Graph-based Transform Coding with Application to Image Compression

Giulia Fracastoro, Dorina Thanou, Pascal Frossard

December 29, 2017

Abstract

In this paper, we propose a new graph-based coding framework and illustrate its application to image compression. Our approach relies on the careful design of a graph that optimizes the overall rate-distortion performance through an effective graph-based transform. We introduce a novel graph estimation algorithm, which uncovers the connectivities between the graph signal values by taking into consideration the coding of both the signal and the graph topology in rate-distortion terms. In particular, we introduce a novel coding solution for the graph by treating the edge weights as another graph signal that lies on the dual graph. Then, the cost of the graph description is introduced in the optimization problem by minimizing the sparsity of the coefficients of its graph Fourier transform (GFT) on the dual graph. In this way, we obtain a convex optimization problem whose solution defines an efficient transform coding strategy. The proposed technique is a general framework that can be applied to different types of signals, and we show two possible application fields, namely natural image coding and piecewise smooth image coding. The experimental results show that the proposed method outperforms classical fixed transforms such as DCT, and, in the case of depth map coding, the obtained results are even comparable to the state-of-the-art graph-based coding method, that are specifically designed for depth map images.

1 Introduction

In the last years, the new field of signal processing on graphs has gained increasing attention [1]. Differently from classical signal processing, this new emerging field considers signals that lie on irregular domains, where the signal values are defined on the nodes of a weighted graph and the edge weights reflect the pairwise relationship between these nodes. Particular attention has been given to the design of flexible graph signal representations, opening the door to new structure-aware transform coding techniques, and eventually to more efficient signal and image compression frameworks. As an illustrative example, an image can be represented by a graph, where the nodes are the image pixels and the edge weights capture the similarity between adjacent pixels. Such a flexible representation permits to go beyond traditional transform coding by moving from classical fixed transforms such as the discrete cosine transform (DCT) [2] to graph-based transforms that are better adapted to the actual signal structure, such as the graph Fourier transform (GFT) [3]. Hence, it is possible to obtain a more compact representation of an image, as the energy of the image signal is concentrated in the lowest frequencies. This provides a strong advantage compared to the classical DCT transform especially when the image contains arbitrarily shaped discontinuities. In this case, the DCT transform coefficients are not necessarily sparse and contain many high frequency coefficients with high energy. The GFT, on the other hand, may lead to sparse representations and eventually more efficient compression.

However, one of the biggest challenges in graph-based signal compression remains the design of the graph and the corresponding transform. A good graph for effective transform coding should lead to easily compressible signal coefficients, at the cost of a small overhead for coding the graph. Most graph-based coding techniques focus mainly on images, and they construct the graph by considering pairwise similarities among pixel intensities [4,5] or using a lookup table that stores the most popular GFTs [6]. It has been shown that these methods could provide a significant gain in the coding of piecewise smooth images. Instead, in the case of natural images, the cost required to describe the graph often outweighs the coding gain provided by the adaptive graph transform, and often leads to unsatisfactory results. The problem of designing a graph transform stays critical and may actually represent the major obstacle towards effective compression of signal that live on an irregular domain.

In this work, we build on our previous work [7], and introduce a new graph-based signal compression scheme and apply it to image coding. First, we propose a novel graph-based compression framework that takes into account the coding of the signal values as well as the cost of transmitting the graph. Second, we introduce an innovative way for coding the graph by treating its edge weights as a graph signal that lies on the dual graph. We then compute the graph Fourier transform of this signal and code its quantized transform coefficients. The choice of the graph is thus posed as a rate-distortion optimization problem. The cost of coding the signal is captured by minimizing the smoothness of the graph signal on the adapted graph. The transmission cost of the graph itself is controlled by penalizing the sparsity of the graph Fourier coefficients of the edge weight signal that lies on the dual graph. The solution of our optimization problem is a graph that provides an effective tradeoff between the sparsity of the signal transform and the graph coding cost.

We apply our method to two different types of signals, namely natural images and piecewise smooth images. Experimental results on natural images confirm that the proposed algorithm can efficiently infer meaningful graph topologies, which eventually lead to improved coding results compared to non-adaptive methods based on classical transforms such as the DCT. Moreover, we show that our method can significantly improve the classical DCT on piecewise smooth images, and it even leads to comparable results to the state-of-the-art graph-based depth image coding solutions. However, in contrary to these dedicated algorithms, it is important to underline that our framework is quite generic and can be applied to very different types of signals.

The outline of the paper is as follows. We first discuss related work in Section II. We then introduce some preliminary definitions on graphs in Section III. Next, we present the proposed graph construction problem in Section IV. The application of the proposed graph construction algorithm to image coding and the entire compression framework are described in Section V. Then, the experimental results on natural images and piecewise smooth images are presented in Section VI and VII, respectively. Finally we draw some conclusions in Section VIII.

2 Related work

In this section, we first provide a brief overview of transform coding. Then, we focus on graph-based coding and learning methods, that are closely related to the framework proposed in this work.

2.1 Transform coding

Lossy image compression usually employs a 2D transform to produce a new image representation that lies in the transform domain [8]. Usually, the obtained transform coefficients are approximately uncorrelated and most of the information is contained in only a few of them. It is proved that the Karhunen-Loève transform (KLT) can optimally decorrelate a signal that has Gaussian entries [9]. However, since the KLT is based on the eigendecomposition of the covariance matrix, this matrix or the transform itself has to be sent to the receiver. For this reason, the KLT is not practical in most circumstances [8]. The most common transform in image compression is the DCT [2], which employs a fixed set of basis vector. It is known that the DCT is asymptotically equivalent to the KLT for signals that can be modelled as a first-order autoregressive process [10]. Nevertheless, this model fails to capture the complex and non-stationary behavior that is typically present in natural images. In the light of the above, transform design is still an active research field and in the last years many signal adaptive transforms have been presented. In this paper, we focus on a specific type of adaptive transforms, namely graph-based transforms.

2.2 Graph-based image coding

In the last years, graph signal processing has been applied to different image coding applications, especially for piecewise smooth images. In [4,5], the authors propose a graph-based coding method where the graph is defined by considering pairwise similarities among pixel intensities. Another efficient graph construction method for piecewise smooth images has been proposed in [6], where the authors use a lookup table that stores the most popular graphs. Then, for each signal, they perform an exhaustive search choosing the best GFT in rate-distortion terms. Furthermore, a new graph transform, called signed graph Fourier transform, has been presented in [11]. This transform is targeted for compression of depth images and its underlying graph contains negative edges that describe negative correlations between pixels pairs.

Recently, a number of methods using a graph-based approach have also been proposed for transform coding of inter and intra predicted residual blocks in video compression. A novel graph-based method for intra-frame coding has been presented in [12], which introduces a new generalized graph Fourier transform. A graph-based method for inter predicted video coding has been introduced in [13], where the authors design a set of simplified graph templates capturing the basic statistical characteristics of inter predicted residual blocks. Furthermore, a few separable graph-based transforms for residual coding have also been introduced. In [14], for example, the authors propose a new class of graph-based separable transforms for intra and inter predictive video coding. The proposed transform is based on two separate line graphs, where the edge weights are optimized using a graph learning problem. Another graph-based separable transform for inter predictive video coding has been presented in [15]. In this case, the proposed transform, called symmetric line graph transform, has symmetric eigenvectors and therefore it can be efficiently implemented.

Finally, a few graph-based methods have also been presented for natural image compression. In [16], a new technique of graph construction targeted for image compression is proposed. This method employs innovative edge metrics, quantization and edge prediction technique. Moreover, in [17], a new class of transforms called graph template transforms has been introduced for natural image compression, focusing in particular on texture images. Finally, a method for designing sparse graph structures that capture principal gradients in image code blocks is proposed in [18]. However, in all these methods, it is still not clear how to define a graph whose corresponding transform provides an effective tradeoff between the sparsity of the transform coefficients and the graph coding cost.

