COMPRESSION OF DYNAMIC 3D POINT CLOUDS USING SUBDIVISIONAL MESHES AND GRAPH WAVELET TRANSFORMS

*Aamir Anis**, Philip A. Chou*† *and Antonio Ortega**

*University of Southern California, Los Angeles, CA
† Microsoft Research, Redmond, WA
Email: aanis@usc.edu, pachou@microsoft.com, ortega@sipi.usc.edu

ABSTRACT

The advent of advanced acquisition techniques in 3D media applications has led to an increasing trend of capturing dynamic objects and scenes via 3D point cloud sequences. This form of data is composed of time-indexed frames, each consisting of a collection of points with position and color attributes. Compression of such datasets is challenging because of the lack of efficient techniques for exploiting spatial and temporal correlations between the attributes. In our approach, we create an intermediate high-resolution representation of the point clouds, using consistent subdivisional triangular meshes, that captures all the features of the underlying object or scene. This representation is easy to obtain, significantly simplifies motion compensation and allows us to design efficient wavelet transforms using the recently developed framework of Biorthogonal Graph Wavelet Filterbanks. Preliminary experiments show that our approach can be an effective compression technique for 3D point cloud sequences.

Index Terms— 3D point cloud sequences, subdivisional meshes, motion compensation, graph wavelet transform.

1. INTRODUCTION

Recently, 3D free viewpoint videos, captured using arrays of color and depth cameras, have become increasingly common in various applications such as virtual reality and immersive communication [1]. Such datasets are generally represented as a sequence of 3D point clouds, consisting of geometry and color information, that evolve over time. The enormous resolution required for capturing all the relevant features of the scene and achieving satisfactory visual quality makes their storage and transmission quite difficult. This necessitates the design of efficient compression techniques for such datasets. This can be particularly challenging since, in this case, adjacent frames have little spatio-temporal structure, i.e., there is little explicit correspondence between points across different frames. Moreover, the number of points in the cloud or occupied voxels, may also vary over time. These issues hinder the use of traditional techniques of motion estimation and compensation used to remove temporal redundancies in color and geometry. Further, designing efficient transforms to remove spatial redundancies is also cumbersome in this scenario because the data points generally occupy a sparse arbitrarily-shaped volume in 3D space that varies over time.

Compression of 3D point cloud sequences has received little attention to date. There have been works on compressing static point clouds, such as those using oct-trees [2–4], hierarchical point clustering [5], and the more recent graph transform-based approach [6]. These approaches can only be used to compress the sequence frame by frame, thereby not exploiting temporal redundancies to improve compression. The approach in [7, 8] alleviates this issue by representing point clouds through a set of graphs and matching features over them to perform motion estimation and predictive coding. However, the feature matching step can be quite intensive computationally. There also exist approaches to model point clouds via polygonal meshes before compression. These methods involve registration of a template mesh to track the point cloud sequence (see [9], references therein). Once a consistent dynamic mesh sequence is obtained, its attributes can be compressed using various existing methods [10, 11]. However, using these approaches to obtain and compress a high-resolution mesh representation can have a prohibitively high complexity, in addition to the overhead of encoding the mesh topology itself. Meshes with subdivisional topology can help reduce this overhead since only a coarse template mesh needs to be encoded. Further, this subdivisional structure is a natural analog of *upsampling* on meshes, and can be particularly exploited for implementing a novel class of wavelet transforms [12–17] used for compressing signals defined on graphs, of which meshes are a special case. A recent example is the work of [18], which compresses a subdivisional quad-mesh representation of a human body sequence using the recently developed framework of Biorthogonal graph wavelet filterbanks (*GraphBior* [17]). This work however is specific to human body sequences only, since it obtains the consistent mesh representation using a parametrized skinned-rig model of the human body.

