SUPERPIXEL-DRIVEN GRAPH TRANSFORM FOR IMAGE COMPRESSION

Giulia Fracastoro[†]*, Francesco Verdoja*[‡]*, Marco Grangetto*[‡]*, Enrico Magli*[†]

† Politecnico di Torino, Dept. of Electronics and Telecommunications
‡ Università degli Studi di Torino, Dept. of Computer Science

ABSTRACT

Block-based compression tends to be inefficient when blocks contain arbitrary shaped discontinuities. Recently, graph-based approaches have been proposed to address this issue, but the cost of transmitting graph topology often overcome the gain of such techniques. In this work we propose a new Superpixel-driven Graph Transform (SDGT) that uses clusters of superpixels, which have the ability to adhere nicely to edges in the image, as coding blocks and computes inside these homogeneously colored regions a graph transform which is shape-adaptive. Doing so, only the borders of the regions and the transform coefficients need to be transmitted, in place of all the structure of the graph. The proposed method is finally compared to DCT and the experimental results show how it is able to outperform DCT both visually and in term of PSNR.

Index Terms— Image compression, graph transform, superpixels, clustering

1. INTRODUCTION

Block transform based compression is by far the most widespread approach to lossy image coding: an image is first subdivided into non-overlapping blocks of pixels, then each block is projected into a chosen transform domain. The most common compression codecs (e.g. JPEG, H.264) use this approach because it can easily adapt to the non-stationary statistics of natural images, it is computationally efficient and amenable to parallel implementation.

The Discrete Cosine Transform (DCT) is widely used for block-based image and video compression [1]. One of the main drawbacks of the DCT is that it becomes inefficient when a block contains arbitrarily shaped discontinuities. In this case, the DCT will generate a non-sparse signal representation, having high-frequency coefficients with large magnitude. This will result in poor coding performance.

To solve this problem, in the past years different solutions have been proposed. Some variations of the DCT have been developed, such as directional DCT [2], adaptive block-size transform [3] or shape-adaptive DCT [4], in which the block size and shape are defined taking into account the edge location or the transform is evaluated avoiding to cross edge discontinuities. Wavelet approaches have also been introduced. To avoid filtering across edges, researchers have studied different wavelet filter-banks based on the image geometry, e.g. bandelets [5], directionlets [6] and curvelets [7]. However, all the proposed methods produce an efficient signal representation only when edges are straight lines, making them inefficient in presence of shaped contours. Recently, a novel graph-based transform approach has been proposed. Any image can be viewed as a graph, where each pixel is a node of the graph and weighted edges

Fig. 1. An image divided into 100 regions by the proposed algorithm.

describe the connectivity relations among the pixels, e.g. in terms of similarity. In the last years, researchers have made some attempts to develop graph-based compression techniques. Indeed, the graph representation allows one to design an edge-aware and shape-adaptive transform in an elegant and effective way. Block-based method using graph Fourier transform have been proposed in [8, 9]. Instead, in [10] a method for image compression using graph-based wavelet transform is presented. However, they all reported unsatisfactory results on natural images that are not piece-wise smooth. One of the main drawbacks of graph-based compression techniques lies in the cost required to represent and encode the graph, which may outweigh the coding gain provided by the edge adaptive transform.

In this work, we propose a novel graph transform approach aiming at reducing the cost of transmitting the graph structure while retaining the advantage of a shape-adaptive and edge-aware operator. To this end, the image is first segmented into uniform regions that adhere well to image boundaries. Such a goal can be achieved using the so-called superpixels, which are perceptually meaningful atomic regions which aim at replacing rigid pixel grid. Examples of algorithms used to generate these kind of regions are Turbopixel [11], VCells [12] and the widely used and very fast SLIC algorithm [13]. Then, we propose to apply a graph transform within each superpixel that, being homogeneous region, can be efficiently represented using an uniform graph, i.e. all graph edges are given the same weight. In this way, the overhead of representing the graph structure within each superpixel is avoided. Nonetheless, we need to transmit additional information to describe region boundaries. To limit such coding overhead, we design a clustering method that is able to aggregate superpixels, thus reducing the number of regions that need to be coded.

The use of superpixels in compression is still an almost unexplored research field and up to date only few works investigated the topic. Moreover, the proposed approaches work in very specific cases, e.g. texture compression [14] or user-driven compression [15]. On the contrary, the joint exploitation of graph transforms and superpixels as a general approach to image compression is completely novel and represents the key idea in this work. The contributions of the paper are the definition of superpixel-driven graph transform, its rate/distortion analysis using a bitplane encoding approach and the comparison with standard DCT transform.