2.3 Graph construction

Several attempts to learn the structure and in particular a graph from data observations have been recently proposed, but not necessarily from a compression point of view. In [19–21], the authors formulate the graph learning problem as a precision matrix estimation with generalized Laplacian constraints. The same method is also used in [14, 15], where the authors use a graph learning problem in order to find the generalized graph Laplacian that best approximates residual video data. Moreover, in [22, 23], a sparse combinatorial Laplacian matrix is estimated from the data samples under a smoothness prior. Furthermore, in [17], the authors use a graph template to impose on the graph Laplacian a sparsity pattern and approximate the empirical inverse covariance based on that template.

Even if all the methods presented above contain some constraints on the sparsity of the graph, none of them explicitly takes into account the real cost of representing and coding the graph. In addition, most of them do not really target images. Instead, in this paper, we go beyond prior art and we fill this gap by defining a new graph construction problem that takes into account the graph coding cost. Moreover, we show how our generic framework can be used for image compression.

3 Basic definitions on graphs

For any graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ and $\mathcal{E}$ represent respectively the node and edge sets with $|\mathcal{V}| = N$ and $|\mathcal{E}| = M$, we define the weighted adjacency matrix $W \in \mathbb{R}^{N \times N}$ where W_{ij} is the weight associated to the edge (i, j) connecting nodes i and j. For undirected graphs with no self loops, W is symmetric and has null diagonal. The graph Laplacian is defined as $L = D - W$, where D is a diagonal matrix whose i-th diagonal element D_{ii} is the sum of the weights of all the edges incident to node i. Since L is a real symmetric matrix, it is diagonalizable by an orthogonal matrix

$$L = \Psi \Lambda \Psi^T,$$

where $\Psi \in \mathbb{R}^{N \times N}$ is the eigenvector matrix of L that contains the eigenvectors as columns, and $\Lambda \in \mathbb{R}^{N \times N}$ is the diagonal eigenvalue matrix, with eigenvalues sorted in ascending order.

In the next sections, we will use also an alternative definition of the graph Laplacian L that uses the incidence matrix $B \in \mathbb{R}^{N \times M}$ [24], which is defined as follows

$$B_{ie} = \begin{cases} 1, & \text{if } e = (i, j) \\ -1, & \text{if } e = (j, i) \\ 0, & \text{otherwise,} \end{cases}$$

where an orientation is chosen arbitrarily for each edge. Let $\widehat{W} \in \mathbb{R}^{M \times M}$ be a diagonal matrix where $\widehat{W}_{ee} = W_{ij}$ if $e = (i, j)$. Then, we can define the graph Laplacian L as

$$L = B \widehat{W} B^T. \tag{1}$$

It is important to underline that the graph Laplacian obtained using (1) is independent from the edge orientation in $\mathcal{G}$.

3.1 Graph Fourier Transform

A graph signal $x \in \mathbb{R}^N$ in the vertex domain is a real-valued function defined on the nodes of the graph $\mathcal{G}$, such that x_i, $i = 1, \ldots, N$ is the value of the signal at node $i \in \mathcal{V}$ [1]. For example, for an image signal we can consider an associated graph where the nodes of the graph are the pixels of the image. Then, the smoothness of x on $\mathcal{G}$ can be measured using the Laplacian L [25]

$$x^T Lx = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} W_{ij}(x_i - x_j)^2. \tag{2}$$

Eq. (2) shows that a graph signal x is considered to be smooth if strongly connected nodes have similar signal values. This equation also shows the importance of the graph. In fact, with a good graph representation the discontinuities should be penalized by low edge weights, in order to obtain a smooth representation of the signal. Finally, the eigenvectors of the Laplacian are used to define the graph Fourier transform (GFT) [1] of the signal x as follows:

$$\hat{x} = \Psi^T x.$$

The graph signal x can be easily retrieved from $\hat{x}$ by inversion, namely $x = \Psi\hat{x}$. Analogously to the Fourier transform in the Euclidean domain, the GFT is used to describe the graph signal in the Fourier domain.

3.2 Comparison between KLT and GFT

As we have said in Section 2, the KLT is the transform that optimally decorrelates a signal that has Gaussian entries. In this section, we discuss the connection of the graph Fourier transform with the KLT, showing that the GFT can be seen as an approximation of the KLT.

Let us consider a signal $x \in \mathbb{R}^N$ that follows a Gaussian Markov Random Field (GMRF) model with respect to a graph $\mathcal{G}$, with a mean μ and a precision matrix Q. Notice that the GMRF is a very generic model, where the precision matrix can be defined with much freedom, as long as its non-zero entries encode the partial correlations between random variables, and as long as their locations correspond to the edges of the graph. It has been proved that, if the precision matrix Q of the GMRF model corresponds to the Laplacian L, then the KLT of the signal x is equivalent to the GFT [26].

As shown before, the graph Laplacian has a very specific structure where the non-zero components correspond to the edges of the graph, and, for this reason, it is a sparse matrix, since typically $|E| \ll N^2$. Since the precision matrix in general does not have such fixed structure, we now study the KLT of a signal whose model is a GMRF with a generic precision matrix Q. In this case, the GFT does not correspond to the KLT anymore and the GFT should be considered as an approximation of the KLT, where the precision matrix is forced to follow this specific structure. In order to find the GFT that best approximates the KLT, we introduce a maximum likelihood estimation problem, using an approach similar to the one presented in [20]. The density function of a GMRF has the following form [27]

$$p(x) = (2\pi)^{-\frac{N}{2}}(\det Q)^{\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu)^T Q(x-\mu)\right).$$

The log-likelihood function can then be computed as follows

$$\log \mathcal{L}(Q, \mu | x) = \log(\det Q)^{\frac{1}{2}} - \frac{1}{2}(x - \mu)^T Q (x - \mu). \tag{3}$$

Given $x_1, ..., x_n$ observations of the signal x, we find the Laplacian matrix L that best approximates Q by solving the following problem

$$\max_{L \in \Gamma} \log \mathcal{L}(L, \mu | x_1, ..., x_n), \tag{4}$$

where Γ denotes the set of valid Laplacian matrices. Then, by using (3), the problem in (4) can be written as

$$\max_{L \in \Gamma} \log(\det{}^* L)^{\frac{1}{2}} - \frac{1}{2}\mathrm{tr}((X - \mu)^T L (X - \mu)), \tag{5}$$

where X is the matrix whose columns are the N column vectors $x_1, x_2, ..., x_N$ and $\det^*$ is the pseudo-determinant (since L is singular). The optimization problem in (5) defines the graph whose GFT best approximates the KLT. The advantage of using the GFT instead of the KLT is that we force the precision matrix to follow the specific sparse structure defined by the Laplacian. In this way, the transform matrix can be transmitted to the decoder in a more compact way. In the next section, we will highlight the connection between the proposed graph construction problem and the maximum likelihood estimation problem presented in (5).

4 Graph-transform optimization

Graph-based compression methods use a graph representation of the signal through its GFT, in order to obtain a data-adaptive transform that captures the main characteristics of the signals. The GFT coefficients are then encoded, instead of the original signal values themselves. In general, a signal that is smooth on a graph has its energy concentrated in the low frequency coefficients of the GFT, hence it is easily compressible. To obtain good compression performance, the graph should therefore be chosen such that it leads to a smooth representation of the signal. At the same time, it should also be easy to encode, since it has to be transmitted to the decoder for signal reconstruction. Often, the cost of the graph representation outweighs the benefits of using an adaptive transform for signal representation. In order to find a good balance between graph signal representation benefits and coding costs, we introduce a new graph construction approach that takes into consideration the above mentioned criteria.

We first pose the problem of finding the optimal graph as a rate-distortion optimization problem defined as

$$\min_{L \in \mathbb{R}^{N \times N}} \mathcal{D}(L) + \gamma(\mathcal{R}_c(L) + \mathcal{R}_G(L)), \tag{6}$$

where $\mathcal{D}(L)$ is the distortion between the original signal and the reconstructed one and is defined as follows

$$\mathcal{D}(L) = \|u - \tilde{u}(L)\|^2,$$

where u and $\tilde{u}(L)$ are respectively the original and the reconstructed signal via its graph transform on L. The total coding rate is composed of two representation costs, namely the cost of the signal transform coefficients $\mathcal{R}_c(L)$ and the cost of the graph description $\mathcal{R}_G(L)$. Each of these terms depends on the graph characterized by L and on the coding scheme. We describe them in more details in the rest of the section.

