Yet, we choose three basic templates for simplicity. The optimization methods used to design weights are discussed in the next section.

C. Optimization method

To find a matrix of interest $\mathbf{Q}$, we solve one of the following convex optimization problems,
(i) Sparse left inverse (SLI):

$$\mathbf{Q} = \underset{\mathbf{A}\in\Gamma}{\arg\min} \|\mathbf{AS} - \mathbf{I}\|_F \tag{3}$$

(ii) Gaussian maximum likelihood (GML):

$$\mathbf{Q} = \underset{\mathbf{A}\in\Gamma}{\arg\min} \left(\log\det(\mathbf{A}) - tr(\mathbf{SA})\right) \tag{4}$$

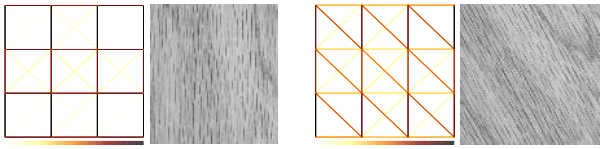
where Γ denotes the constraint set given by a *matrix type - graph template* pair. For example $\Gamma(\mathrm{GL}, E_8)$ is the set of all generalized Laplacians with edge set E_8. Note that the problem in (3) approximates the inverse covariance matrix by a sparse left inverse (SLI). In (4), the problem abbreviated as GML can be interpreted as a constrained maximum-likelihood (ML) estimation under Gaussian-Markov assumptions. Hence, solving (4) is equivalent to approximating the data as a GMRF with precision matrix $\mathbf{Q}$. In [3], authors solve $\min_{\mathbf{A}\in\Gamma} \log\det(\mathbf{A}) - tr(\mathbf{SA}) + \lambda\|\mathbf{A}\|_1$, with $\mathbf{A}$ positive semidefinite, where the extra term is included to promote sparsity in the solution. Our formulation in (4) is a constrained version with $\lambda = 0$, because the ℓ_1 term is not necessary since we impose a fixed sparsity pattern. In practice these methods would produce sparse graphs by zeroing out coefficients below a certain threshold.

IV. Results

In this section, we evaluate the performance of different GTT designs by benchmarking against DCT and KLT in terms of rate-distortion (RD) performance. GTTs are investigated extensively by selecting different (i) optimization methods, (ii) design matrices and (iii) connectivity constraints/patterns whose details are discussed in Section III.

In our experiments, we use the USC-SIPI texture dataset[1]

[1]USC-SIPI Image Dataset: http://sipi.usc.edu/database/



(a) (b)

Fig. 4: An example of 8-connected graphs designed for *wood* image with (a) no rotation and (b) with 30 degree counter-clockwise rotation. The darker colors correspond to larger weights.

which includes thirteen *Brodatz* texture images and their different pre-processed versions. In particular, we choose all thirteen original *Brodatz* textures and their 30 degree counter-clockwise rotated versions so that a total of 26 images are used. For each image, KLT and GTT transforms are designed based on a covariance matrix generated using 4×4 or 8×8 non-overlapping image blocks as data samples. Fig.4 illustrates two sample graphs associated to two different GTTs designed for *wood* image and its rotated version, respectively. As seen in Fig.4, the graphs allow us to capture anisotropic behavior in images by adjusting the edge-weights.

The transforms are applied directly on the images and the transform coefficients are first uniformly quantized and then encoded using a symbol grouping-based arithmetic entropy encoder called AGP, which uses an amplitude group partition technique to efficiently encode image transform coefficients [13]. The AGP encoder allows us to fairly compare the RD performance of different transforms since AGP can flexibly learn and exploit amplitude distribution of transform coefficients. For ordering of the quantized coefficients, we employ zig-zag scanning for DCT coefficients, and descending and ascending order of eigenvalues are used for KLT and GTT coefficients, respectively. After decoding the quantized transform coefficients using AGP decoder, we reconstruct the texture image and PSNR with respect to the original image is calculated.

The average RD performances of different transforms are presented in Fig.2 and Table I in terms of PSNR and total bits spent for encoding quantized transform coefficients per-pixel (BPP). Fig.2 compares GTTs generated based on different design matrix types (L, GL and SIC) with different connectivity constraints against traditional DCT and KLT. More comprehensive results are available in Table I where we show percent bit reductions gained by using GTTs and KLT at different PSNR values (i.e., 28, 32 and 36 dBs) with respect to using DCT. Note that, positive values in the table means that better RD performance (bit reduction) is achieved compared to using DCT. According to these results:

- Encoding performance of GTTs can closely approach that of KLT with reduced number of parameters.
- For sparse graph templates, GL matrices lead to a better performance, while for dense graph templates (i.e., graphs with 24 or more connected templates) SIC matrices work better.
- The best results shown in Table I are similar for different

TABLE I: Average percentage reduction in bitrate (BPP) with respect to average bitrate obtained with DCT.