The paper is organized as follows: in Section 2 the proposed algorithm is going to be presented in detail, while in Section 3 the results of our experimental tests are going to be presented. A final discussion on the method is going to be conducted in Section 4.

2. THE PROPOSED TECHNIQUE

Given an image $I = \{x_i\}_{i=1}^N$ of N pixels, the proposed Superpixel-driven Graph Transform (SDGT) performs the following steps:

- divide I in m regions by using SLIC [13];
- cluster similar superpixels, to reduce the number of borders to be coded to a desired number m';
- inside each region, compute a piece-wise smooth graph transform.

Superpixels are used to get a computationally efficient segmentation of the image into homogeneous regions, that can be modeled with simple uniform graph structure for the following transform stage.

2.1. Superpixel clustering

In this section the preliminary segmentation step based on superpixel is described.

We define an m-regions segmentation of an image I as a partition $P^m = \{l_i\}_{i=1}^m$ of the pixels in I; more precisely:

$$\begin{aligned} \forall x \in I, & \quad \exists l \in P^m \mid x \in l \\ \forall l \in P^m, & \quad \nexists l' \in P^m - \{l\} \mid l \cap l' \neq \emptyset \end{aligned} \quad (1)$$

Starting from an image I and a partition P^m composed of m regions, output by some superpixel algorithm, the proposed algorithm aims at merging at each iteration the pair of labels representing the most similar regions between the ones determined in the previous step until the desired number of regions $m' < m$ is reached. In particular at the k-th iteration the two most similar segments of P^k are merged to obtain a new set P^{k-1} composed of $k-1$ segments. This process can be iterated for $k = m, m-1, \ldots, m'$, generating a hierarchy of regions in terms of their respective similarity. The number of regions m' to be clustered must be chosen as a tradeoff between the segmentation accuracy and the coding overhead required to represent and compress the borders of the regions as discussed in more detail in Section 3.

We represent the merging process using a weighted graph. An initial undirected weighted graph $G^m = (P^m, W^m)$ is constructed over the superpixel set P^m, where

$$W^m = \{w_{ij}^m, \ \forall i \neq j \mid l_i^m, l_j^m \in P^m \wedge C(l_i^m, l_j^m) = 1\} \quad (2)$$

for some region adjacency function C, i.e. 4-connectivity. Since G^m is an undirected graph we have that $w_{ij}^m = w_{ji}^m$; the weights represent the distance (or dissimilarity measure) between a pair of regions $w_{ij}^m = \delta(l_i^m, l_j^m)$.

The approach proposed here can be used in conjunction with several distance metrics capable to capture the dissimilarity between a pair of segmented regions. In this study, CIELAB color space and the standard CIEDE2000 color difference [16] have been chosen thanks to their ability to reliably cluster similar superpixels as shown in [17]. Given two regions l_i and l_j, we compute the mean values of the L*a*b* components $M_i = (\mu_{L^*,i}, \mu_{a^*,i}, \mu_{b^*,i})$ and $M_j = (\mu_{L^*,j}, \mu_{a^*,j}, \mu_{b^*,j})$, and we define the distance between the two labels as

$$\delta(l_i, l_j) = \Delta E_{00}(M_i, M_j) \quad (3)$$

where ΔE_{00} is the CIEDE2000 color difference [16].

At each iteration, we pick the pair of labels $l_p^k, l_q^k \in P^k$ having $w_{pq}^k = \min\{W^k\}$ and merge them; as a consequence a new partition $P^{k-1} = P^k - \{l_q^k\}$ having all the pixels $x \in l_p^k \cup l_q^k$ assigned to the label l_p^{k-1} is formed, where P^{k-1} now comprises $k-1$ segments. After that, edges and corresponding weights needs to be updated as well. W^{k-1} is generated according to the following rule:

$$w_{ij}^{k-1} = \begin{cases} \delta(l_p^{k-1}, l_j^{k-1}) & \text{if } i = p \vee i = q \\ w_{ij}^k & \text{otherwise} \end{cases} \quad (4)$$

It must be noted that w_{pq}^k is no longer included in W^{k-1} since the corresponding label has been merged into a single one.

When $k = m'$, the algorithm stops returning the partition $P^{m'}$ composed of the desired number of regions. A segmentation example with $m' = 100$ is shown Figure 1.