4.1 Distortion approximation

The distortion $\mathcal{D}(L)$ is defined as follows

$$\mathcal{D}(L) = \|u - \tilde{u}(L)\|^2 = \|\hat{u}(L) - \hat{u}_q(L)\|^2,$$

where u and $\tilde{u}(L)$ are respectively the original and the reconstructed signal, and $\hat{u}(L)$ and $\hat{u}_q(L)$ are respectively the transform coefficients and the quantized transform coefficients. The equality holds due to the orthonormality of the GFT. Considering a uniform scalar quantizer with the same step size q for all the transform coefficients, if q is small the expected value of the distortion $\mathcal{D}(L)$ can be approximated as follows [28]

$$\mathcal{D} = \frac{q^2 N}{12}.$$

With this high-resolution approximation, the distortion depends only on the quantization step size and it does not depend on the chosen L [6]. For simplicity, in the rest of the paper we adopt this assumption. Therefore, the optimization problem (6) is reduced to finding the graph that permits to minimize the rate terms.

4.2 Rate approximation of the transform coefficients

We can evaluate the cost of the transform coefficients $\mathcal{R}_c(L)$ by evaluating the smoothness of the signal on the graph described by L. We use the approximation proposed in [6], [5], namely

$$\begin{aligned}\mathcal{R}_c(L) = u^T L u &= u^T \left(\sum_{l=0}^{N-1} \lambda_l(L) \psi_l(L) \psi_l(L)^T \right) u \\ &= \sum_{l=0}^{N-1} \lambda_l(L) (u^T \psi_l(L))(\psi_l(L)^T u) = \sum_{l=0}^{N-1} \lambda_l \hat{u}_l^2(L),\end{aligned} \tag{7}$$

where λ_l and ψ_l are respectively the l-th eigenvalue and l-th eigenvector of L. Therefore, $\mathcal{R}_c(L)$ is an eigenvalue-weighted sum of squared transform coefficients. It assumes that the coding rate decreases when the smoothness of the signal over the graph defined by L increases. In addition, (7) relates the measure of the signal smoothness with the sparsity of the transform coefficients. The approximation in (7) does not take into account the coefficients that corresponds to $\lambda_0 = 0$ (i.e., the DC coefficients). Thus, (7) does not capture the variable cost of DC coefficients in cases where the graph contains a variable number of connected components. However, in our work we ignore this cost as we impose that the graph is connected.

It is also interesting to point out that there is a strong connection between (7) and (5). In fact, if we suppose that $\mu = 0$ and if we consider u as the only observation of the signal x, then the second term of the log-likelihood in (5) is equal to $-\mathcal{R}_c(L)$. For this reason, we can say that the solution of our optimization problem can be seen as an approximation of the KLT.

4.3 Rate approximation of the graph description

The graph description cost $\mathcal{R}_G(L)$ depends on the method that is used to code the graph. Generally, a graph could have an arbitrary topology. However, in order to reduce the graph transmission cost, we choose to use a fixed incidence matrix B for the graph and to vary only the edge weights. Therefore, the graph can be defined simply by a vector $w \in \mathbb{R}^M$, where w_e with $1 \leq e \leq M$ is the weight of the edge e. Then, by using (1) we can define the graph Laplacian $L = B^T \text{diag}(w) B$.

In order to compress the edge weight vector w, we propose to treat it as a graph signal that lies on the dual graph $\mathcal{G}_d$. Given a graph $\mathcal{G}$, we define its dual graph $\mathcal{G}_d$ as an unweighted graph where each node of $\mathcal{G}_d$ represents an edge of $\mathcal{G}$ and two nodes of $\mathcal{G}_d$ are connected if and only if their corresponding edges in $\mathcal{G}$ share a common endpoint. An example of a dual graph is shown in Fig. 1. We choose to use this graph representation for the edge weight signal w because consecutive edges $\mathcal{G}$ often have similar weights, since the signals have often smooth regions or smooth transitions between regions. The latter is generally true in case of images. In this way, the dual graph can provide a smooth representation of w. We can define the graph Laplacian matrix $L_d \in \mathbb{R}^{M \times M}$ of the dual graph $\mathcal{G}_d$ and the corresponding eigenvector and eigenvalue matrices $\Psi_d \in \mathbb{R}^{M \times M}$ and $\Lambda_d \in \mathbb{R}^{M \times M}$ such that $L_d = \Psi_d \Lambda_d \Psi_d^T$. We highlight that, since $\mathcal{G}_d$ is an unweighted graph, it is independent of the choice of L, and by consequence also Λ_d and Ψ_d are independent from L.

Since w can be represented as a graph signal, we can compute its GFT $\hat{w} \in \mathbb{R}^M$ as

$$\hat{w} = \Psi_d^T w.$$

Therefore, we can use $\hat{w}$ to describe the graph $\mathcal{G}$ and we evaluate the cost of the graph description by measuring the coding cost of $\hat{w}$. It has been shown that the total bit budget needed to code a vector is proportional to the number of non-zero coefficients [29], thus we approximate the cost of the graph description by measuring the sparsity of $\hat{w}$ as follows

$$\mathcal{R}_G(L) = \|\hat{w}\|_1 = \|\Psi_d^T w\|_1. \tag{8}$$

We highlight that we use two different types of approximations for $\mathcal{R}_c(L)$ and $\mathcal{R}_G(L)$, even if both of them are treated as graph signals. This is due to the

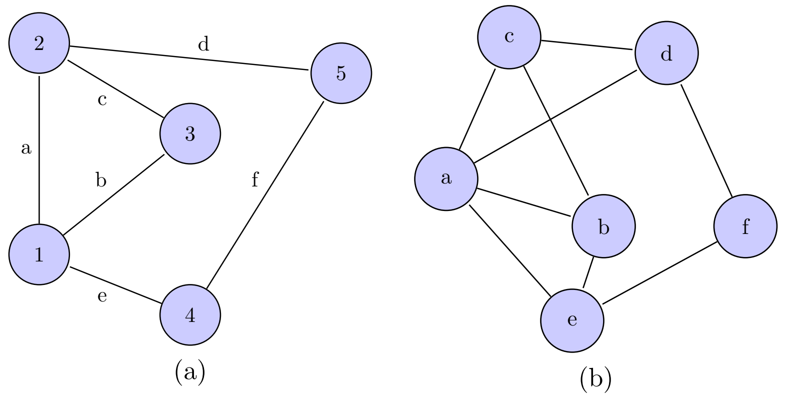


Figure 1: An example of a graph (a) and its corresponding dual graph (b). The edges in the first graph (labeled with lower case letters) become the nodes of the corresponding dual graph.

fact that the two signals have different characteristics. In the case of an image signal u, we impose that the signal is smooth over $\mathcal{G}$, building the graph $\mathcal{G}$ with this purpose. Instead for w, even if we suppose that consecutive edges usually have similar values, we have no guarantees that w is smooth on $\mathcal{G}_d$, since $\mathcal{G}_d$ is fixed and it is not adapted to the image signal. Therefore, in the second case using a sparsity constraint is more appropriate for capturing the characteristics of the edge weight signal w.

To be complete, we finally note that the dual graph has already been used in graph learning problems in the literature. In particular, in [30] the authors propose a method for joint denoising and contrast enhancement of images using the graph Laplacian operator, where the weights of the graph are defined through an optimization problem that involves the dual graph. Moreover, [31] presents a graph-based dequantization method by jointly optimizing the desired graph-signal and the similarity graph, where the weights of the graph are treated as another graph signal defined on the dual graph. The approximation of $\mathcal{R}_G(L)$ presented in (8) may look similar to the one used in [31]. The main difference between the two formulations is that in (8) we minimize the sparsity of w in the GFT domain in order to lossy code the signal w; instead, in [31], the authors minimize the differences between neighboring edges in order to optimize the graph structure without actually coding it.

