GRAPH-BASED LIFTING TRANSFORM FOR INTRA-PREDICTED VIDEO CODING

Yung-Hsuan Chao, Antonio Ortega *

University of Southern California
Los Angeles, CA, USA

Sehoon Yea

LG Electronics Inc.
Seoul, Korea

ABSTRACT

In this paper, we propose a graph-based lifting transform for intra-predicted video sequences. The transform can approximate the performance of a Graph Fourier Transform (GFT) for a given graph, but does not require computing eigenvectors. A predict-update bipartition is designed based on a Gaussian Markov Random Field (GMRF) model with the goal to minimize the energy in the prediction set. Additionally, a novel re-connection method is applied for multi-level graphs, leading to significant gain for the proposed bipartition method and for the conventional MaxCut based bipartition. Experiments on intra-predicted video sequences show that the proposed method, even considering the extra overhead for edge information, outperforms the Discrete Cosine Transform (DCT) and approximates the performance of the higher complexity GFT.

Index Terms— Graph, Lifting transform, Intra-prediction

1. INTRODUCTION

The Graph Fourier transform (GFT) is been recently proposed for image/video processing, including compression [11, 12, 7, 16] and denoising [11, 6, 8]. The advantage of using GFT is its ability to adapt to the characteristics of the signals, for which the position of discontinuities have been signaled and are represented by a graph. In [17], the use of GFT for block-based processing is first introduced for depth map coding, where the high frequency edge structures can be easily represented with low bitrate. In the work by Hu *et al.*[7], edges with small pixel gradient are represented with an additional optimized weight $w \in (0, 1)$. A major challenge in using the GFT is that it requires an eigen-decomposition, which can be complex. Also, the computation of the transform requires $O(N^2)$ operations. The first problem is partly addressed in [4], where a few graphs are chosen as "templates", and their corresponding eigenvectors are precomputed and stored. However, this method is applicable only for small block size, e.g. 4×4 and 8×8, where the number of different templates to consider is relatively small.

To deal with the complexity associated to the GFT, we apply a graph-based lifting transform [13], which can be applied on irregular graphs. By using localized filtering, the complexity can be much lower than GFT. Lifting on graphs has been applied to image denoising and video compression [13, 10, 9], but these were global transforms, applied to the whole image (or video frame). In [1], lifting was applied for *block-based* coding of depth maps, with results that outperformed DCT-based coding and were comparable to using GFT.

The simple lifting-based transform design in [1] performs well for piecewise constant images, but does not provide advantages over the DCT for more general signals such as natural images or intra-prediction residuals. In this paper, we improve the block-based lifting transform of [1] and evaluate its performance on intra-predicted residual video sequences. We first define a model for intra-predicted residues using GMRF, and design a bipartition approach to minimize the energy in the prediction set. Second, we propose a novel re-connection technique to improve the prediction in localized filterbanks. The performance of lifting using the conventional MaxCut based bipartition [10] can also be enhanced with the re-connection. The results of the proposed method on test sequences outperform DCT and are comparable to those obtained with GFT, but without high complexity computation such as eigen-decomposition.

The rest of the paper is organized as follows. In Section 2, we review some preliminaries for graphs and the lifting transform, and discuss the bipartition algorithm based on GMRF model. Section 3 describes the filterbank design with graph re-connection based on the proposed bipartition. In Section 4, we compare the complexity of our approach and that of the GFT. The experiments and conclusion are presented in Sections 5 and 6.

2. UPDATE-PREDICT BIPARTITION IN LIFTING TRANSFORM

2.1. Preliminaries

A weighted graph $G = (V, E)$ consists of a set of nodes $v \in V$, and edges $e_{i,j} \in E$, which capture the similarities between two connected nodes i and j. The similarity is measured using a weight $w_{i,j} \in [0, 1]$. The information can be represented with an adjacency matrix $\mathbf{A}$ where element $\mathbf{A}(i, j) = w_{i,j}$. The combinatorial Laplacian matrix is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is a diagonal matrix with element $\mathbf{D}(i, i) = \sum_k w_{i,k}$.