2.2. Intra-region graph transform

Now we move to the description of the graph transform employed within each region that leads to the computation of the proposed SDGT.

Given a m'-regions segmentation $P^{m'}$ of the image I, in each segment l of $P^{m'}$ we can define a graph $G_l = (l, E)$, where the nodes are the pixels of the segment l and $E \subset l \times l$ is the set of edges. The adjacency matrix A is defined in the following way:

$$A_{ij} = \begin{cases} 1 & \text{if } j \in N_i \wedge i, j \in l \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where N_i is the set of 4-connected neighbors of the pixel i.

The adjacency matrix is used to compute the Laplacian matrix $L = D - A$, where D is a diagonal matrix whose i-th diagonal element d_i is equal to the sum of the weights of all edges incident to node i. The Laplacian matrix L is a symmetric positive semi-definitive matrix. Then, it has an eigen decomposition:

$$L = U^T \Lambda U \quad (6)$$

where U is the matrix whose rows are the eigenvectors of the graph Laplacian and Λ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues.

The matrix U is used to compute the graph Fourier transform (GFT): for any signal $\mathbf{f} \in \mathbb{R}^N$ defined on the vertices of the graph, its GFT $\hat{\mathbf{f}}$ is defined in [18] as:

$$\hat{\mathbf{f}} = U\mathbf{f} \quad (7)$$

The inverse graph Fourier transform is then given by

$$\mathbf{f} = U^T \hat{\mathbf{f}} \quad (8)$$

2632
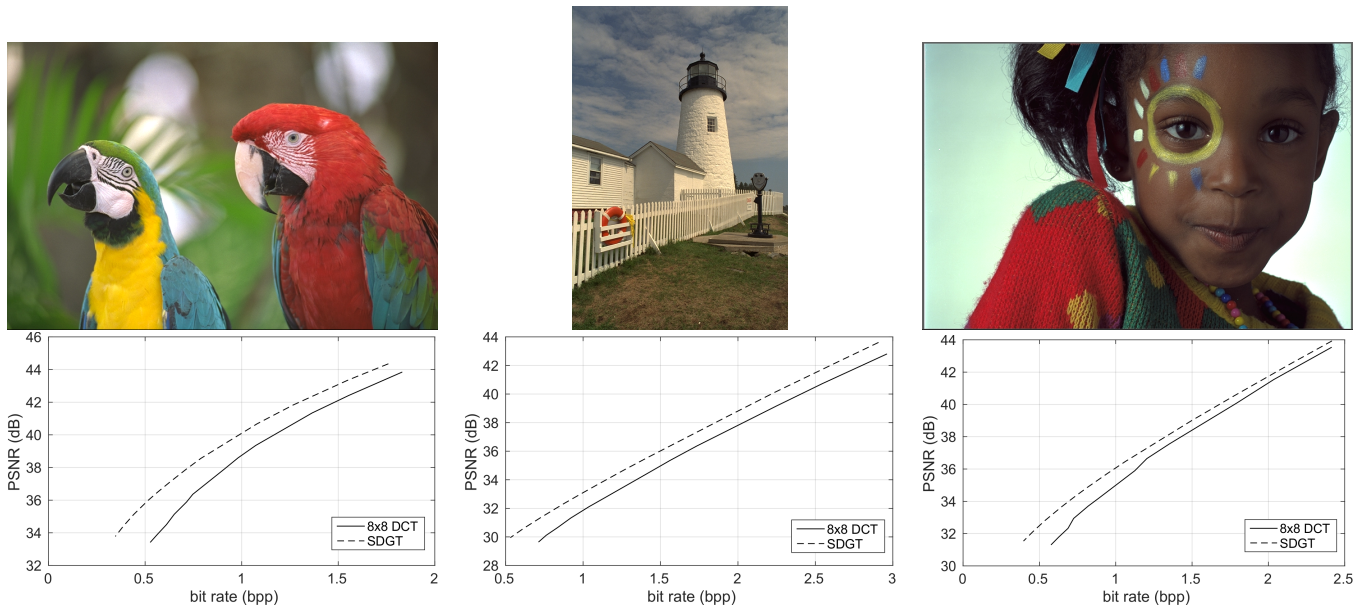
Fig. 2. Three of the sample images (top), for each of them the performance of the proposed SDGT and DCT 8×8 is presented in term of PSNR values over bitrate (bottom).