4.4 Graph construction problem

By using (1), (7) and (8), our graph construction problem (6) is reduced to the following optimization problem

$$\min_{w \in \mathbb{R}^M} u^T B(\mathrm{diag}(w)) B^T u + \alpha \|\Psi_d^T w\|_1, \tag{9}$$

where α is a weighting constant parameter, that allows us to balance the contribution of the two terms.

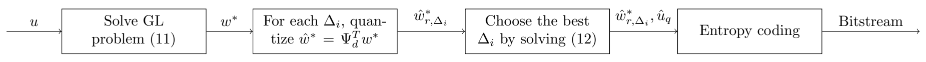

u
Solve GL problem (11)
w^*
For each Δ_i, quantize $\hat{w}^* = \Psi_d^T w^*$
$\hat{w}^*_{r,\Delta_i}$
Choose the best Δ_i by solving (12)
$\hat{w}^*_{r,\Delta_i}, \hat{u}_q$
Entropy coding
Bitstream

Figure 2: Block diagram of the proposed coding method for an input image u.

Building on the rate-distortion formulation of (9), we design the graph by solving the following optimization problem

$$\begin{aligned} \min_{w \in \mathbb{R}^M} \; & u^T B(\text{diag}(w))B^T u + \alpha \|\Psi_d^T w\|_1 - \beta \mathbf{1}^T \log(w), \\ \text{s. t.} \quad & w \leq \mathbf{1}, \end{aligned} \tag{10}$$

where α and β are two positive regularization parameters and $\mathbf{1}$ denotes the constant one vector. The inequality constraint has been added to guarantee that all the weights are in the range $(0, 1]$, which is the same range of the most common normalized weighting functions [32]. Then, the logarithmic term has been added to penalize low weight values and to avoid the trivial solution. In addition, this term guarantees that $w_m > 0$, $\forall m$, so that the graph is always connected. A logarithmic barrier is often employed in graph learning problems [23]. In particular, it has further been shown that a graph with Gaussian weights can be seen as the result of a graph learning problem with a specific logarithmic barrier on the edge weights [23].

The problem in (10) can be cast as a convex optimization problem with a unique minimizer. To solve this problem, we write the first term in the following form

$$\begin{aligned} u^T B(\text{diag}(w))B^T u &= \text{tr}((B^T u u^T B)\text{diag}(w)) \\ &= \text{vec}(B^T u u^T B)^T \text{vec}(\text{diag}(w)) \\ &= \text{vec}(B^T u u^T B)^T M_{\text{diag}} w, \end{aligned}$$

where $\text{tr}(\cdot)$ denotes the trace of a matrix, $\text{vec}(\cdot)$ is the vectorization operator, and $M_{\text{diag}} \in \mathbb{R}^{M^2 \times M}$ is a matrix that converts the vector w into $\text{vec}(\text{diag}(w))$. Then, we can rewrite problem (10) as

$$\begin{aligned} \min_{w \in \mathbb{R}^M} \; & \text{vec}(B^T u u^T B)^T M_{\text{diag}} w + \alpha \|\Psi_d^T w\|_1 - \beta \mathbf{1}^T \log(w), \\ \text{s. t.} \quad & w \leq \mathbf{1}. \end{aligned} \tag{11}$$

The problem in (11) is a convex problem with respect to the variable w and can be solved efficiently via interior-point methods [33].

5 Graph-based image compression

We now describe how the graph construction problem of the previous section can be applied to block-based image compression. It is important to underline that the main goal of this section is to present an application of our framework.

Therefore, we do not present an optimization of the full coding process, but we mainly focus on the transform block.

As pointed out in the previous sections, given an image block u we have two different types of information to transmit to the decoder: the transform coefficients of the image signal $\hat{u}$ and the description of the graph $\hat{w}$. The image coefficients $\hat{u}$ are quantized and then coded using an entropy coder. Under the assumption of high bitrate, the optimal entropy-constrained quantizer is the uniform quantizer [34]. Moreover, it has been proved that, under the assumption that all the transform coefficients follow the same probability distribution, the transform code is optimized when the quantization steps of all coefficients are equal [35]. For these reasons, we quantize the image transform coefficients $\hat{u}$ using an uniform quantizer with the same step size q for all the coefficients. Then, since we assume that the non-zero coefficients are concentrated in the low frequencies, we code the quantized coefficients until the last non-zero coefficient using an adaptive bitplane arithmetic encoder [36] and we transmit the position of the last significant coefficient.

The graph itself is transmitted by its GFT coefficients vector $\hat{w}$, which is quantized and then transmitted to the decoder using an entropy coder. In order to reduce the cost of the graph description, we reduce the number of elements in $\hat{w}$ by taking into account only the first $\widetilde{M} \ll M$ coefficients, which usually are the most significant ones, and setting the other $M - \widetilde{M}$ coefficients to zero. The reduced signal $\hat{w}_r \in \mathbb{R}^{\widetilde{M}}$ is quantized using the same step size for all its coefficients and then coded with the same entropy coder used for the image signal.

Given an image signal, we first solve the optimization problem in (11) obtaining the optimal solution w^*. To transmit w^* to the decoder, we first compute its GFT coefficients $\hat{w}^*$ and the reduced vector $\hat{w}^*_r$, then we quantize $\hat{w}^*_r$ and code it using the entropy coder described above. It is important to underline that, since we perform a quantization of $\hat{w}^*_r$, the reconstructed signal $\tilde{w}^*$ is not strictly equal to the original w^* and its quality depends on the quantization step size used. The graph described by $\tilde{w}^*$ is then used to define the GFT transform for the image signal.

Since it is important to find the best tradeoff between the quality of the graph and its transmission cost, for each block in an image we test different quantization step sizes $\{\Delta_i\}_{1 \le i \le Q}$ for a given graph represented by $\hat{w}^*_r$. To choose the best quantization step size, we use the following rate-distortion problem

$$\min_i \mathcal{D}(\Delta_i) + \gamma(\mathcal{R}_c(\Delta_i) + \mathcal{R}_G(\Delta_i)), \tag{12}$$

where $\mathcal{R}_G(\Delta_i)$ is the rate of $\hat{w}^*_{r,\Delta_i}$, the coefficient vector $\hat{w}^*_r$ quantized with Δ_i, and $\mathcal{D}(\Delta_i)$ and $\mathcal{R}_c(\Delta_i)$ are respectively the distortion and the rate of the reconstructed image signal obtained using the graph transform described by $\hat{w}^*_{r,\Delta_i}$. We point out that the choice of Δ_i depends on the quantization step size q used for the image transform coefficients $\hat{u}$. In fact, at high bitrate (small q) we expect to have a smaller Δ_i and thus a more precise graph, instead at low bitrate (large q) we will have a larger Δ_i that corresponds to a coarser graph approximation. We also underline that, in (12), we evaluate the actual distortion and rate without using the approximation introduced previously in (6), (7), (8). The actual coding methods described above are used to compute the rates $\mathcal{R}_c(\Delta_i)$ and $\mathcal{R}_G(\Delta_i)$. The principal steps of the proposed image compression method are summarized in Fig. 2.

6 Experimental results on natural images

In this section, we evaluate the performance of our illustrative graph-based encoder for natural images. We first describe the general experimental settings, then we present the obtained experimental results.

