GRAPH-BASED MOTION ESTIMATION AND COMPENSATION FOR DYNAMIC 3D POINT CLOUD COMPRESSION

Dorina Thanou[†]*, Philip A. Chou*[*]*, and Pascal Frossard*[†]

[†]Signal Processing Laboratory (LTS4), Swiss Federal Institute of Technology, Lausanne (EPFL)
[*]Microsoft Research, Redmond, WA

Email:{dorina.thanou, pascal.frossard}@epfl.ch, pachou@microsoft.com

ABSTRACT

This paper addresses the problem of motion estimation in 3D point cloud sequences that are characterized by moving 3D positions and color attributes. Motion estimation is key to effective compression of these sequences, but it remains a challenging problem as the temporally successive frames have varying sizes without explicit correspondence information. We represent the time-varying geometry of these sequences with a set of graphs, and consider 3D positions and color attributes of the points clouds as signals on the vertices of the graph. We then cast motion estimation as a feature matching problem between successive graphs. The motion is estimated on a sparse set of representative vertices using new spectral graph wavelet descriptors. A dense motion field is eventually interpolated by solving a graph-based regularization problem. The estimated motion is finally used for color compensation in the compression of 3D point cloud sequences. Experimental results demonstrate that our method is able to accurately estimate the motion and to bring significant improvement in terms of color compression performance.

Index Terms— 3D sequences, voxels, spectral graph wavelets, motion compensation

1. INTRODUCTION

Dynamic 3D scenes such as humans in motion are increasingly being captured by arrays of color plus depth video cameras [1]. The resulting captured geometry, unlike computer-generated geometry, has little explicit spatio-temporal structure, and is often represented by sequence of point clouds, where there may be different numbers of points in each frame, and no explicit association between points over time. Performing motion estimation, motion compensation, and compression of such data is a challenging task.

Unfortunately, the compression of 3D point cloud sequences has been largely overlooked so far in the literature. A few works have been proposed to compress static 3D point clouds. Some examples include the 2D wavelet transform based scheme of [2], and the octree based geometry compression algorithms of [3], [4], which focus on the compression of the 3D geometry positions. More recently, the authors in [5] have proposed to use a graph transform to remove the spatial redundancy for compression of the 3D point cloud attributes, with significant improvement over traditional methods. However, all the above methods consider each frame of the sequence independently, without exploiting the temporal redundancy that exists in geometry sequences. There does exist literature for compressing dynamic 3D meshes with either fixed connectivity and known correspondences (e.g., [6–10]) or varying connectivity (e.g., [11, 12]). However, there is only one work to our knowledge that exploits temporal and spatial redundancy of point cloud sequences [13]. The authors compress the geometry by comparing the octree data structure of consecutive point clouds and encoding their structural difference. Since their coding scheme is based on the set difference between octree stuctures and not the motion of the voxels, reducing the temporal correlation for coding the color attributes is not straightforward.

In this paper, we focus on the compression of the 3D color attributes and propose a novel motion estimation and compensation scheme that exploits temporal correlation in sequences of point clouds. We consider points as vertices in a graph $\mathcal{G}$, with edges between nearby vertices. Unlike a traditional polygonal mesh, this graph need not represent a surface. Attributes of each point n, including 3D position $p(n) = [x, y, z](n)$ and color components $c(n) = [r, g, b](n)$, are treated as signals residing on the vertices of the graph. As frames in the 3D point cloud sequences are correlated, the graph signals at consecutive time instants are also correlated. The estimation of the correlation is however a challenging task as the frames usually appear in different sizes and no explicit correspondence information is available in the sequence.