It is important to underline that to construct the graph we only need the information about the coordinates of the region borders, that can be easily summarized in a binary image. In this way, the cost for transmitting the graph structure is considerably reduced and the GFT is used as an effective transform for the arbitrarily shaped regions computed by the algorithm described in Section 2.1. Finally, we refer to the whole set of transformed regions as the SDGT of the entire image.

3. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed SDGT, we need to take into account its energy compaction ability and the cost for coding overhead information, i.e. the region-borders.

A popular and simple method for evaluating the transform compaction efficiency is to study the quality of the reconstructed image, e.g. using PSNR with respect to the original image, as a function of the percentage of retained transformed coefficients [19]; albeit interesting, this approach would neglect the cost required to encode the ancillary information required to compute the inverse transform.

To overcome this, in the following we estimate the coding efficiency provided by SDGT by considering bit plane encoding of SDGT transformed coefficients. Each bitplane is progressively extracted, from the most significant down to least significant one, and the bitrate of each bitplane is estimated by its entropy. To this end, each bitplane is modeled as an independent and memoryless binary source.

It is worth pointing out that such an estimate represents an upper bound to the actual bitrate that would be obtained using a proper entropy coding algorithm that is likely to exploit further the residual spatial correlation of the transformed coefficients and the dependency between different bitplanes. Nonetheless, the proposed bitplane approach can be replicated on any other transform, e.g. the standard 8×8 DCT, allowing us to analyze the achievable gain in a fair way.

Finally, to estimate the SDGT penalty due to coding of the region borders, we use the standard compression algorithm for bi-level images JBIG [20]. The regions boundaries are represented as a binary mask that is then compressed with JBIG, whose bitrate is considered as coding overhead; from our experimentation we have seen that this overhead is, on average, around 0.06 bpp. The use of other more specific methods for transmitting the region borders, such as [21], will be evaluated in future.

Therefore using bitplane coding and JBIG we get a rough estimation of the total bitrate needed to code the image with the SDGT transform. We compare the obtained results with the standard DCT computed on 8×8 blocks. As proved by Zhang and Florêncio in [22], if the graph is a uniform 4-connected grid the 2D DCT basis functions are eigenvectors of the graph Laplacian, and thus the transform matrix U used in (6) turns to be the 2D DCT matrix. Therefore, the 8×8 DCT can be seen as a graph transform like the SDGT, with the major difference that instead of using superpixels as coding blocks it uses a fixed grid of 8×8 blocks.

We have tested the transforms on several images from a dataset of lossless images widely used in compression evaluation [23]. All the images in that dataset are either 768×512 or 512×768 in size. In Figure 2 three sample images are shown along with the respective coding results (PSNR in dB vs. bitrate measured in bit per pixel); these results have been obtained setting $m = 600$, $m' = 100$ and coding the luminance component only.

We can see that SDGT significantly outperforms the DCT, in particular at low bitrate, where it is able to achieve a maximum gain of more than 2 dB. Overall, the average gain obtained is approximately 1 dB. This achievement is particularly significant if one recall that the SDGT bitrate includes the constant penalty yielded by JBIG coding of the borders. A detail of the significant improvement at low bitrate obtained by SDGT can be visually appreciated in Figure 3.

Since standard image compression data set are historically biased by low resolution images we conclude our analysis by considering high resolution images that are typically acquired by current

(a) DCT 8×8

(b) SDGT

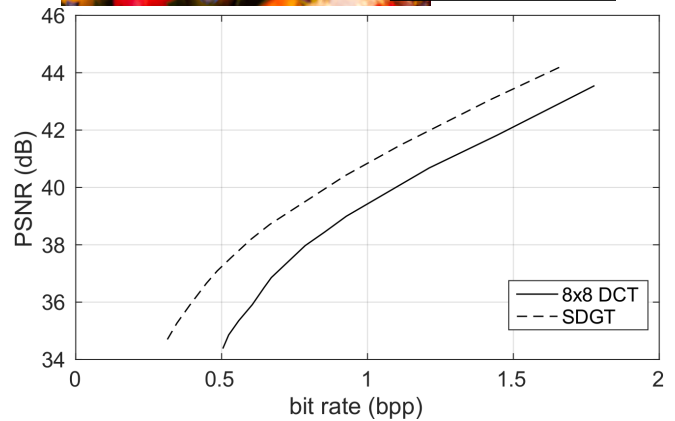**Fig. 3**. A detail on the luminance component of one image compressed with both DCT 8×8 and the proposed SDGT at bitrate of 0.75 bpp.