The lifting transform is a multi-level filterbank that guarantees invertibility. At each level m, nodes are first divided into two disjoint sets, a prediction set (S_P^m) and an update set (S_U^m). The values in S_U^m are used to predict the values in S_P^m. The resulting prediction errors are stored in S_P^m, and are then used to update the values in S_U^m. The smoothed signal in S_U^m will serve as the input signal to level $m + 1$. The computation for coefficients in S_P^m uses only the information in S_U^m, and vice versa. Carrying on the process iteratively produces a multiresolution decomposition.

For video/image compression applications, the coefficients in the update set of the highest level q, along with the prediction set information in levels $\{q - 1, q - 2, \cdots, 1\}$ will be quantized and entropy coded. Therefore, in order to enhance the coding efficiency, the bipartition should be designed so that the energy in the prediction set is minimized. The most commonly applied method is based on computing a maximum cut (MaxCut) [10, 9, 14], where the total link weight between S_P and S_U is maximized. Since each weight measures the similarity between the connected nodes, by doing MaxCut, the similarity between two sets is maximized. In [14], MaxCut is proved to be optimal in minimizing a lower bound of the l_1 norm

*The work is funded in part by LG Electronics Inc.

prediction error using linear filters if entries in the signal are identically distributed. However, the model fails to consider the correlation among signal entries. In this paper, we will show that by exploiting the relations between signal elements modeled as a Gaussian Markov Random Field (GMRF), the coding gain can be significantly improved.

2.2. Multi-level bipartition based on GMRF model

The proposed transform is implemented independently for $m \times m$ blocks in the intra-predicted residual video. Denote $\mathbf{f} \in \mathbb{R}^{m^2}$ the columnized block. Each block is then modeled as a realization of a GMRF, where $\mathbf{f} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma} \in \mathbb{R}^{m^2 \times m^2})$. In our work, we assume zero mean for each block. $\boldsymbol{\Sigma}$ is the covariance matrix, and its inverse $\mathbf{Q} = \Sigma^{-1}$, called precision matrix, measures the partial correlation among samples. Samples i and j are conditionally independent if $\mathbf{Q}(i,j) = 0$. Instead of estimating elements in $\mathbf{Q}$ using actual signal statistics, we define $\mathbf{Q}$ as a function of the graph Laplacian as $\mathbf{Q} = \sigma\tilde{\mathbf{L}}$, where $\tilde{\mathbf{L}}$ is defined as $\mathbf{L} + \delta\mathbf{I}$. A small self-loop weight δ is introduced here to ensure the matrix invertibility so that the probabilistic interpretation holds. The graph Laplacian has been utilized widely in the literature as a prior for image blocks [17, 7, 1]. In our work, the graph for each block is constructed using the information from the edge map. Each block is represented first as a 4-connected grid graph. Image edges are detected based on pixel gradient. Given a threshold value T, for two nodes i and j with pixel difference greater than T, the link is assigned a weight $c \in (0, 1)$. Two blocks with different image edge structure will have different graph Laplacian and GMRF model.

The bipartition is done such that the expected l_2 norm error in the prediction set based on MAP estimate from the update set is minimized. A similar idea has been proposed for sampling and to define a predictive transform [5, 21]. Define $\mathbf{f}_P$ as the vector containing entries corresponding to the prediction in $\mathbf{f}$, and $\mathbf{f}_U$ as the vector for the pixels in the update set, the MAP estimate of $\mathbf{f}_P$ given $\mathbf{f}_U$ is computed as

$$\begin{aligned} \boldsymbol{\mu}_{P|U} &= \boldsymbol{\Sigma}_{PU}\boldsymbol{\Sigma}_{UU}^{\dagger}\mathbf{f}_U \\ &= \tilde{\mathbf{L}}_{PP}^{\dagger}\tilde{\mathbf{L}}_{PU}\mathbf{f}_U, \end{aligned} \tag{1}$$

where $\tilde{\mathbf{L}}^{\dagger}$ is the pseudo-inverse of $\tilde{\mathbf{L}}$. This is equivalent to the MMSE estimate in GMRF. The bipartition problem can then be formulated as a sampling problem, where we look for the update set that minimizes the MAP estimate error in the prediction set. The objective function can be written as:

$$\begin{aligned} \mathbf{S}_U^* &= \arg\min_{\mathbf{S}_U} E_{MAP}(\mathbf{S}_U, \mathbf{S}_P) \\ &= \arg\min_{\mathbf{S}_U} \mathbb{E}[\|\mathbf{f}_P - \boldsymbol{\mu}_{P|U}\|^2] \\ &= \arg\min_{\mathbf{S}_U} tr(\mathbb{E}[(\mathbf{f}_P - \boldsymbol{\mu}_{P|U})(\mathbf{f}_P - \boldsymbol{\mu}_{P|U})^t]) \\ &= \arg\min_{\mathbf{S}_U} tr(\boldsymbol{\Sigma}_{PP} - \boldsymbol{\Sigma}_{PU}\boldsymbol{\Sigma}_{UU}^{\dagger}\boldsymbol{\Sigma}_{UP}) \\ &= \arg\min_{\mathbf{S}_U} tr(\tilde{\mathbf{L}}_{PP}^{\dagger}). \end{aligned} \tag{2}$$

In our experiments, we select half the nodes at each lifting level to form the update set. If q as the number of levels, the update set S_U^m for level m, where $m = \{1, 2, \cdots, q\}$, is computed using the sampling method shown in Algorithm 1.

Note that we use a greedy approach to solve (2), since the original problem is NP hard. In each iteration, one node v_i is selected and added to S_U.

Algorithm 1 Bipartition in multi-level lifting transform

Input graph $G = (V, E)$; $S_U = \emptyset$; $S_P = V$; $\{S_U^m = \emptyset\}_{m=1:q}$

for $m = q : -1 : 1$ **do**
  compute the update set size $|S_U^m| = |V|/2^m$
  for $s = 1 : 1 : |S_U^m|$ **do**
    Select $v_i* = \arg\min_{v_i} E_{MAP}(S_U \cup \{v_i\}, S_P/\{v_i\})$
    $S_P = S_P/\{v_i*\}$
    $S_U = S_U \cup \{v_i*\}$
  end for
  $S_U^m = S_U$
end for

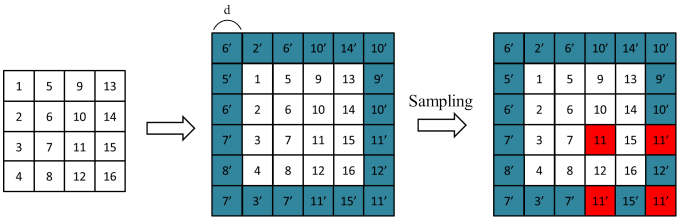


Fig. 1: Boundary extension for pixels around block boundaries.

2.3. Boundary and edge extension

In GMRF, the diagonal element (i, i) in the precision matrix can be interpreted as the inverse of the prediction error for node i given $V/\{i\}$. Hence, the nodes around block boundaries and image edges, which have lower degree, are considered to have larger prediction error, and therefore are given higher priority in sampling. However, the node near boundaries tend to be further away from other nodes, making prediction using these nodes less efficient. To address this issue, we equalize the number of links for each node by using a symmetric boundary extension as shown in Fig. 1. As a result, the graph used for sampling is augmented. The approach is consistent with the filterbank used later, which also uses a boundary extension with degree normalization. An example is shown in Fig. 1. If node v (node 11 in the example) is selected as a sample, its mirrored nodes (denoted $11'$) are also selected. Note that the weight between extended node v' and a boundary node x (e.g., node 15) is equivalent to the weight between v and x. The same idea is also applied for nodes around image edges, see Algorithm 2.

Algorithm 2 Boundary/Edge extension for sampling on graphs

Input $S_U = \emptyset$, $S_P = V$

Output S_U and S_P after sampling

1: Extend the Adjacency matrix $\mathbf{A}$ to include the extended nodes (called $\mathbf{A}_{ext}$) around boundaries and edges.
2: Compute the Laplacian matrix as $\mathbf{D}_{ext} - \mathbf{A}_{ext} + \delta\mathbf{I}$. of data in former batches;
3: Choose the sample y s.t. y, along with its extended nodes $\{y', y'' \cdots\}$ and previously sampled set, minimize the MAP error in set $V/\{y\}$.
4: $S = S \cup \{y\}$, and $S_c = S_c/\{y\}$.
5: Iterate steps 3-4 until the specified sample size is reached.