In this paper, we propose a general method for compressing 3D point cloud sequences that efficiently captures and removes spatio-temporal correlations in the data. The key aspect of our formulation is to model the point clouds by an intermediate representation based on *consistently-evolving subdivisional triangular meshes* that have a resolution higher than each of the original point clouds. Essentially, our approach can be considered as a resampling of the underlying object or scene to obtain a new set of high-resolution consistently-evolving point clouds represented by the vertices of the meshes. This new representation allows us to obtain well-defined correspondences across frames that is integral for motion estimation and compensation. Further, we exploit the connectivity of the resampled points defined by the mesh topology to apply a graph-based wavelet transform for efficiently compressing the geometry and color attributes and their corresponding motion vectors.

The advantage of our method over previous schemes stems from the introduction of the subdivisional structure in the mesh topology. In particular, this constraint allows us to efficiently compute the mesh representation in a hierarchical manner from a very coarse base

This work was supported in part by NSF under grant CCF-1410009. The authors would like to thank the Microsoft HCap group for providing datasets used in the experiments.

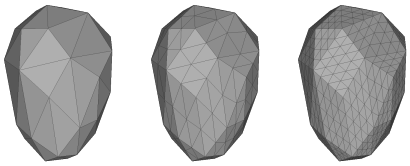

Fig. 1: Subdivision of a triangular mesh

mesh, which is the only information overhead required at the decoder's side, over the transform coefficients. This makes our method more general in comparison to [18], since we do not assume any parametric models for the meshes. Further, we also use a novel Laplacian-based regularizer while estimating the meshes, thereby making our procedure robust to noise and missing points. The subdivisional structure also allows us to obtain a sequence of bipartite graphs that facilitate the use of *GraphBior* [17] to compute the wavelet transform coefficients of the geometry and color attributes. Note that although the encoded sequence cannot yield an exact reproduction of the input point cloud sequence due to the resampling step involved in the intermediate representation, choosing a higher resolution ensures that all features of the underlying structure are captured and the rendered output is visually equivalent to the input. This is evident in our experiments from visual comparisons between the original point clouds and the obtained mesh representations.

2. PROPOSED COMPRESSION SCHEME

A point cloud frame at time t, consisting of N_t points, can be characterized as $\mathcal{P}^{(t)} = \{\mathbf{X}_p^{(t)}, \mathbf{X}_c^{(t)}\}$, where $\mathbf{X}_p^{(t)}, \mathbf{X}_c^{(t)} \in \mathbb{R}^{N_t \times 3}$ are attribute matrices, whose i^{th} rows $\mathbf{X}_p^{(t)}(i), \mathbf{X}_c^{(t)}(i) \in \mathbb{R}^{1\times 3}$ denote the position and color of the i^{th} point respectively. Additionally, we denote the surface normals for each point by the matrix $\mathbf{X}_n^{(t)} \in \mathbb{R}^{N_t \times 3}$, which can be obtained either through a standard estimation algorithm from the cloud or through the data acquisition technique itself.

2.1. Subdivisional triangular meshes

A subdivisional triangular mesh can be obtained by repeatedly subdividing a given base mesh (see Figure 1). Such a mesh consisting of V_k vertices and F_k faces at the k^{th} subdivision level, can be characterized as $\mathcal{M}^{(k)} = \{\mathbf{V}_p^{(k)}, \mathbf{V}_c^{(k)}, \mathbf{F}^{(k)}\}$, where $\mathbf{V}_p^{(k)}, \mathbf{V}_c^{(k)} \in \mathbb{R}^{V_k \times 3}$ are *vertex matrices* whose v^{th} rows $\mathbf{V}_p^{(k)}(v), \mathbf{V}_c^{(k)}(v) \in \mathbb{R}^{1\times 3}$ denote the position and the color of the v^{th} vertex respectively, and $\mathbf{F}^{(k)} \in \mathbb{N}^{F_k \times 3}$ is the *face matrix* whose i^{th} row $\mathbf{F}^{(k)}(i) \in \mathbb{N}^{1\times 3}$ indicates the indices of the vertices contained in the i^{th} face. The face matrix $\mathbf{F}^{(k)}$ captures the topology of the mesh, and can be used to define the unweighted vertex-to-vertex adjacency matrix $\mathbf{A}^{(k)}$ and the Laplacian $\mathbf{L}^{(k)}$ in the usual sense. In addition, one can also estimate the normal at each vertex using a combination of the adjacent face normals weighted by the corresponding triangle areas. We represent these vertex normals through the matrix $\mathbf{V}_n^{(k)} \in \mathbb{R}^{M_k \times 3}$. Note that $\mathbf{V}_n^{(k)}(v)$ equals the sum of pair-wise cross-products of the adjacent vertex positions $\mathbf{V}_p^{(k)}(u), \forall u \in \mathcal{N}_1(v)$, where $\mathcal{N}_1(v)$ denotes the 1-neighborhood of vertex v.