Fig. 4. A 2592×3888 sample image with a 512×512 cropped patch (top) and the performance of the proposed SDGT and 8×8 DCT on the cropped region in term of PSNR values over bitrate (bottom).

imaging devices. We have tested our method and the 8×8 DCT on some HD images acquired using a DSLR camera; in particular, for complexity reasons, we have applied SDGT to non trivial 512×512 patches cropped from the original images. In Figure 4 the results obtained on a sample image are shown; it is worth pointing out that the SDGT gain over DCT is larger in this case and span all the considered bitrate range. This is due to the fact that regions in HD images are usually wider and smoother and therefore the segmentation algorithm and, consequently, the graph transform can be even more effective.

4. CONCLUSIONS AND FUTURE WORK

In this study we have explored a new graph transform for image compression applications. It is shown that the proposed algorithm achieves better performance than DCT, especially at lower bitrates and on high-resolution images.

The main contribution of this work is to set the foundation for a new approach to graph-based image compression. Thanks to exploitation of superpixel ability to adhere to image borders, we can subdivide the image in uniform regions and use the graph transform inside each region as a shape adaptive transform.

Future work on the proposed algorithm might include trying to interpolate the pixels inside the regions starting from the ones on the borders and then encode only the prediction errors, reducing in a significant way the information needed to be encoded.

5. REFERENCES

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

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

[3] M. Wien, "Variable block-size transforms for H.264/AVC," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 604–613, 2003.

[4] T. Sikora and B. Makai, "Shape-adaptive DCT for generic coding of video," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 5, no. 1, pp. 59–62, 1995.

[5] E. Le Pennec and S. Mallat, "Sparse geometric image representations with bandelets," *Image Processing, IEEE Transactions on*, vol. 14, no. 4, pp. 423–438, 2005.

[6] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P.L. Dragotti, "Directionlets: anisotropic multidirectional representation with separable filtering," *Image Processing, IEEE Transactions on*, vol. 15, no. 7, pp. 1916–1933, 2006.

[7] E.J. Candes and D.L. Donoho, "Curvelets: A surprisingly effective nonadaptive representation for objects with edges," Tech. Rep., DTIC Document, 2000.

[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. 2808–2811.

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

[10] S.K. Narang, Y.H. Chao, and A. Ortega, "Critically sampled graph-based wavelet transforms for image coding," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*. IEEE, 2013, pp. 1–4.

[11] A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, and K. Siddiqi, "Turbopixels: Fast superpixels using geometric flows," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2290–2297, Dec. 2009.

[12] J. Wang and X. Wang, "VCells: Simple and efficient superpixels using edge-weighted centroidal voronoi tessellations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 6, pp. 1241–1247, June 2012.

[13] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, nov 2012.

[14] P. Krajcevski and D. Manocha, "SegTC: Fast texture compression using image segmentation," *Eurographics Association, Lyon, France, I. Wald and J. Ragan-Kelley, Eds*, pp. 71–77, 2014.

[15] N. Brewer, L. Wang, N. Liu, and L. Cheng, "User-driven lossy compression for images and video," in *Image and Vision Computing New Zealand, 2009. IVCNZ'09. 24th International Conference*. 2009, pp. 346–351, IEEE.

[16] G. Sharma, W. Wu, and E.N. Dalal, "The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations," *Color Research and Application*, vol. 30, no. 1, pp. 21–30, 2005.

[17] F. Verdoja and M. Grangetto, "Fast Superpixel-based Hierarchical Approach to Image Segmentation," in *International Conference on Image Analysis and Processing 2015 (ICIAP15)*, Genoa, Italy, Sept. 2015.

[18] 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.

[19] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electronics Letters*, vol. 44, no. 13, pp. 800–801, June 2008.

[20] ISO/IEC JTC 1/SC 29, *Information technology – Coded representation of picture and audio information – Progressive bi-level image compression*, International Organization for Standardization, 1st edition, 1993.

[21] I. Daribo, D. Florencio, and G. Cheung, "Arbitrarily shaped motion prediction for depth video coding using arithmetic edge coding," *Image Processing, IEEE Transactions on*, vol. 23, no. 11, pp. 4696–4708, Nov 2014.

[22] 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.

[23] R. Franzen, "Kodak lossless true color image suite," Jan. 2013, http://r0k.us/graphics/kodak/index.html.