3. FILTERBANK DESIGN IN MULTI-LEVEL LIFTING TRANSFORM

For filterbank design, we adopt the CDF53 based approach proposed in [18]. Define the input signal (level 0) as $\mathbf{s}^0$, In level m, the nodes

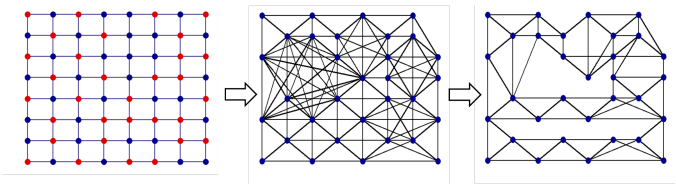
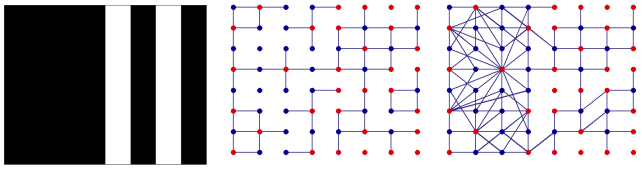


Fig. 2: Graph construction in higher level decomposition via kron reduction. $k = 4$ for sparsification.



(a) (b) (c)

Fig. 3: (a) A toy example consists of smooth (left half) and high variance (right half) areas. (b) The bipartite graph used for prediction/update operators with 1-hop links. (c) The bipartite graph used for prediction/update operators with links derived from Kron reduction ($k = 4$). **The nodes in the update set are marked as red.**

are first divided into (S_P^m, S_U^m), and the coefficients are computed as

$$\begin{aligned} d^m_{i\in S_P^m} &= s^{m-1}_{i\in S_P^m} - \sum_{k\in S_U^m} p^m(i,k)\cdot s_k^{m-1} \\ s^m_{j\in S_U^m} &= s^{m-1}_{j\in S_U^m} + \sum_{r\in S_P^m} u^m(j,r)\cdot d_r^m, \end{aligned} \tag{3}$$

where the s_j^m are the smooth coefficients stored in the update set in level m, and d_i^m are the detail coefficients stored in the prediction set in level m. The smooth coefficients in level $m-1$ are taken as the input signal in level m. The prediction and update operators are functions of the adjacency matrix $\mathbf{A}^m$ of the graph at level m.

3.1. Graph construction in higher level decomposition

After the process in the 1^{st} level lifting transform, only the smoothed values in the update set are utilized in the next level (e.g. the red nodes in Fig. 2). Since the number of nodes is reduced and the average distance between them is longer, a connected graph for filtering cannot be built solely with the $1-$hop links used in level 1. Therefore, graph links providing more global correlation among graph nodes are required. To achieve this, for graph construction at level $m > 1$, we apply Kron reduction [19] using the graph from level $m-1$. The Kron reduction computes the graph Laplacian for a subgraph S as $\boldsymbol{\mathcal{L}}(V,S) = \mathbf{L}_{S,S} - \mathbf{L}_{S,S_c}\mathbf{L}^{\dagger}_{S_c,S_c}\mathbf{L}_{S_c,S}$. However, the graph produced with Kron reduction is usually dense. In fact, without further sparsification, for small blocks like 8×8, the graph becomes fully connected after level 3. Thus, to achieve a good compromise between localization and graph connectivity, we apply a simple sparsification by keeping only the k nearest neighbors for each node. In Fig. 2, we show an example of graph construction for a 8×8 block taken from the intra-predicted residual sequences.

3.2. Graph re-connection for prediction

In the lifting transform at each level, the coefficients in the prediction set are computed based only on information in the update set, and vice versa. Therefore, the graph utilized for transform is in fact

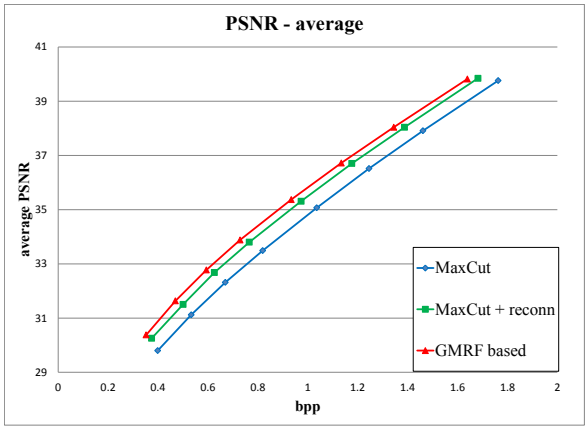


Fig. 4: The comparison between proposed lifting scheme, MaxCut based lifting, and the MaxCut based lifting with proposed reconnection technique.