PSNR	Matrix	4 × 4 block transform							8 × 8 block transform						
		GML			SLI			KLT	GML			SLI			KLT
		E_4	E_8	E_{24}	E_4	E_8	E_{24}		E_4	E_8	E_{24}	E_4	E_8	E_{24}	
28	L	1.050	2.619	2.651	N/A	N/A	N/A	5.014	2.224	5.176	5.443	N/A	N/A	N/A	11.17
	GL	**1.594**	**3.084**	3.124	**1.700**	**3.306**	**3.284**		**3.070**	**5.794**	6.304	**1.161**	**3.894**	2.808	
	SIC	**1.597**	1.247	**4.609**	1.662	1.134	**4.638**		**3.073**	1.238	**8.743**	0.752	-0.859	**8.411**	
32	L	1.492	2.866	2.837	N/A	N/A	N/A	6.118	0.896	3.292	3.240	N/A	N/A	N/A	10.49
	GL	**2.245**	**3.519**	3.555	**0.833**	**1.935**	1.997		**2.163**	**4.420**	4.638	-1.890	-1.247	-4.408	
	SIC	**2.248**	1.480	**5.462**	0.628	0.982	**5.322**		**2.167**	-0.568	**8.340**	-2.467	-3.366	**6.303**	
36	L	0.790	1.574	1.474	N/A	N/A	N/A	5.470	-0.602	0.862	0.520	N/A	N/A	N/A	8.728
	GL	**1.594**	**2.464**	2.460	-1.121	-0.237	-0.439		**0.849**	**2.277**	2.158	-4.058	-4.833	-8.428	
	SIC	**1.599**	1.244	**4.804**	-1.319	**0.792**	**4.563**		**0.857**	-0.579	**6.698**	-4.551	-2.511	**4.474**	

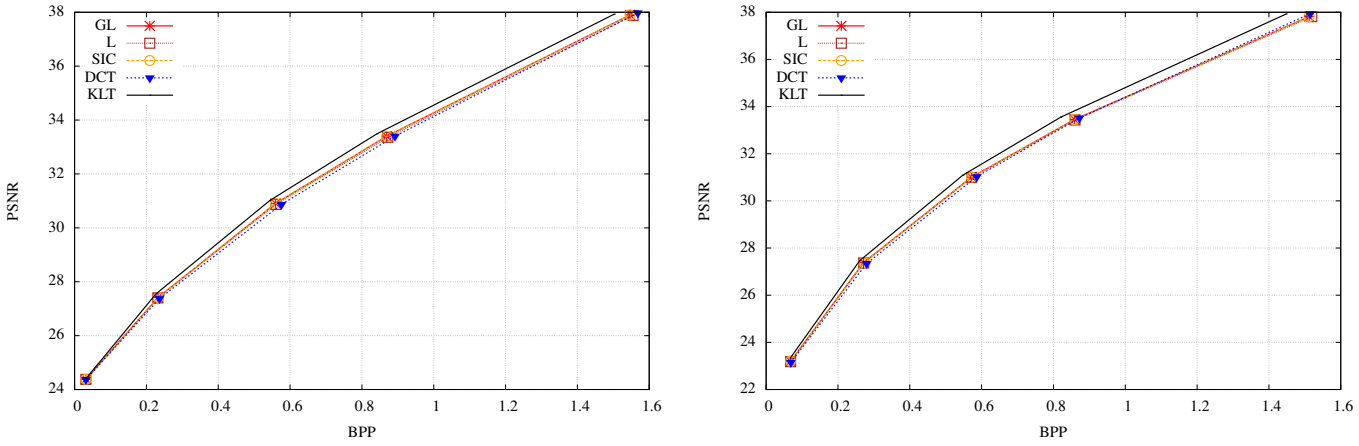


Fig. 2: Average PSNR vs. BPP results using GML method with (left) 4 and (right) 8 connected models for 4×4 transforms.