6.1 Experimental setup

First of all, we subdivide the image into non-overlapping 16×16 pixel blocks. For each block, we define the edge weights using the graph learning problem described in the previous sections. The chosen topology of the graph is a 4-connected grid: this is the most common graph topology for graph-based image compression, since its number of edges is not too high, and thus the coding cost is limited. In a 4-connected square grid with N nodes, we have $M = 2\sqrt{N}(\sqrt{N}-1)$ edges. In all our experiments on natural images, we use $Q = 8$ possible quantization step sizes Δ_i for $\hat{w}_r$ and we set $\widetilde{M} = 64$, which is the length of the reduced coefficient vector $\hat{w}_r$. In order to set the value of the parameter α in (11), we first have to perform a block classification. In fact, we recall that the parameter α in (11) is related to the l_1-norm of $\hat{w}$, where $\hat{w}$ are the GFT coefficients of the signal w that lies on the dual graph. As we have explained previously, the motivation for using the dual graph is that consecutive edges usually have similar values. However, this statement is not always true, but it depends on the characteristics of the block. In smooth blocks nearly all the edges will have similar values. Instead, in piecewise smooth blocks there could be a small percentage of edges whose consecutive ones have significantly different values. Finally, in textured blocks this percentage may even increase in a significant way. For this reason, we perform a priori a block classification using a structure tensor analysis, as done in [18]. The structure tensor is a matrix derived from the gradient of an image patch, and it is commonly used in many image processing algorithms, such as edge detection [37], corner detection [38,39] and feature extraction [40]. Let μ_1 and μ_2 be the two eigenvalues of the structure tensor, where $\mu_1 \geq \mu_2 \geq 0$. We classify the image blocks in the following way:

- Class 1: smooth blocks, if $\mu_1 \approx \mu_2 \approx 0$;
- Class 2: blocks with a dominant principal gradient, if $\mu_1 \gg \mu_2 \approx 0$;
- Class 3: blocks with a more complex structure, if μ_1 and μ_2 are both large.

Fig. 3 shows an example of block classification. For each block class, we have set the values of parameters α and β by fine tuning. We set $\alpha = 100$ for blocks

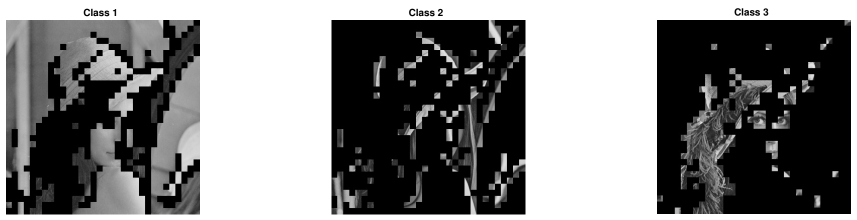


Figure 3: Block classification of the image Lena. Class 1 contains smooth blocks, class 2 blocks with a dominant principal gradient, and class 3 consists of blocks with a more complex structure.

	Class 1		Class 2		Class 3		Total	
Image	Learned graph	Gaussian graph	Learned graph	Gaussian graph	Learned graph	Gaussian graph	Learned graph	Gaussian graph
Lena	0.11	0.11	0.70	0.49	0.51	0.32	0.31	0.23
Boat	0.09	0.07	0.33	0.21	0.46	0.34	0.28	0.21
Peppers	0.18	0.14	0.87	0.45	0.88	0.57	0.47	0.31
House	0.05	0.05	0.68	0.59	0.61	0.49	0.36	0.31
Couple	0.26	0.29	1.17	0.75	0.98	0.81	0.66	0.55
Stream	0.17	0.19	0.50	0.30	0.31	0.22	0.31	0.23

Table 1: Bjontegaard average gain in PSNR for natural images w.r.t. DCT

that belong to the first class, $\alpha = 500$ for blocks that belong to the second class and $\alpha = 800$ for blocks that belong to the third class. For all the three classes, we set the same value for the other optimization parameter, i.e., $\beta = 1$.

We compare the performance of the proposed method to a baseline coding scheme built on the classical DCT transform. In order to obtain comparable results, we code the transform coefficients $\hat{u}$ of the image signal using the same entropy coder for the graph-based method and for the DCT-based encoder. In the first case, in addition to the bitrate of $\hat{u}$, we count the bitrate due to the transmission of $\hat{w}_i^*$ and $\log Q$ additional bits per block to transmit the chosen quantization step size Δ_i for $\hat{w}_r$. For both methods, we vary the quantization step size q of the transform coefficients to vary the encoding rates. In addition, in our method, for each block, we compare the RD-cost of the GFT and the one of the DCT. Then, we eventually code the block with the transform that has the lowest RD-cost and we use 1 additional bit per block to signal if we are using the GFT or the DCT.

In order to show the advantages of the proposed graph construction problem, we compare our method with a classical graph construction technique that uses a Gaussian weight function [32] to define the edge weights

$$W_{ij} = e^{-\frac{(u_i - u_j)^2}{\sigma^2}},$$

where σ is a gaussian parameter that we defined as $\sigma = 0.15 \max_{i,j} |u_i - u_j|$. In order to have comparable results, we use the coding scheme described in Sec. V also for the Gaussian graph.

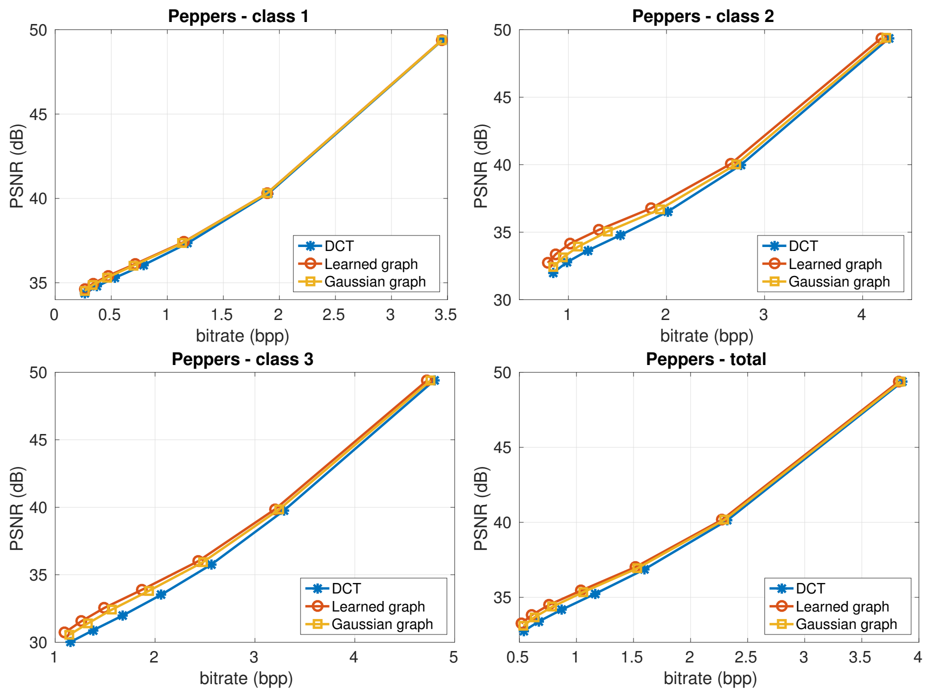


Figure 4: Rate-distortion curves for the image Peppers.

6.2 Results

The experiments are performed on six classical grayscale images (House, Lena, Boat, Peppers, Stream, Couple) [41]. This dataset contains different types of natural images, for example some of them have smooth regions (e.g. House and Peppers), others instead are more textured (e.g. Boat, Lena, Couple and Stream). In Table 1, we show the obtained performance results in terms of average gain in PSNR compared to DCT, evaluated through the Bjontegaard metric [42]. Moreover, in Fig. 4 we show the rate-distortion curves for the image Peppers. Instead, in Fig. 5 we show a visual comparison between the DCT and the proposed method for the image Peppers. We see that, in the second and third classes, the proposed method outperforms DCT providing an average PSNR gain of 0.6 dB for blocks in the second class and 0.64 dB for blocks in the third class. It should be pointed out that there is not a significant difference in performance between the second class and the third one. This probably is due to the fact that the proposed graph construction method is able to adapt the graph and its description cost to the characteristics of each block. Instead, in the first class, which corresponds to smooth blocks, the gain is nearly 0, as DCT in this case is already optimal. Finally, we notice that, in the classes where the DCT is not optimal, the learned graph always outperforms the Gaussian graph.