We propose a novel algorithm for motion estimation and compensation in 3D point cloud sequences. We cast motion estimation as a feature matching problem on dynamic graphs. In particular, we compute new local features at different scales with spectral graph wavelets (SGW) [14] for each node of the graph. Spectral features are stable to small perturbations of the edges or nodes of the graphs, and different instances of such features have been used successfully in graph matching problems [15] or in mesh segmentation and surface alignment problems [16]. We then match our SGW features in different graphs with a criteria that is based on the Mahalanobis distance and trained from the data. We first compute the motion on a sparse set of matching nodes, and we interpolate the motion of the other nodes of the graph by solving a new graph-based quadratic regularization problem, which promotes smoothness of the motion vectors on the graph in order to build a consistent motion field. We finally exploit the estimated motion information in the predictive coding of the color information, where we take benefit of the temporal redundancy by coding only the difference between the actual color information and the results of the motion compensation. We show by experimental results that the integration of our new motion compensation scheme in a state-of-the-art encoder [5] results in significant improvement in terms of rate-distortion compression performance of the color information in 3D point cloud sequences.

The rest of the paper is organized as follows. Section 2 first describes the representation of 3D point clouds using graphs, and introduces spectral graph wavelet descriptors. The motion estimation and composition scheme is presented in Section 3. Experimental results and conclusions are given in Section 4 and 5, respectively.

2. GRAPH-BASED REPRESENTATION OF 3D POINT CLOUDS

We represent the set of points in each frame using a weighted and undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$, where $\mathcal{V}$ and $\mathcal{E}$ represent the vertex and edge sets of $\mathcal{G}$. Graph-based representations are flexible and well adapted to data that lives on an irregular domain [17]. Each node in $\mathcal{V}$ corresponds to a point in the point cloud, while each edge in $\mathcal{E}$ connects neighbouring points. In our datasets, the point clouds are voxelized, that is, their 3D positions are quantized to a regular, axis-aligned, 3D grid having a given stepsize. Each quantization cell is called a voxel, a voxel containing a point is said to be occupied, and an occupied voxel is identified as a vertex in the graph. Two vertices are connected by an edge if they are 26-neighbors in the voxel grid, that is, if the distance between them is a maximum of one step along any axis. Thus the distance between connected pixels is either 1, $1/\sqrt{2}$ or $1/\sqrt{3}$ times the stepsize. The matrix W is a matrix of positive edge weights, with $W(i,j)$ denoting the weight of an edge connecting vertices i and j. This weight captures the connectivity pattern of nearby occupied voxels and are chosen to be inversely proportional to the distances between voxels, following the definition proposed in [5]. Finally, we compute the graph Laplacian operator defined as $\mathcal{L} = D - W$, where D is the diagonal degree matrix whose i^{th} diagonal element is equal to the sum of the weights of all the edges incident to vertex i [18]. It is a real symmetric matrix that has a complete set of orthonormal eigenvectors with corresponding nonnegative eigenvalues. We denote its eigenvectors by $\chi = [\chi^1, \chi^2, ..., \chi^N]$, and the spectrum of eigenvalues by $\Lambda := \left\{0 = \lambda_0 < \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_{(N-1)}\right\}$.

We consider the components of the 3D coordinates $p = [x, y, z]^T \in \mathbb{R}^{3\times N}$ and respectively the color attributes $c = [r, g, b]^T \in \mathbb{R}^{3\times N}$, as signals that reside on the vertices of the graph $\mathcal{G}$. These signals are used to define features on each node of the graph in the motion estimation process. A meaningful definition of a feature on a node of a graph requires a multi-resolution analysis of the graph signals with respect to that particular node. Due to the irregular graph domain, classical wavelet descriptors are however not applicable in these irregular settings. The analysis of signals defined on the vertices of an arbitrary weighted graph should rather be performed with specific tools such as the spectral graph wavelets (SGW) [14]. We therefore propose to construct SGW-based descriptors built on SGW-features $W_f(s,n)$ for each node in the graph. Such features are computed by taking the inner product between a given signal f and the graph wavelet $\psi_{s,n}$ of scale s placed at that particular node n, i.e.,

$$W_f(s,n) =< f, \psi_{s,n} >, \tag{1}$$

where the spectral graph wavelets are operator-valued functions of the graph Laplacian defined as

$$\psi_{s,n} = T_g^s \delta_n = \sum_{\ell=0}^{N-1} g(s\lambda_\ell)\chi_\ell^*(n)\chi_\ell. \tag{2}$$

The graph wavelets are determined by the choice of a generating kernel g, which acts as a band-pass filter in the spectral domain, and a scaling kernel h that acts as a lowpass filter. The scaling is defined in the spectral domain, i.e., the wavelet operator at scale s is given by $T_g^s = g(s\mathcal{L})$. Spectral graph wavelets are finally realized through localizing these operators via the impulse δ on a single vertex n.