We use the Loop subdivision rules for subdividing and computing the attributes of the resultant mesh [19]. In this scheme, each triangle is subdivided into four by introducing new vertices at the midpoints of its sides. Thus, the resolution of the meshes increases as we go to higher levels of subdivision. The new vertex attributes $\mathbf{V}_p^{(k+1)}, \mathbf{V}_c^{(k+1)}$ are computed from $\mathbf{V}_p^{(k)}, \mathbf{V}_c^{(k)}$ using a simple linear transformation $\mathbf{T}^{(k)} \in \mathbb{R}^{M_{k+1} \times M_k}$ with entries depending on the degree (i.e., number of neighbors) of each vertex (see [19]).

2.2. Representation scheme for point cloud sequences

In our representation model, the point cloud sequence for $t = 1, 2, \ldots$ can be modeled by subdivisional triangular meshes $\mathcal{M}^{(k,t)} = \{\mathbf{V}_p^{(k,t)}, \mathbf{V}_c^{(k,t)}, \mathbf{F}^{(k,t)}\}$ at different levels. For a given base mesh $\mathcal{M}^{(0,t)}$, the final level $k = K$ used for representation is chosen such that the mesh has higher resolution than the point cloud. Roughly, the number of vertices $V_{K,t}$ in the mesh must exceed the number of points N_t in the cloud for each t. We follow a scheme similar to current video coding techniques, where frames in the sequence are classified either as reference frames that are intra-coded, or as predicted frames that are inter-coded. Thus, for the purpose of motion estimation, we require that the topology of the predicted frames must be identical to that of the last reference frame, i.e., the number of vertices $V_{k,t}$, faces $F_{k,t}$, and the face matrices $\mathbf{F}^{(k,t)}$ must match for all k and t in that set of frames. When this condition is true, one can define the motion fields for a given frame to be equal to the difference in vertex attributes of that frame and the previous frame's representation. Specifically, for a frame t at level k, the position and color motion fields can be computed as $\mathbf{M}_p^{(k,t)} = \mathbf{V}_p^{(k,t)} - \mathbf{V}_p^{(k,t-1)}$ and $\mathbf{M}_c^{(k,t)} = \mathbf{V}_c^{(k,t)} - \mathbf{V}_c^{(k,t-1)}$, whenever the frames t and $t-1$ have equivalent topology.

2.3. Estimation of mesh attributes

In order to estimate the attributes $\mathbf{V}_p^{(k,t)}, \mathbf{V}_c^{(k,t)}$ of a mesh $\mathcal{M}^{(k,t)}$ that models the point cloud $\mathcal{P}^{(t)}$ as closely as possible, we first need a way to determine how *close* a given mesh is to the point cloud. To this end, we introduce a *correspondence* mapping $\pi^{(k,t)} : \mathbb{N} \to \mathbb{N}$ from points in $\mathcal{P}^{(t)}$ to the vertices in $\mathcal{M}^{(k,t)}$. $\pi^{(k,t)}(i)$ indicates the vertex to which the i^{th} point in the cloud is *assigned* and can be obtained using a metric based on geometric features, i.e., the positions (and if available, the surface normals) of the points. Specifically, let us define the following metric for determining geometric proximity between a point i in $\mathcal{P}^{(t)}$ and a vertex v in $\mathcal{M}^{(k,t)}$:

$$D^{(k,t)}(i,v) = \alpha_p \|\mathbf{X}_p^{(t)}(i) - \mathbf{V}_p^{(k,t)}(v)\|_F^2 + \alpha_n \|\mathbf{X}_n^{(t)}(i) - \mathbf{V}_n^{(k,t)}(v)\|_F^2, \quad (1)$$

where α_p and α_n are scalar weights to control the contribution of positions and surface normals respectively. Then, for the point i, the corresponding vertex is given as

$$\pi^{(k,t)}(i) = \underset{v}{\operatorname{argmin}}\ D^{(k,t)}(i,v). \quad (2)$$

Let us define a correspondence matrix $\mathbf{\Pi}^{(k,t)} \in \{0,1\}^{N_t \times V_{k,t}}$ with $0/1$ entries, i.e., $\mathbf{\Pi}_{ij}^{(k,t)} = 1$ if $\pi^{(k,t)}(i) = j$, and 0 otherwise. Then, the *deviations* in geometry and color between a point cloud $\mathcal{P}^{(t)}$ and its mesh representation $\mathcal{M}^{(k,t)}$ are given by

$$\delta_p^{(k,t)} = \|\mathbf{\Pi}^{(k,t)}\mathbf{V}_p^{(k,t)} - \mathbf{X}_p^{(t)}\|_F^2, \quad (3)$$

$$\delta_c^{(k,t)} = \|\mathbf{\Pi}^{(k,t)}\mathbf{V}_c^{(k,t)} - \mathbf{X}_c^{(t)}\|_F^2. \quad (4)$$

Based on the equations above, one can conclude that finding the best mesh model at level $k = K$ involves estimating $\mathbf{V}_p^{(K,t)}$, $\mathbf{V}_c^{(K,t)}$, $\mathbf{\Pi}^{(K,t)}$ that minimize the geometry and color deviations $\delta_p^{(K,t)}$ and

$\delta_c^{(K,t)}$ independently. Since this problem can be undetermined, we grow this mesh hierarchically from $k = 0$ to $k = K$ along with the introduction of certain regularization terms.

Reference frames: We begin by describing the mesh estimation procedure for reference frames. The first step involves creating a "skeletal mesh model" for the point cloud, i.e., a coarse base mesh $\mathcal{M}^{(0,t)} = \{\mathbf{V}_p^{(0,t)}, \mathbf{F}^{(0,t)}\}$ using Poisson Surface Reconstruction with a low oct-tree depth [20]. This procedure is fast and generates a coarse, colorless triangular mesh that approximately captures the underlying geometrical structure of the point cloud. The color attributes $\mathbf{V}_c^{(0,t)}$ can be initialized to a neutral value, such as uniform grayscale. Once we have the base mesh, we assign the correspondences $\pi^{(0,t)}$ using the global search procedure in (2). This assignment is computationally feasible since the base mesh has a very low resolution for $k = 0$. To minimize the deviations (3) and (4) at each level k, we alternately update the vertex attributes $\mathbf{V}_p^{(k,t)}$, $\mathbf{V}_p^{(k,t)}$ and the correspondences $\pi^{(k,t)}$, which we describe next.

Given a current state of the mesh $\mathcal{M}^{(k,t)}$ and correspondences $\pi^{(k,t)}$, we perform updates on the geometry and color attributes as $\mathbf{V}_p^{(k,t)} \leftarrow \mathbf{V}_p^{(k,t)} + \mathbf{\Delta}_p^{(k,t)}$ and $\mathbf{V}_c^{(k,t)} \leftarrow \mathbf{V}_c^{(k,t)} + \mathbf{\Delta}_c^{(k,t)}$, where $\mathbf{\Delta}_p^{(k,t)}, \mathbf{\Delta}_c^{(k,t)} \in \mathbb{R}^{V_{k,t} \times 3}$ are update matrices given by