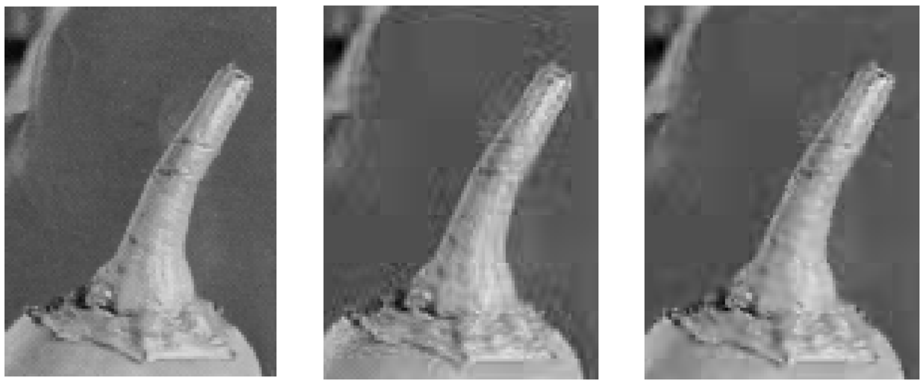


Figure 5: Visual comparison for a detail of the image Peppers at 0.6 bpp.

7 Experimental results on piecewise smooth images

In this section, we evaluate the performance of the proposed method on piecewise smooth images, comparing our method with classical DCT and the state-of-the-art graph-based coding method of [6]. We first describe the specific experimental setting used for this type of signals, then we present the obtained results.

7.1 Experimental setup

We choose as piecewise smooth signals six depth maps taken from [43,44]. Similarly to the case of natural images, we split them into non-overlapping 16×16 pixel blocks and the chosen graph topology is a 4-connected grid. In addition, we keep for Q the same setting as the one used for natural images. Then, to define the parameters α and β we again subdivide the image blocks into classes using the structure tensor analysis. In [6], the authors have identified three block classes for piecewise smooth images: smooth blocks, blocks with weak boundaries (e.g., boundaries between different parts of the same foreground/background) and blocks with sharp boundaries (e.g., boundaries between foreground and backgound). In our experiments, since we have observed that the first two classes have a similar behavior, we decided to consider only two different classes:

- Class 1: smooth blocks and blocks with weak edges, if $\mu_1 \approx \mu_2 \approx 0$.
- Class 2: blocks with sharp edges, if $\mu_1 \gg 0$,

where μ_1 and μ_2, with $\mu_1 \geq \mu_2 \geq 0$, are the two eigenvalues of the structure tensor. An example of block classification is shown in Fig. 6. As done for natural images, for each class we set parameters α and β by fine tuning. For

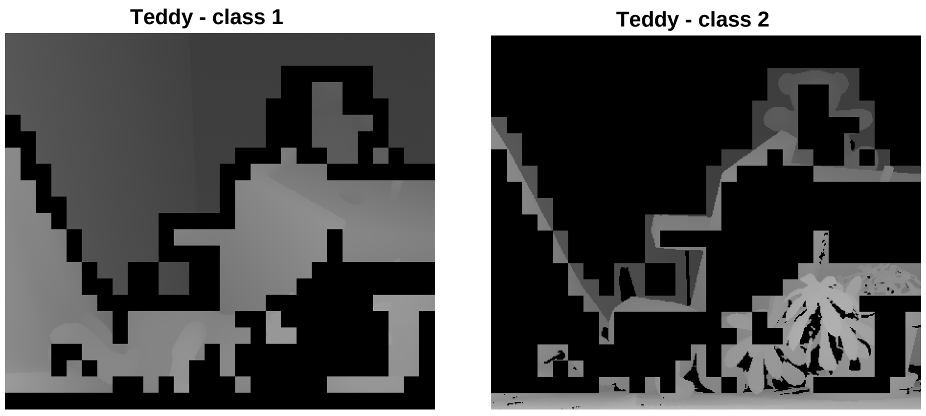


Figure 6: Block classification of the image Teddy.

the first class, we set $\alpha = 40$ and $\beta = 0.02$. For the second class, we set $\alpha = 400$ and $\beta = 1$.

With this type of signals, we have observed that the coefficients $\hat{w}^*$ of the learned graph are very sparse, as shown in Fig. 7. For this reason, we decided to modify the coding method used for $\hat{w}^*$. As done for natural images, we reduce the number of elements in $\hat{w}$ by taking into account only the first $\widetilde{M}$ coefficients (in this case we set $\widetilde{M} = 256$). Then, we use an adaptive binary arithmetic encoder to transmit a significance map that signals the non-zero coefficients. In this way, we can use an adaptive bitplane arithmetic encoder to code only the values of the non-zero coefficients. This allows a strong reduction of the number of coefficients that we have to transmit to the decoder.

Similarly to the case of natural images, we compare our method to a transform coding method based on the classical DCT. However, in the specific case of depth map coding it has been shown that graph-based methods significantly outperforms the classical DCT. For this reason, we also propose a comparison with a graph-based coding scheme that is specifically designed for piecewise smooth images. The method presented in [6] achieves the state-of-the-art performance in graph-based depth image coding. This method uses a table-lookup based graph transform: the most popular GFTs are stored in a lookup table, and then for each block an exhaustive search is performed to choose the best GFT in rate-distortion terms. In this way, the side information that has to be sent to the decoder is only the table index. Moreover, the method in [6] incorporates a number of coding tools, including multiresolution coding scheme and edge-aware intra-prediction. Since in our case we are interested in evaluating the performance of the transform, we only focus on the transform part and we use as reference method a simplified version of the method in [6] that is similar to the one used in [45]. The simplified version of [6] that we implemented employs 16$\times$16 blocks and it does not make use of edge-aware prediction and multiresolution coding. Since the transform used in [6] is based on a lookup

Table 2: Bjontegaard average gain in PSNR compared to DCT.

Image	Class 1	Class 2	Total
Teddy	0.59	6.12	4.82
Cones	0.70	8.37	6.88
Art	0.56	8.66	7.62
Dolls	0.55	8.59	5.57
Moebius	0.82	7.36	5.52
Reindeer	0.45	8.34	5.75

Table 3: Bjontegaard average gain in PSNR between the proposed method and the reference method.

Image	Class 1	Class 2	Total
Teddy	-0.87	-0.12	-0.38
Cones	-1.21	1.17	0.54
Art	-0.89	0.86	0.49
Dolls	-0.78	1.16	0.26
Moebius	-1.14	-0.47	-0.76
Reindeer	-0.51	0.02	-0.33

table, we use 40 training depth images to build the table as suggested in [6]. In the training phase, we identify the most common graph transforms. As a result, the obtained lookup table contains 718 transforms. Then, in the coding phase each block is coded using one of the transforms contained in the lookup table or the DCT. The coding method used for the table index is the same as in [6]. Instead for the transform coefficients $\hat{u}$, in order to have comparable results, we use the coding method described in Sec. 5.

7.2 Results

The first coding results on depth maps are summarized in Table 2, where we show the average gain in PSNR compared to DCT. Instead, in Table 3 we show the Bjontegaard average gain in PSNR between the proposed method and the reference method described previously. Moreover, in Fig. 8 we show the rate-distortion curves for the image Dolls. Finally, Fig. 9 shows an example of a decoded image obtained using the proposed method.

The results show that the proposed technique provides a significant quality gain compared to DCT, displaying a behavior similar to other graph-based techniques. Moreover, it is important to highlight that the performance of the proposed method are close to that of the state-of-the-art method [6], although our method is not optimized for piecewise smooth images, but it is a more general method that can be applied to a variety of signal classes. In particular, for the blocks belonging to the second class, in 4 out of 6 images (namely Cones, Art, Dolls and Moebius) we are able to outperform the reference method, reaching in some cases a quality gain larger than 1 dB (see Table 3). Overall, with our more generic compression framework, we outperform the reference method in approximately half of the test images. In general, we observe that the proposed method outperforms the reference one in blocks that have several edges or edges that are not straight. This is probably due to the fact that, in these cases, it is more difficult to represent the graph using a lookup table. It is also worth noting that our method shows better performance at low bitrate, as it is possible to see in Fig. 8.

8 Conclusion

In this paper, we have introduced a new graph-based framework for signal compression. First, in order to obtain an effective coding method, we have formulated a new graph construction problem targeted for compression. The solution of the proposed problem is a graph that provides an effective tradeoff between the energy compaction of the transform and the cost of the graph description. Then, we have also proposed an innovative method for coding the graph by treating the edge weights as a new signal that lies on the dual graph. We have tested our method on natural images and on depth maps. The experimental results show that the proposed method outperforms the classical DCT and, in the case of depth map coding, even compares to the state-of-the-art graph-based coding method.