3. MOTION ESTIMATION AND COMPENSATION IN 3D POINT CLOUD SEQUENCES

We use the spectral graph wavelets described in Sec. 2 to define spectral features at different resolutions and compute point-to-point correspondences between graphs of different frames by matching local invariant descriptors. We select a subset of matching nodes to define a sparse set of motion vectors that describe the temporal correlation in the sequence. A dense motion field is then interpolated from the sparse set of motion vectors in order to enable motion compensated color prediction.

3.1. Feature extraction and matching on graphs

For each node i of a graph $\mathcal{G}$, we define the following octant indicator function

$$o_{1,i}(j) = \mathbf{1}_{\{x(j)\geq x(i), y(j)\geq y(i), z(j)\geq z(i)\}}(j),$$

where $\mathbf{1}_{\{\cdot\}}(j)$ is the indicator function on node $j \in \mathcal{G}$, evaluated in the set $\{\cdot\}$ that depends on the 3D coordinates of the voxels. We consider all possible combinations of inequalities that results in a total of 2^3 indicator functions, i.e., $o_{k,i}(j)$, $k = [1, 2, ..., 8]$. These functions provide a notion of orientation of j with respect to i, which is clearly provided by the voxel grid.

We compute features based on both geometry and color information in each orientation. In particular, for each node i and each geometry and color component $f \in \{x, y, z, r, g, b\}$ in a specific orientation k, we compute the spectral graph wavelet coefficients

$$\phi_{i,s,o_{k,i},f} =< f \cdot o_{k,i}, \psi_{s,i} >, \tag{3}$$

where $k = 1, 2, ..., 8$, $s = s_1, ..., s_{max}$ and $\cdot$ denotes the pairwise product. The feature vector is the concatenation of these wavelet coefficients, including the features obtained from the scaling function, i.e., $\phi_i = \{\phi_{i,s,o_{k,i},f}\} \in \mathbb{R}^{8\times 6\times (s_{max}+1)}$.

Given two graphs $\mathcal{G}_t$, $\mathcal{G}_{t+1}$, each representing a frame in the 3D sequence, we use the above definition of features to find correspondences between vertices. We compute the matching score between two nodes $m \in \mathcal{G}_t, n \in \mathcal{G}_{t+1}$ as the Mahalanobis distance of the corresponding feature vectors, i.e.,

$$\sigma(m,n) = (\phi_m - \phi_n)^T P(\phi_m - \phi_n), \tag{4}$$

where P is a covariance matrix estimated from training features that are known to be in correspondence. We define as the best match for node $n \in \mathcal{G}_{t+1}$, the node $n^* \in \mathcal{G}_t$ with the minimum Mahalanobis distance, i.e.,

$$n^* = \underset{m\in\mathcal{G}_t}{\operatorname{argmin}}\, \sigma(m,n).$$

The choice of this distance metric is motivated by the combination of both geometry and color features, which are measured in different units. Hence, by learning the covariance matrix, we discover the relation between different feature components in each sequence.

We now compute motion vectors only on a sparse set of matching points, i.e., we take into consideration only accurate matches and ignore the rest. The selection of the sparse set of matching nodes is based on the intuition that a sampling of the nodes that covers all the 3D space can help later in interpolating the motion across all the nodes of the graph. We thus cluster the vertices of $\mathcal{G}_{t+1}$ in different regions and we keep only a representative vertex per region. Clustering is performed by applying K-means in the 3D coordinates of the nodes. K is usually set equal to the desirable sparse number of nodes. In order to avoid inaccurate matches, a representative vertex

is included in the sparse set only if its best score is smaller than a predefined threshold. Therefore, our sparse set of matching points tend to be accurate and well distributed spatially.