a bipartite graph with only the links connecting nodes in the opposite sets. In our work, the proposed bipartition scheme is based on minimizing the prediction error of MAP estimate. As a result, the update set nodes in high variance region, e.g. areas with lots of image edges, are densely distributed compared to the distribution in smoothed area. Intuitively, in smooth regions pixels have similar values even they are several hops away, thus selecting a large number of update samples in these areas is redundant. In [10, 9], the bipartite graph used for update and prediction operators consists of only the 1-hop links in the original graph. In consequence, for areas where the sampled nodes for update set are sparsely distributed, the nodes in the prediction set may few or no links connecting the update set, resulting in higher prediction error. An example is shown in Fig.3(b) for the toy example in Fig.3(a).

Based on the density variation, the links in the bipartite graph for transform need to be adjusted accordingly. Again, we use the Kron reduction to reconnect the bipartite graph. The algorithm consists of two parts: First, each node v in the prediction set S_P is connected to the update set S_U by calculating the subgraph Laplacian $\boldsymbol{\mathcal{L}}(V, S_U \cup \{v\})$, where only the links between v and S_U are kept. Later, a sparsification that keeps only k nearest neighbors for each S_P node is applied. The CDF53 filterbank previously described is implemented on the sparsified bipartite graph (Fig.3(c)).

We compare the coding gain of the proposed bipartition and reconnection approach with the MaxCut based lifting in [10]. The test sequences consist of 7 frames from 7 video sequences. The result is shown in Fig. 4. The result also include the result of MaxCut based bipartition with re-connection using Kron reduction. Note that even with a simple bipartition scheme like the greedy approximated MaxCut, with better designed re-connection, the performance is comparable to the one with GMRF model, making further simplification of sampling a direction for the future work.

4. COMPLEXITY ANALYSIS

In this section, we discuss the complexity of the graph-based lifting transform and GFT, for a graph G of size N. The GFT operations for each graph include i) finding the eigenvector decomposition and ii) applying the transform on the signal. Eigenvector decomposition is a process of high complexity that takes $O(N^3)$ even when the Laplacian matrix is sparse. The transform needs matrix-vector multiplication which takes $O(N^2)$, due to the fact that the matrix is dense in general.

Methods	GFT		Lifting with GMRF sampling		Lifting (Max Cut w/ re-connection)		Lifting (Max Cut based)	
	Δ PSNR (dB)	**Δ rate (%)**	**Δ PSNR (dB)**	**Δ rate (%)**	**Δ PSNR (dB)**	**Δ rate (%)**	**Δ PSNR (dB)**	**Δ rate (%)**
Foreman	0.34	-7.28	0.29	-6.42	0.26	-5.77	0.17	-3.63
Mobile	0.17	-1.46	0.10	-0.97	0.10	-0.96	0.08	-0.51
Silent	0.22	-4.28	0.20	-3.88	0.18	-3.58	0.09	-1.66
Deadline	0.37	-4.97	0.31	-3.98	0.30	-3.90	0.24	-3.19

Table 1: PSNR-bitrate comparison with bjontegaard metric. The negative value for Δrate indicates the average bitrate reduction against DCT, and the positive Δ PSNR shows the average PSNR gain.

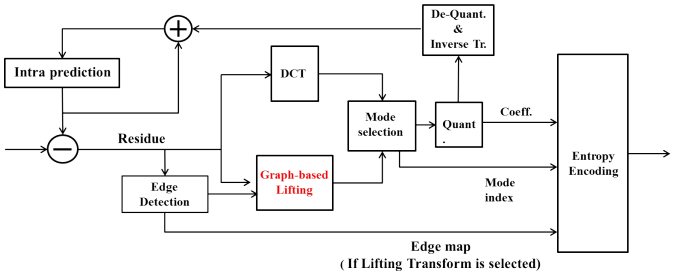


Fig. 5: Encoder for intra-predicted videos with mode selection between DCT and the graph based lifting transform.