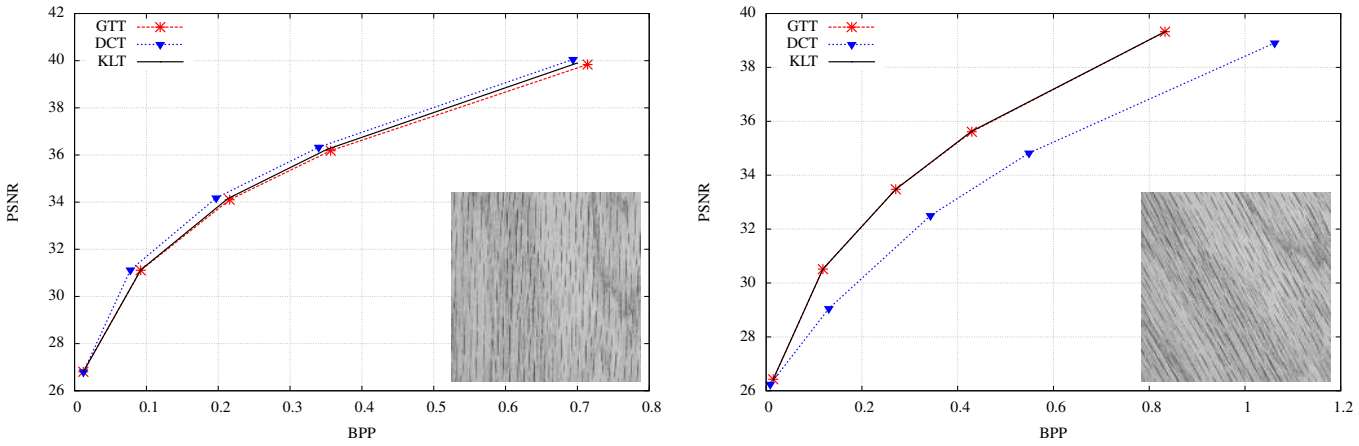


Fig. 3: PSNR vs. BPP results for *wood* image (left) with no rotation and (right) with 30 degree counter-clockwise rotation.

block sizes, (i.e., 4×4 and 8×8), so our solutions scale nicely for larger blocks unlike KLT-based solutions.

- In terms of the optimization procedures, generally GML method is better than SLI method.
- SLI method does not work well on Laplacian (L) matrices due to convergence issues during the graph optimization, since L matrices are singular by definition. Therefore, corresponding results are omitted (see N/A in Table I).

Moreover, Fig.3 illustrates the RD performance results for the *wood* texture image. Note that for the image with no rotation, DCT performs better than KLT and GTT, since the image is mostly smooth and the pixel value discontinuities appear vertically. This is a very special case where KLT and GTT cannot produce effective transforms. On the other hand, KLT and GTT significantly outperform DCT for 30 degree rotated version of the same image where the discontinuities

appear diagonally. Note also that the performance of GTT closely approaches to KLT.

V. Conclusions

In this paper, we have proposed a new class of transforms, called graph template transforms (GTT), to approximate the empirical KLT with a reduced number of parameters. By introducing the notion of graph templates, we add prior knowledge about signal classes. In terms of graph learning, we propose and compare (i) three types of matrices to characterize the graph, (ii) three graph templates to constraint connectivity of graphs, and (iii) two optimization methods to optimize edge weights of a graph based on a given template. Our experimental results lead us to the following conclusions:

- We propose to use Gaussian maximum-likelihood (GML) method since it provides a better optimization for learning edge weights of a graph compared to sparse left inverse (SLI) method in general.
- For sparse graph templates (e.g., 4 or 8 connected templates), we suggest to optimize generalized Laplacian (GL) matrices using GML method.
- For dense graph designs, using 24 connected or more dense graph templates, we propose to optimize sparse inverse covariance (SIC) matrices using GML method.
- As we choose more dense graph templates, sparse inverse covariance (SIC) matrix can closely achieve the rate-distortion performance of KLT. However, any transform design procedure should also consider the trade-off between coding gain and transform signaling/storing overhead.

VI. Acknowledgement

This work has been supported in part by LG Electronics, Inc. and by an USC Annenberg Graduate Fellowship.

References

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

[2] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1649–1668, 2012.

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

[4] C. Hu, L. Cheng, J. Sepulcre, G. El Fakhri, Y. M. Lu, and Q. Li, "A graph theoretical regression model for brain connectivity learning of alzheimer's disease," in *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*. IEEE, 2013, pp. 616–619.

[5] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning graphs from observations under signal smoothness prior." [Online]. Available: http://arxiv.org/abs/1406.7842

[6] 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," *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 83–98, 2013.

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

[8] G. Shen, W.-S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *Picture Coding Symposium (PCS), 2010*. IEEE, 2010, pp. 566–569.

[9] E. Martinez-Enriquez, E. Diaz-de Maria, and A. Ortega, "Video encoder based on lifting transforms on graphs," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, Sept 2011, pp. 3509–3512.

[10] A. Sandryhaila and J. Moura, "Nearest-neighbor image model," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*, Sept 2012, pp. 2521–2524.

[11] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *Journal of Applied Statistics*, vol. 16, no. 2, pp. 131–164, 1989.

[12] C. Zhang and D. Florêncio, "Analyzing the optimality of predictive transform coding using graph-based models," *Signal Processing Letters, IEEE*, vol. 20, no. 1, pp. 106–109, 2013.

[13] A. Said and W. A. Pearlman, "Low-complexity waveform coding via alphabet and sample-set partitioning," in *SPIE Visual Communications and Image Processing*, 1997, pp. 25–37.