3.2. Computation of the motion vectors

Next, we study the interpolation of the dense motion field from the sparse set of matching nodes. Interpolation is basically done by treating the motion vector in each direction as a signal that varies smoothly on the graph. We estimate the entries of these signals for each matching pair (n^*, n), as $v(n) = p^{t+1}(n) - p^t(n^*)$, with p^t and p^{t+1} the 3D coordinate signals on $\mathcal{G}_t$ and $\mathcal{G}_{t+1}$, respectively. To allow some signal smoothing on these known entries, we model the matching score in their local neighborhood, with respect to node $n \in \mathcal{G}_{t+1}$. In particular, for each node $m \in \mathcal{G}_t$ that belongs to the two-hop neighborhood of n^* i.e., $m \in \mathcal{N}^2_{n^*}$, we express $\sigma(m, n)$ in terms of the best match score $\sigma(n^*, n)$, and the geometric distance of m from n^* using the second-order Taylor series expansion

$$\sigma(m,n) \approx \sigma(n^*,n) + (p^t(m) - p^t(n^*))^T M_n^{-1} (p^t(m) - p^t(n^*)).$$

We estimate M_n as the normalized covariance matrix of the 3D offsets,

$$M_n = \frac{1}{|\mathcal{N}^2_{n^*}|} \sum_{m \in \mathcal{N}^2_{n^*}} \frac{(p^t(m) - p^t(n^*))^T (p^t(m) - p^t(n^*))}{\sigma(m,n) - \sigma(n^*,n)}.$$

We define $Q = \begin{bmatrix} M_1^{-1} & \cdots & \mathbf{0}_{3\times 3} \\ \vdots & \ddots & \vdots \\ \mathbf{0}_{3\times 3} & \cdots & M_{N_t}^{-1} \end{bmatrix}$, where $M_l^{-1} = \mathbf{0}_{3\times 3}$ if node l does not belong to the sparse set of motion vectors.

Finally, we interpolate the dense set of motion vectors $\widehat{v}^*$ by smoothing the sparse set of motion vectors on the graph

$$\widehat{v}^* = \underset{v \in \mathbb{R}^{3N_t}}{\operatorname{argmin}} (v - v^*)^T Q (v - v^*) + \mu \sum_{i=1}^{3} (S_i v)^T \mathcal{L}_t (S_i v), \quad (5)$$

where $\{S_i\}_{i=1,2,3}$ is a selection matrix for each of the 3D components respectively, and $v^* = [v^*(1), v^*(2), \cdots, v^*(N_t)]^T \in \mathbb{R}^{3N_t}$, is the concatenation of the motion vectors, with $v^*(l) = \mathbf{0}_{3\times 1}$, if l does not belong to the sparse set. We note that the optimization problem consists of two terms: a fitting term that penalizes the excess matching score on the sparse set of matching nodes, and a regularization term that imposes smoothness of the motion vectors in each of the position components independently. Similar regularization techniques, that are based on the notion of smoothness of the graph Laplacian, have been widely using in the semi-supervised learning literature [19, 20]. The optimization problem is convex and it has a closed form solution given by

$$\widehat{v}^* = \left(Q + \mu \sum_{i=1}^{3} S_i^T \mathcal{L}_t S_i\right)^{-1} Q v^*, \quad (6)$$

which can be solved iteratively using MINRES-QLP [21] for efficiency on large systems.

3.3. Motion compensation for color prediction

We use the estimated motion vectors to warp the graph $\mathcal{G}_t$ to $\mathcal{G}_{t+1}$. In particular, the position of node m on the warped graph $\widetilde{\mathcal{G}}_t$ is estimated by using the corresponding position in $\mathcal{G}_t$ and the motion

$$\widetilde{p^t}(m) = p^t(m) + \widehat{v}^*(m).$$

(a) (b) (c) (d)

Fig. 1. Superimposition of the reference and target frame in two datasets ((a), (c)), and target frame and motion estimated reference frame ((b), (d)). Each small cube corresponds to a voxel in the motion compensated frame.