$$\mathbf{\Delta}_p^{(k,t)} = \underset{\mathbf{\Delta}}{\operatorname{argmin}} \Big[\|\mathbf{\Pi}^{(k,t)}(\mathbf{V}_p^{(k,t)} + \mathbf{\Delta}) - \mathbf{X}_p^{(t)}\|_F^2 + \lambda_p^{(k)} \operatorname{Tr}(\mathbf{\Delta}^T \mathbf{L}^{(k)} \mathbf{\Delta}) \Big], \quad (5)$$

$$\mathbf{\Delta}_c^{(k,t)} = \underset{\mathbf{\Delta}}{\operatorname{argmin}} \Big[\|\mathbf{\Pi}^{(k,t)}(\mathbf{X}_c^{(k,t)} + \mathbf{\Delta}) - \mathbf{X}_c^{(t)}\|_F^2 + \lambda_c^{(k)} \operatorname{Tr}(\mathbf{\Delta}^T \mathbf{L}^{(k)} \mathbf{\Delta}) \Big]. \quad (6)$$

$\lambda_p^{(k)}, \lambda_c^{(k)}$ are the regularization parameters for position and color respectively. The equations above have a closed form solution, which involves solving the following linear systems:

$$\left({\mathbf{\Pi}^{(k,t)}}^T \mathbf{\Pi}^{(k,t)} + \lambda_p^{(k)} \mathbf{L}^{(k)} \right) \mathbf{\Delta}_p^{(k,t)} = {\mathbf{\Pi}^{(k,t)}}^T \mathbf{R}_p^{(k,t)}, \quad (7)$$

$$\left({\mathbf{\Pi}^{(k,t)}}^T \mathbf{\Pi}^{(k,t)} + \lambda_c^{(k)} \mathbf{L}^{(k)} \right) \mathbf{\Delta}_c^{(k,t)} = {\mathbf{\Pi}^{(k,t)}}^T \mathbf{R}_c^{(k,t)}, \quad (8)$$

where $\mathbf{R}_p^{(k,t)} = \mathbf{X}_p^{(t)} - \mathbf{\Pi}^{(k,t)} \mathbf{V}_p^{(k,t)}$ denotes the position residuals, and $\mathbf{R}_c^{(k,t)} = \mathbf{X}_c^{(t)} - \mathbf{\Pi}^{(k,t)} \mathbf{V}_c^{(k,t)}$ denotes the color residuals. Note that the matrix ${\mathbf{\Pi}^{(k,t)}}^T \mathbf{\Pi}^{(k,t)}$ is diagonal with entries equal to the number of assigned points for each vertex. Thus, in the absence of regularization, the updates for the mesh attributes essentially involve averaging the residuals with respect to the assigned correspondence points in the cloud. The mesh Laplacian regularizer $\operatorname{Tr}(\mathbf{\Delta}^T \mathbf{L}^{(k)} \mathbf{\Delta})$ ensures that attributes of nearby points evolve smoothly in a similar fashion. This is useful if there are vertices in the mesh with no corresponding points in the cloud, which is likely for meshes at higher levels of subdivision. The updates in this case, are interpolated from the updates of the neighbors. Since a smoother representation results in a lower bit rate, the regularizer can also be regarded as a rate term added to the distortion metric during minimization.

Next, we update the correspondences for the updated mesh. Since, at higher levels, the exhaustive search in (2) can be computationally intensive, we resort to a local update as follows:

$$\pi^{(k,t)}(i) \leftarrow \underset{v \in \left\{\pi^{(k,t)}(i)\right\} \cup \mathcal{N}_d\left(\pi^{(k,t)}(i)\right)}{\operatorname{argmin}} D^{(k,t)}(i, v) \quad (9)$$

For a point i, (9) essentially restricts the search domain to an inclusive d-depth local neighborhood ($d \geq 1$) of the current vertex $\pi^{(k,t)}(i)$ assigned to it.