Next, we consider the complexity of lifting transform based on the greedy MaxCut bipartition and graph re-connection, which provides performance comparable to GMRF based bipartition. Creating the lifting basis consists of 3 parts: Bipartition with greedy approximated MaxCut [9], re-connection with Kron reduction, and generation of the graph in the next higher level. The greedy MaxCut selects the node with the maximum gain, corresponding initially to the degree. The initialization requires sorting, which takes $O(N \log N)$ if a better data structure like max-heap is used. In each iteration, after a node with the maximum gain is selected, the gain of its neighboring node will be adjusted. The max-heap for gain values will be adjusted accordingly, which take $O(\log N)$ since the number of neighbors is constant for sparse graphs.

For Kron reduction, even though the calculation of $\boldsymbol{\mathcal{L}}(V, S)$ requires taking the matrix inverse, [3] shows that the computation can be implemented iteratively, where nodes in V/S are eliminated one by one, and the complexity depends on the number of links connected to the eliminated node. The paper also shows that the graph maintains sparsity pattern within Kron iteration. Define c to be the maximum number of links for the eliminated node during Kron iteration. The cost for eliminating one node thus takes $O(c^2)$, and the total complexity for Kron reduction is $O(c^2 N)$. The re-connection needs to be done for all the nodes in the prediction set. However, during each Kron reduction, only the connection for the specified prediction node to the update set is considered, so the cost for eliminating one node is $O(c)$. Therefore, the overall complexity is $O(cN^2)$ for the re-connection. The number of lifting levels needed is $\log N$, since in each level, half of the nodes are taken in the update set. In the end, the overall complexity to define the lifting basis is $O(cN^2 \log N)$. Note that the graph size in the higher lifting levels is smaller. Hence, we can expect much lower complexity in the real implementation.

The lifting transform has complexity $O(N)$ at each level, since the operation for each node requires only its neighbors, which is constant for a sparse graph. Therefore, the total complexity of transform application is $O(N \log N)$. In a real application, if the graphs can be pre-computed as templates as in [4], the only computation needed is the transform part, where the graph lifting provides significant complexity reduction as compared to the GFT. In our implementation, graph based lifting transforms were over 50% faster than GFTs.

5. EXPERIMENTAL RESULTS

In the experiment, we generate the intra-predicted residual frames for test sequences *Foreman*, *Mobile*, *Silent*, and *Deadline* using HEVC(HM-14) with transform unit size fixed as 8×8. The encoder system is shown in Fig. 5. The mode selection is done by choosing the transform with minimum RD cost defined as SSE + $\lambda \cdot$ bitrate, where $\lambda = 0.85 \cdot 2^{(QP-12)/5}$. The *QP* values used in this experiment are from 24 to 36 with step size 2. For blocks encoded with graph based lifting, the image edge overhead is coded with the arithmetic edge coding (AEC) [2], first connecting adjacent edge components as contours, then encoding each contour with context-based method. In order to further reduce the overhead cost, in each 8×8 block only one contour is allowed. Beside, the encoder needs to send a flag indicating the transform chosen. Both DCT and lifting transformed coefficients are uniformly quantized and encoded with an amplitude group partition technique called AGP [15]. AGP can learn and adapt to different coefficient distribution, thus provides fair comparison between different transforms. Before quantization, the coefficients of CDF53 lifting transform are first normalized based on [20] so as to compensate for the lack of orthogonality. The DCT coefficients are zig-zag scanned, and the coefficients for lifting are ordered from lowest frequency (update set in the highest level) to highest frequency (the prediction set in the 1^{st} level). Within each subband, coefficients are ordered from low reliability to high reliability [18].

The average PSNR gain and bitrate reduction are presented in Table 1. For videos with simple edge structures like *Foreman* and *Deadline*, graph based lifting has around $0.3dB$ gain in PSNR. While in the videos with complicated edge structure like *Mobile*, the gain is limited since the edge map dominates the cost. We also show that MaxCut based bipartition with graph re-connection provides a good approximation to both the higher complexity GMRF based bipartiton and GFT.

6. CONCLUSION

In this paper, we propose a transform for intra-predicted video coding based on the graph based lifting transform. The bipartition algorithm is justified based on GMRF model, and provides better coding gain than MaxCut based bipartition. We also show that with better graph reconnection, even with simple MaxCut bipartition, we can outperform DCT and approximate the higher complexity GMRF based approach and GFT. These promising results motivate us to consider other settings, such as video coding with larger blocksize, in future work.