The color is transferred directly from $\mathcal{G}_t$ to $\widetilde{\mathcal{G}}_t$ i.e., $\widetilde{c^t}(m) = c^t(m)$. These values can then be used to predict the color values at the nodes of $\mathcal{G}_{t+1}$. For each $n \in \mathcal{G}_{t+1}$, we predict $\widehat{c^{t+1}}(n)$ by finding the nearest neighbors NN_n of $p^{t+1}(n)$ in terms of the 3D positions $\widetilde{p^t}$, and attributing to n their average color i.e.,

$$\widehat{c^{t+1}}(n) = \sum_{m \in \text{NN}_n} \frac{1}{|\text{NN}_n|} \widetilde{c^t}(m),$$

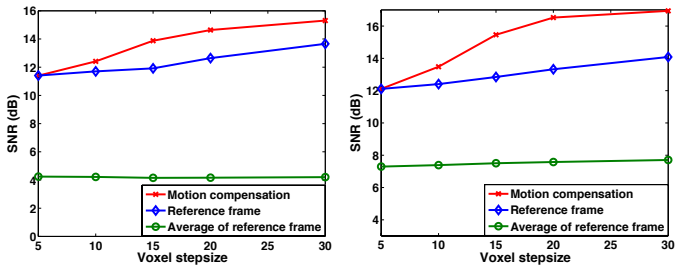
where $|\text{NN}_n|$ is the cardinality of NN_n, that is usually set to 3.

4. EXPERIMENTAL RESULTS

We illustrate the performance of our motion estimation and compensation scheme on two different datasets, i.e., the man sequence and the yellow dress, which were both constructed by a real-time high resolution sparse voxalization algorithm [1].

We first provide some illustrative results of the motion estimation performance. For each dataset, we select two consecutive frames, namely the reference and the target frame. For each frame, we voxelize the point cloud in the frame to a voxel stepsize that generates a set of approximately 8500 occupied voxels out of a total of 75000 initial 3D points with color attributes. The exact voxel number depends on the size of the actual frames. The graph for each frame is constructed as described in Section 2. We define spectral graph wavelets of 4 scales on these graphs, and for computational efficiency, we approximate them with Chebyshev polynomials of degree 30 [14]. We select the number of representative feature points to be around 500, which corresponds to fewer than 10% of the total occupied voxels, and we compute the sparse motion vectors on the corresponding nodes. We estimate the motion on the rest of the nodes by smoothing the motion vectors on the graph based on Eq.(5). In Fig. 1(a), we superimpose the reference and the target frame for the man sequence and in Fig. 1(c), the corresponding frames for the yellow dress sequence. Accordingly, in Figs. 1(b), 1(d) we superimpose the target frame and the voxel representation of the motion compensated reference frame. We observe that our algorithm is able to compensate quite accurately the motion. In particular, in both datasets the motion compensated reference frame, which is represented in a voxelized form, is close to the target frame.

In the next set of experiments, we use motion compensation for color prediction, as described in Section 3.3. That is, using the



(a) Man (b) Yellow dress

Fig. 2. Comparison of the color prediction performance obtained from the (i) motion compensated reference frame, (ii) reference frame, and (iii) average of the reference frame.

smoothed motion field, we warp the reference graph $\mathcal{G}_t$ to the target graph $\mathcal{G}_{t+1}$, and predict the color of each node in $\mathcal{G}_{t+1}$ as the average of the nearest nodes in $\mathcal{G}_t$. In Fig. (2), we illustrate the signal-to-noise ratio (SNR) of the color prediction for different voxel stepsizes defined as $\text{SNR} = 20\log_{10}\frac{\|\text{target color}\|}{\|\text{prediction error}\|}$. In particular, we compute the SNR by predicting the color of the target frame from the 3D positions of (i) the motion compensated reference frame, and (ii) the reference frame. For the sake of completeness, we show as well the SNR that is obtained by having as prediction the average of the color of the reference frame. The results are indicative of the dependency of the motion estimation on the voxel stepsize and the graph construction. When the stepsize is small, many nodes of the graph are isolated, which reduces the efficiency of the smoothing step that computes the dense motion field. As a result the color prediction error is similar to that obtained by predicting simply based on the reference frame. A more efficient construction of the graph could however improve the performance. On the other hand, when the motion is correctly estimated, motion compensation can significantly reduce the prediction error.