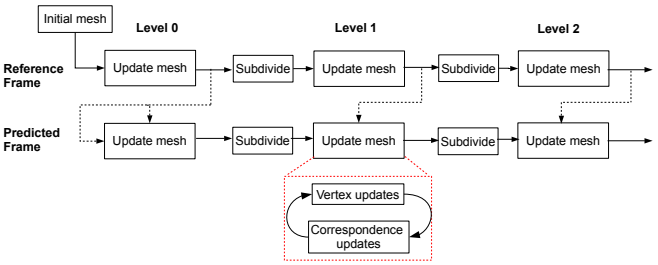

Fig. 2: Schematic flow of the mesh estimation procedure.

After a few iterations of vertex and correspondence updates until satisfactory convergence, the mesh $\mathcal{M}^{(k,t)}$ is subdivided to obtain $\mathcal{M}^{(k+1,t)}$ using the Loop subdivision rule [19]. The new correspondences $\pi^{(k+1,t)}$ can be obtained simply by performing the local search procedure stated in (9). The mesh estimation and subdivision processes are repeated until we reach the desired level of detail.

Predicted frames: The mesh representation for predicted frames is obtained in a similar manner. Here, we begin with the lowest level mesh representation of the previous frame as the coarse base mesh and repeat the estimation and subdivision steps to reach the desired level of representation. To obtain $\mathcal{M}^{(K,t)}$ for a predicted frame $\mathcal{P}^{(t)}$, we use the 0^{th} level mesh estimate of the previous frame $\mathcal{M}^{(0,t-1)}$ as the coarse base mesh. The correspondences are once again assigned through the global search step in (2). Note that the initial base mesh may not be perfectly aligned geometrically with the point cloud, leading to some erroneous correspondence assignments. Increasing the weight α_n of the normals in the geometric proximity metric (1) may help in such situations.

In order for motion estimation to be accurate, we need to consider texture misalignment in certain patches of the mesh due to lateral shifts. An example of such a scenario would be a rolling striped ball. In our approach, this can be handled by modifying the correspondence update step to include color matching with the previous frame. Specifically, we add an extra term to the proximity metric $D^{(k,t)}(i, v)$ in (1) to obtain

$$D_c^{(k,t)}(i, v) = D^{(k,t)}(i, v) + \alpha_c \|\mathbf{V}_c^{(k,t)}(\pi^{(k,t)}(i)) - \mathbf{V}_c^{(k,t-1)}(v)\|_F^2, \quad (10)$$

where $\mathbf{V}_c^{(k,t-1)}$ is the best level-k mesh representation for the previous frame. The correspondences are then updated using $D_c^{(k,t)}(i, v)$ in (9). This modification basically assigns points to vertices while taking into account the similarity in colors between vertices of the adjacent frames. A point is more likely to be reassigned to a vertex whose color in the previous frame is closer to the currently assigned vertex, resulting in gradual texture-based alignment across frames.

Finally, to ensure that the motion fields are smooth for compression gains, we add $\mu_p^{(k)} \operatorname{Tr}((\mathbf{M}_p^{(k,t)} + \mathbf{\Delta})^T \mathbf{L}^{(k)} (\mathbf{M}_p^{(k,t)} + \mathbf{\Delta})$ and $\mu_c^{(k)} \operatorname{Tr}((\mathbf{M}_c^{(k,t)} + \mathbf{\Delta})^T \mathbf{L}^{(k)} (\mathbf{M}_c^{(k,t)} + \mathbf{\Delta})$ to the objective functions of the vertex updates in (5) and (6) respectively, where $\mathbf{M}_p^{(k,t)} = \mathbf{V}_p^{(k,t)} - \mathbf{V}_p^{(k,t-1)}$ and $\mathbf{M}_c^{(k,t)} = \mathbf{V}_c^{(k,t)} - \mathbf{V}_c^{(k,t-1)}$ are the position and color motion vectors between the current vertex attributes and the attributes of the previous frame. These additional regularization terms, at each iteration, tend to make the updated motion fields $\mathbf{M}_p^{(k,t)} + \mathbf{\Delta}$ and $\mathbf{M}_c^{(k,t)} + \mathbf{\Delta}$ smooth over the mesh, thereby making the final motion fields smooth. The resulting optimization problems, as before, also admit closed form solutions. A schematic overview of the complete mesh estimation procedure is illustrated in Figure 2.