We believe that the proposed technique participates to opening a new research direction in graph-based image compression. As future work, it would be interesting to investigate other possible representation for the edge weights of the graph, such as graph dictionaries or graph wavelets. This may lead to further improvements in the coding performance of the proposed method.

Acknowledgment

This work was partially supported by Sisvel Technology.

References

[1] D. Shuman, S. 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,” *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.

[2] N. Ahmed, T. Natarajan, and K. Rao, “Discrete cosine transform,” *IEEE Trans. Computers*, vol. C-23, no. 1, pp. 90–93, 1974.

[3] D. K. Hammond, P. Vandergheynst, and R. Gribonval, “Wavelets on graphs via spectral graph theory,” *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.

[4] G. Shen, W. S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, “Edge-adaptive transforms for efficient depth map coding,” in *Proc. Picture Coding Symposium (PCS)*, 2010, pp. 2808–2811.

[5] W. Kim, S. K. Narang, and A. Ortega, “Graph based transforms for depth video coding,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 813–816.

[6] W. Hu, G. Cheung, A. Ortega, and O. C. Au, “Multiresolution graph fourier transform for compression of piecewise smooth images,” *IEEE Trans. on Image Process.*, vol. 24, no. 1, pp. 419–433, 2015.

[7] G. Fracastoro, D. Thanou, and P. Frossard, “Graph transform learning for image compression,” in *Proc. Picture Coding Symposium (PCS)*, 2016, pp. 1–5.

[8] K. Sayood, *Introduction to data compression.* Newnes, 2012.

[9] V. K. Goyal, J. Zhuang, and M. Vetterli, “Transform coding with backward adaptive updates,” *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1623–1633, 2000.

[10] A. K. Jain, “A sinusoidal family of unitary transforms,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 4, pp. 356–365, 1979.

[11] W.-T. Su, G. Cheung, and C.-W. Lin, “Graph fourier transform with negative edges for depth image coding,” *arXiv preprint arXiv:1702.03105*, 2017.

[12] W. Hu, G. Cheung, and A. Ortega, “Intra-prediction and generalized graph fourier transform for image coding,” *IEEE Signal Process. Lett.*, vol. 22, no. 11, pp. 1913–1917, 2015.

[13] H. E. Egilmez, A. Said, Y.-H. Chao, and A. Ortega, “Graph-based transforms for inter predicted video coding,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 3992–3996.

[14] H. E. Egilmez, Y.-H. Chao, A. Ortega, B. Lee, and S. Yea, “Gbst: Separable transforms based on line graphs for predictive video coding,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 2375–2379.

[15] K.-S. Lu and A. Ortega, “Symmetric line graph transforms for inter predictive video coding,” in *Proc. Picture Coding Symposium (PCS)*, 2016.

[16] G. Fracastoro and E. Magli, “Predictive graph construction for image compression,” in *Proc. IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 2204–2208.

[17] E. Pavez, H. E. Egilmez, Y. Wang, and A. Ortega, "GTT: Graph template transforms with applications to image coding," in *Proc. Picture Coding Symposium (PCS)*, 2015, pp. 199–203.

[18] I. Rotondo, G. Cheung, A. Ortega, and H. E. Egilmez, "Designing sparse graphs via structure tensor for block transform coding of images," in *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2015, pp. 571–574.

[19] E. Pavez and A. Ortega, "Generalized laplacian precision matrix estimation for graph signal processing," in *Proc. IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2016, pp. 6350–6354.

[20] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under structural and laplacian constraints," *arXiv preprint arXiv:1611.05181*, 2016.

[21] E. Pavez, H. E. Egilmez, and A. Ortega, "Learning graphs with monotone topology properties and multiple connected components," *arXiv preprint arXiv:1705.10934*, 2017.

[22] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160–6173, 2016.

[23] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016, pp. 920–929.

[24] J. Gallier, "Elementary spectral graph theory applications to graph clustering using normalized cuts: a survey," *arXiv preprint arXiv:1311.2492*, 2013.

[25] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *Proc. ICML Workshop on Statistical Relational Learning and its Connections to other Fields*, 2004, pp. 132–137.

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

[27] H. Rue and L. Held, *Gaussian Markov random fields: theory and applications.* CRC Press, 2005.

[28] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2325–2383, 1998.

[29] S. Mallat and F. Falzon, "Analysis of low bit rate image transform coding," *IEEE Trans. on Signal Process.*, vol. 46, no. 4, pp. 1027–1042, 1998.

[30] X. Liu, G. Cheung, and X. Wu, "Joint denoising and contrast enhancement of images using graph laplacian operator," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 2274–2278.

[31] W. Hu, G. Cheung, and M. Kazui, "Graph-based dequantization of block-compressed piecewise smooth images," *IEEE Signal Process. Lett.*, vol. 23, no. 2, pp. 242–246, 2016.

[32] L. J. Grady and J. Polimeni, *Discrete calculus: Applied analysis on graphs for computational science.* Springer Science & Business Media, 2010.

[33] S. Boyd and L. Vandenberghe, *Convex optimization.* Cambridge university press, 2004.

[34] T. Wiegand and H. Schwarz, *Source coding: Part I of fundamentals of source and video coding.* Now Publishers Inc, 2010.

[35] S. Mallat, *A wavelet tour of signal processing: the sparse way.* Academic press, 2008.

[36] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[37] U. Köthe, "Edge and junction detection with an improved structure tensor," in *Joint Pattern Recognition Symposium.* Springer, 2003, pp. 25–32.

[38] C. Harris and M. Stephens, "A combined corner and edge detector." in *Alvey vision conference*, vol. 15, no. 50. Manchester, UK, 1988, pp. 10–5244.

[39] C. S. Kenney, M. Zuliani, and B. Manjunath, "An axiomatic approach to corner detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 191–197.

[40] W. Förstner, "A feature based correspondence algorithm for image matching," *International Archives of Photogrammetry and Remote Sensing*, vol. 26, no. 3, pp. 150–166, 1986.

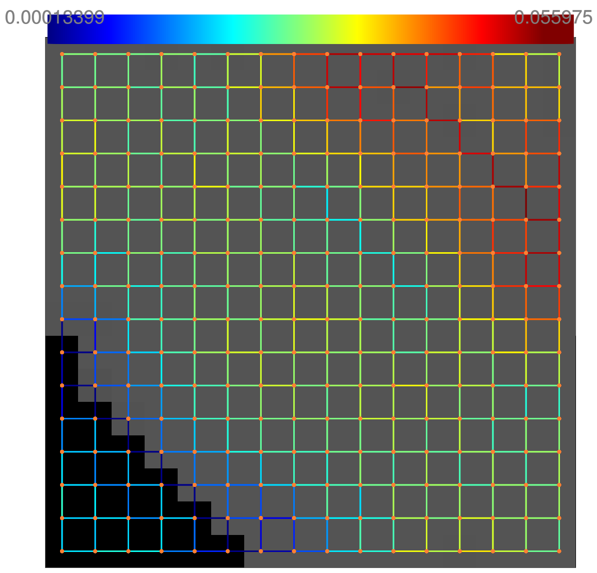
[41] USC-SIPI, "Image database, volume 3: Miscellaneous," http://sipi.usc.edu/database/database.php?volume=misc.

[42] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," *Doc. VCEG-M33 ITU-T Q6/16, Austin, TX, USA, 2-4 April 2001*, 2001.

[43] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2003, pp. 195–202.

[44] D. Scharstein and C. Pal, “Learning conditional random fields for stereo,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8.

[45] D. Zhang and J. Liang, “Graph-based transform for 2d piecewise smooth signals with random discontinuity locations,” *IEEE Trans. on Image Process.*, vol. 26, no. 4, pp. 1679–1693, 2017.