We finally use the prediction obtained from our motion estimation and compensation scheme to compress the color attributes of the target frame, for a voxel stepsize of 20. An overview of the predictive color coding structure is shown in Fig. 3. We assume that the set of occupied voxels (i.e., the geometry information) has already been coded according to the method described in [13], with the difference that coding is based on the set difference of the target frame and the *motion compensated* reference frame. We exploit the smoothness of the motion vectors on the graph, by coding them in the graph Fourier domain, which has been shown to be efficient in compressing smooth signals [22]. The graph Fourier coefficients are uniformly quantized, entropy coded with the RLGR entropy coder [23] and sent to the decoder. The set of occupied voxels of the target frame and the motion vectors can therefore be reconstructed at the decoder. The cost of the motion vectors is included in the geometry coding (to be described more fully elsewhere), which depending on the sequence results in either a small gain or a small penalty in the range of 0.01-0.3 bits per vertex over the coding rate of state-of-the-art geometry coding [13]. This essentially means that the coding of the motion vectors is basically transparent, and that almost no overhead has to be included for predictive coding of the color.

Compression of color attributes is thus obtained by coding the residual of the target frame with respect to the color prediction obtained with the scheme described in Section 3. Quantization and entropy coding of the residuals are performed using the recently introduced graph-based compression scheme of [5]. This algorithm

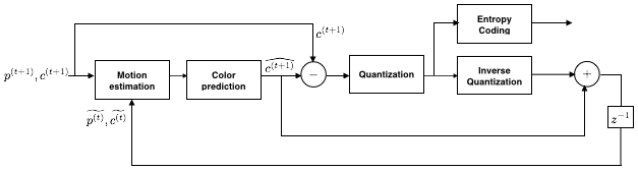


Fig. 3. Schematic overview of predictive color coding

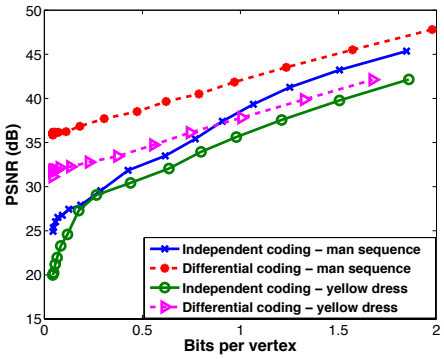


Fig. 4. Compression performance (dB) vr. bits per vertex for independent and differential coding of the target frame on both datasets.

removes the spatial redundancy of the voxels by applying the graph transform on small blocks of voxels. This step, combined with the differential coding step, exploits both temporal and spatial correlation for color coding. In our experiments, we choose small blocks of $16 \times 16 \times 16$ voxels. We measure the PSNR obtained for different coding rates of the color information, for both independent and differential coding. The results are shown in Fig. 4 for both datasets. We observe that differential coding provides a gain of approximately 10 dB at low bit rate, for the same number of bits per vertex, with respect to independent coding. Given the very small potential overhead introduced by the coding of the motion vectors, these results clearly confirm the benefit of motion compensation for color compression in 3D point cloud sequences.

5. CONCLUSIONS

In this paper, we have proposed a novel algorithm for motion estimation on 3D point cloud sequences. Our algorithm is based on the assumption that 3D models are representable by a sequence of weighted and undirected graphs and the geometry and the color of each model can be considered as graph signals residing on the vertices of the corresponding graphs. Correspondence between a sparse set of nodes in each graph is first determined by matching descriptors based on spectral features that are localized on the graph. The motion on the rest of the nodes is interpolated by exploiting the smoothness of the motion vectors on the graph. Motion compensation is finally used to perform color prediction. Experimental results have shown that the proposed method is efficient in estimating the motion and it eventually provides significant gain in compressing the color information with respect to independent coding of frame sequences.

6. ACKNOWLEDGEMENTS

The authors would like to thank Cha Zhang and Dinei Florêncio for their help with the color compression experiments.