7. REFERENCES

[1] Yung-Hsuan Chao, Antonio Ortega, Wei Hu, and Gene Cheung. Edge-adaptive depth map coding with lifting transform on graphs. In *Picture Coding Symposium (PCS), 2015*, pages 60–64. IEEE, 2015.

[2] Ismael Daribo, Gene Cheung, and Dinei Florencio. Arithmetic edge coding for arbitrarily shaped sub-block motion prediction in depth video compression. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 1541–1544. IEEE, 2012.

[3] Florian Dorfler and Francesco Bullo. Kron reduction of graphs with applications to electrical networks. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 60(1):150–163, 2013.

[4] Hilmi E Egilmez, Amir Said, Yung-Hsuan Chao, and Antonio Ortega. Graph-based transforms for inter predicted video coding. 2015.

[5] Akshay Gadde and Antonio Ortega. A probabilistic interpretation of sampling theory of graph signals. *arXiv preprint arXiv:1503.06629*, 2015.

[6] Wei Hu, Gene Cheung, Xin Li, and Oscar C Au. Graph-based joint denoising and super-resolution of generalized piecewise smooth images. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 2056–2060. IEEE, 2014.

[7] Wei Hu, Gene Cheung, Antonio Ortega, and Oscar C Au. Multiresolution graph fourier transform for compression of piecewise smooth images. *Image Processing, IEEE Transactions on*, 24(1):419–433, 2015.

[8] Wei Hu, Xin Li, Gene Cheung, and Oscar Au. Depth map denoising using graph-based transform and group sparsity. In *Multimedia Signal Processing (MMSP), 2013 IEEE 15th International Workshop on*, pages 001–006. IEEE, 2013.

[9] Eduardo Martínez-Enríquez, Fernando Díaz-de María, and Antonio Ortega. Video encoder based on lifting transforms on graphs. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 3509–3512. IEEE, 2011.

[10] Eduardo Martínez-Enríquez and Antonio Ortega. Lifting transforms on graphs for video coding. In *Data Compression Conference (DCC), 2011*, pages 73–82. IEEE, 2011.

[11] Sunil K Narang, Yung Hsuan Chao, and Antonio Ortega. Graph-wavelet filterbanks for edge-aware image processing. In *Statistical Signal Processing Workshop (SSP), 2012 IEEE*, pages 141–144. IEEE, 2012.

[12] Sunil K Narang, Yung-Hsuan Chao, and Antonio Ortega. Critically sampled graph-based wavelet transforms for image coding. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*, pages 1–4. IEEE, 2013.

[13] Sunil K Narang and Antonio Ortega. Lifting based wavelet transforms on graphs. In *Proceedings: APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference*, pages 441–444. Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, International Organizing Committee, 2009.

[14] Ha Q Nguyen and Minh N Do. Downsampling of signals on graphs via maximum spanning trees. *Signal Processing, IEEE Transactions on*, 63(1):182–191, 2015.

[15] Amir Said and William A Pearlman. Low-complexity waveform coding via alphabet and sample-set partitioning. In *Electronic Imaging'97*, pages 25–37. International Society for Optics and Photonics, 1997.

[16] Alfonso Sánchez, Godwin Shen, and Antonio Ortega. Edge-preserving depth-map coding using graph-based wavelets. In *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, pages 578–582. IEEE, 2009.

[17] Godwin Shen, Woo-Shik Kim, Sunil K Narang, Antonio Ortega, Jaejoon Lee, and Hocheon Wey. Edge-adaptive transforms for efficient depth map coding. In *Picture Coding Symposium (PCS), 2010*, pages 566–569. IEEE, 2010.

[18] Godwin Shen and Antonio Ortega. Tree-based wavelets for image coding: Orthogonalization and tree selection. In *Picture Coding Symposium, 2009. PCS 2009*, pages 1–4. IEEE, 2009.

[19] David I Shuman, Mohammad Javad Faraji, and Pierre Vandergheynst. A multiscale pyramid transform for graph signals. *arXiv preprint arXiv:1308.4942*, 2013.

[20] Bryan Usevitch. Optimal bit allocation for biorthogonal wavelet coding. In *Data Compression Conference, 1996. DCC'96. Proceedings*, pages 387–395. IEEE, 1996.

[21] Cha Zhang and Dinei Florêncio. Analyzing the optimality of predictive transform coding using graph-based models. *Signal Processing Letters, IEEE*, 20(1):106–109, 2013.