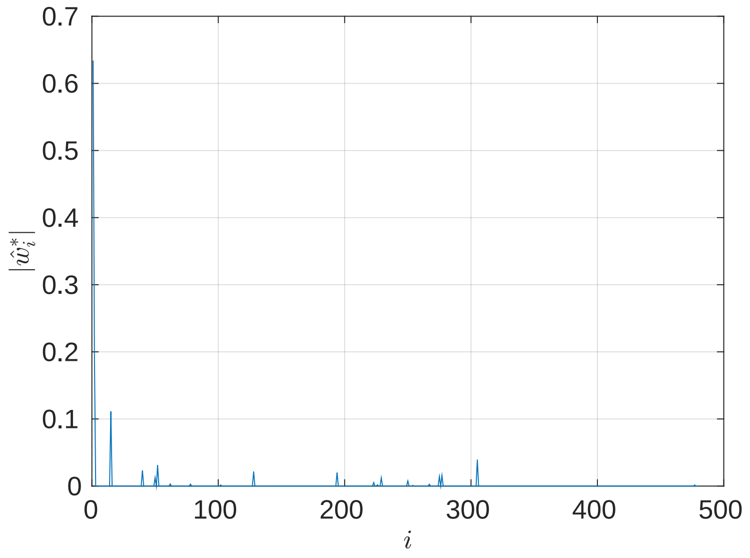


Figure 7: Top: Example of a piecewise smooth block and the corresponding learned graph. Bottom: The corresponding GFT coefficients $\hat{w}$.

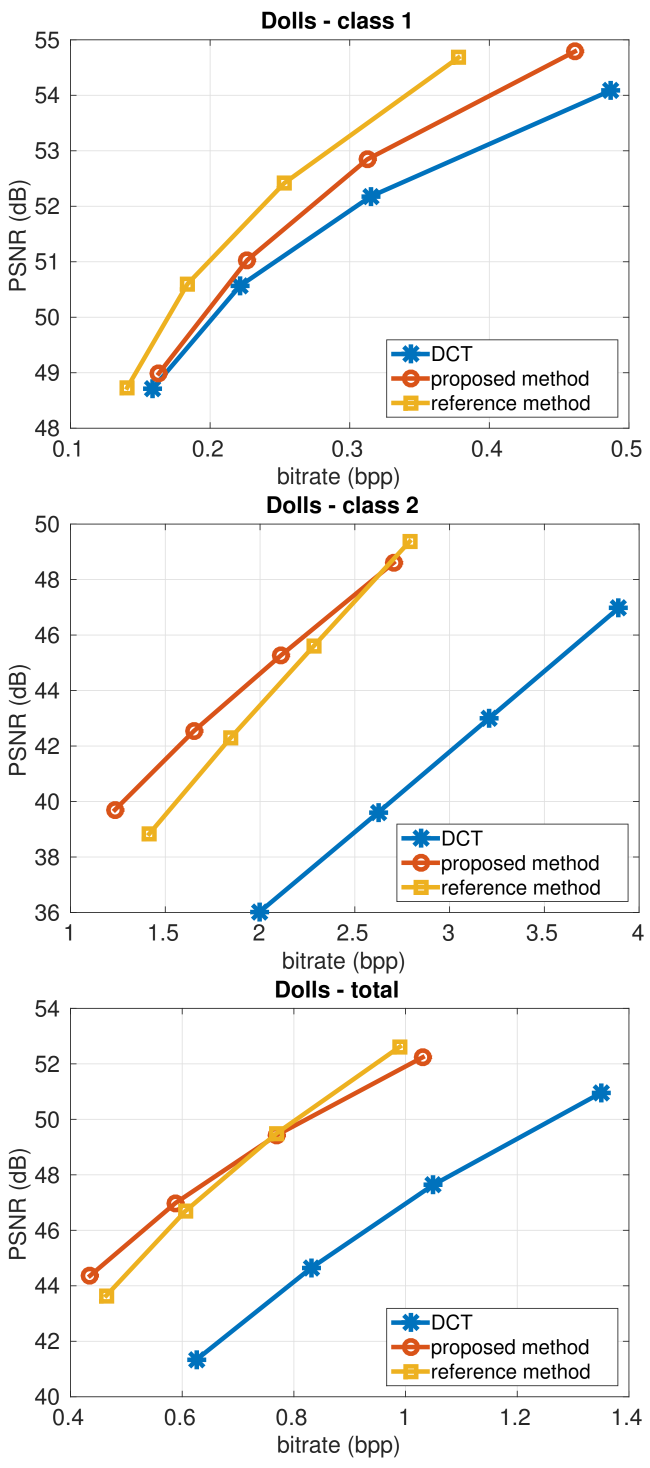
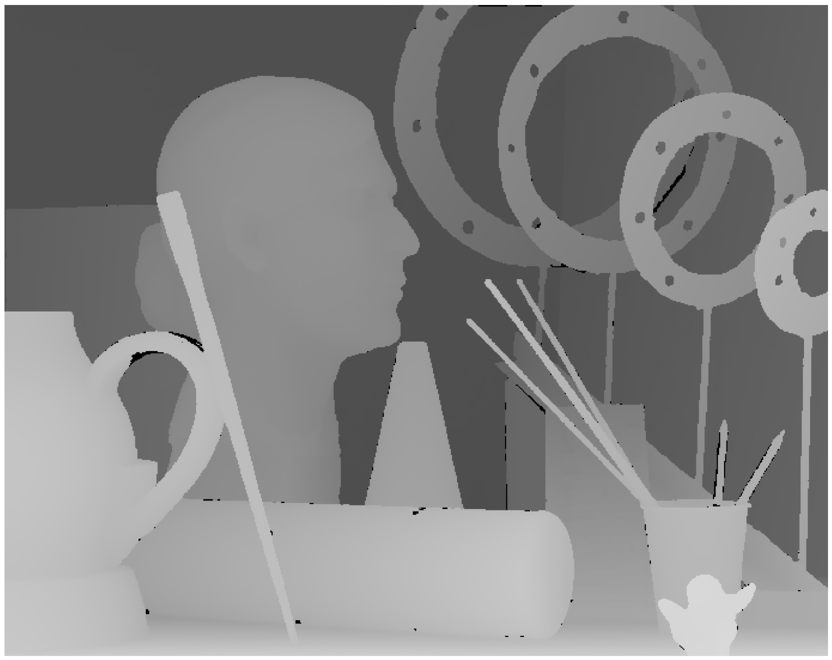


Figure 8: Rate-distortion curves for the image Dolls.



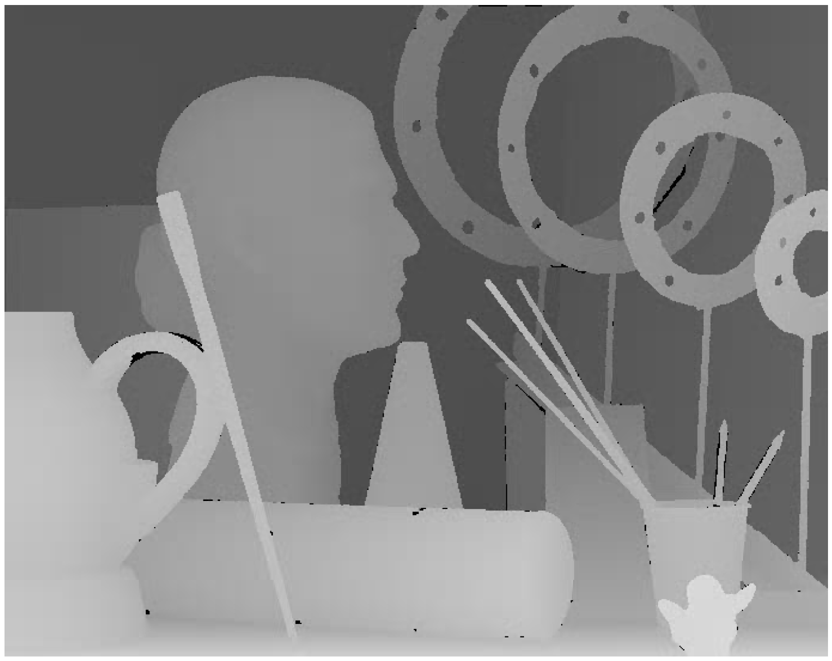


Figure 9: Top: Original image. Bottom: Decoded image using the proposed method (0.6 bpp).