7. REFERENCES

[1] C. Loop, C. Zhang, and Z. Zhang, "Real-time high-resolution sparse voxelization with application to image-based modeling," in *Proc. of the 5th High-Performance Graphics Conference*, New York, NY, USA, 2013, pp. 73–79.

[2] T. Ochotta and D. Saupe, "Compression of Point-Based 3D Models by Shape-Adaptive Wavelet Coding of Multi-Height Fields," in *Proc. of the First Eurographics Conference on Point-Based Graphics*, 2004, pp. 103–112.

[3] R. Schnabel and R. Klein, "Octree-based point-cloud compression," in *Symposium on Point-Based Graphics*, July 2006.

[4] Y. Huang, J. Peng, C. C. J. Kuo, and M. Gopi, "A generic scheme for progressive point cloud coding.," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 2, pp. 440–453, 2008.

[5] C. Zhang, D. Florêncio, and C. Loops, "Point cloud attribute compression with graph transform"," in *Proc. IEEE Int. Conference on Image Processing*, Paris, France, Sept 2014.

[6] J. Peng, Chang-Su Kim, and C. C. Jay Kuo, "Technologies for 3d mesh compression: A survey," *Journal of Vis. Comun. and Image Represent.*, vol. 16, no. 6, pp. 688–733, December 2005.

[7] J. Rossignac, "Edgebreaker: Connectivity compression for triangle meshes," *IEEE Trans. on Visualization and Computer Graphics*, vol. 5, no. 1, pp. 47–61, Jan. 1999.

[8] M. Alexa and W. Müller, "Representing animations by principal components," *Comput. Graph. Forum*, vol. 19, no. 3, pp. 411–418, Sept. 2000.

[9] L. Váša and V. Skala, "Geometry driven local neighborhood based predictors for dynamic mesh compression," *Comput. Graph. Forum*, vol. 29, no. 6, pp. 1921–1933, 2010.

[10] H. Q. Nguyen, P. A. Chou, and Y. Chen, "Compression of human body sequences using graph wavelet filter banks," in *Proc. IEEE Int. Conf. Acc., Speech, and Signal Process.*, Florence, Italy, May 2014, pp. 6152–6156.

[11] S.-R. Han, T. Yamasaki, and K. Aizawa, "Time-varying mesh compression using an extended block matching algorithm.," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 17, no. 11, pp. 1506–1518, Nov. 2007.

[12] S. Gupta, K. Sengupta, and A. A. Kassim, "Registration and partitioning-based compression of 3-d dynamic data.," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 13, no. 11, pp. 1144–1155, Nov. 2003.

[13] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *IEEE Int. Conference on Robotics and Automation*, Minnesota, USA, May 2012.

[14] D. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Appl. Comput. Harmon. Anal.*, vol. 30, no. 2, pp. 129–150, Mar. 2010.

[15] N. Hu, R. M. Rustamov, and L. Guibas, "Stable and informative spectral signatures for graph matching," in *IEEE Conf. on Comp. Vision and Pattern Recogn.*, June 2014.

[16] W. H. Kim, M. K. Chung, and V. Singh, "Multi-resolution shape analysis via non-euclidean wavelets: Applications to mesh segmentation and surface alignment problems," in *IEEE Conf. on Comp. Vision and Pattern Recogn.*, 2013, pp. 2139–2146.

[17] 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," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[18] F. Chung, *Spectral Graph Theory*, American Mathematical Society, 1997.

[19] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proc. Int. Conf. on Machine Learning*, Washington, DC, USA, 2003, vol. 13, pp. 912–919.

[20] D. Zhou, O. Bousquet, T .N. Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems*, 2004, vol. 16, pp. 321–328.

[21] S.-C. T. Choi and M. A. Saunders, "MINRES-QLP for symmetric and hermitian linear equations and least-squares problems," *ACM Trans. Math. Softw.*, vol. 40, no. 2, pp. 16, 2014.

[22] 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.

[23] H. S. Malvar, "Adaptive run-length / golomb-rice encoding of quantized generalized gaussian sources with unknown statistics," in *Data Compression Conference*, Mar. 2006.