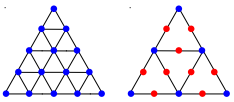

Fig. 3: Obtaining $\mathcal{G}^{(k)}$ (right) from $\mathcal{M}^{(k)}$ (left)

2.4. Compression using graph wavelet transforms

We now describe briefly how our intermediate mesh representation can be used to design graph wavelet transforms for compression. The key property of our representation that we exploit is the subdivisional structure that, with certain approximations, allows us to create a natural sequence of unweighted bipartite graphs, similar to the scheme in [18]. These graphs are suitable for implementing *GraphBior* filterbanks [17] that provide a vertex-frequency localized transform with several favorable properties such as critical sampling, perfect reconstruction, near orthogonality and compact support in the vertex domain [17, Table I]. The design is applicable for bipartite graphs, while one has to resort to bipartite subgraph decompositions for arbitrary graphs.

In order to implement the filterbanks, we first obtain an unweighted bipartite graph $\mathcal{G}^{(k)}$ for each level of the subdivisional triangular mesh $\mathcal{M}^{(k)}$ by ignoring certain edges in the mesh. Let us consider the subdivision of $\mathcal{M}^{(k-1)}$ to obtain $\mathcal{M}^{(k)}$. This step involves introducing new vertices at the midpoints of all triangles in $\mathcal{M}^{(k-1)}$ along with new edges that connect them. However, if we ignore the new edges, the resultant graph $\mathcal{G}^{(k)}$ is bipartite, since each new vertex is connected to only a pair existing vertices as shown in Figure 3. Note that $\mathcal{G}^{(k)}$ is a subgraph of the mesh $\mathcal{M}^{(k)}$ with the same number of vertices. In addition, $\mathcal{G}^{(k-1)}$ is a subgraph of $\mathcal{G}^{(k)}$, and thus we get the following nested sequence of bipartite graphs $\mathcal{G}^{(0)} \subset \mathcal{G}^{(1)} \subset \cdots \subset \mathcal{G}^{(K-1)} \subset \mathcal{G}^{(K)}$, over which we can apply the zero-DC *GraphBior* filterbank. For the 0^{th} level, we simply apply a full Graph Fourier Transform (GFT) [21]. After transforming the vertex attributes for reference frames and motion vectors for predicted frames, the coefficients are uniformly quantized with different step sizes at each level to balance the quantization noise, since the filterbanks are not perfectly orthogonal. The quantized coefficients are then entropy coded using the RLGR coder [22]. We plan to investigate context-adaptive entropy coding schemes that exploit mesh connectivity in future work.

3. EXPERIMENTS

We evaluate our compression scheme on a 204-frame point cloud sequence capturing various motions performed by a person. Each frame of this dataset has between 175,000 and 225,000 voxelized points. Starting from frame 1, every 30^{th} frame is designated as a reference frame (a better alternative would be to consider the mesh fitting error with respect to the previous frame). A base mesh consisting of 720 to 750 vertices (1440 to 1500 faces) is obtained for these reference frames using Poisson Surface Reconstruction. In Figure 4, we illustrate the original point clouds and the obtained subdivisional mesh representations for a reference frame (frame number 31) and a predicted frame (frame number 32). We observe that our mesh estimation procedure produces a detailed representation of the data. However, we observed folds or "webbings" in the mesh in certain regions of high geometrical detail (for example, the fingers of the hands), because the base mesh is very coarse for these regions. These folds do not have any corresponding points from the original point cloud and can be removed through simple post-processing

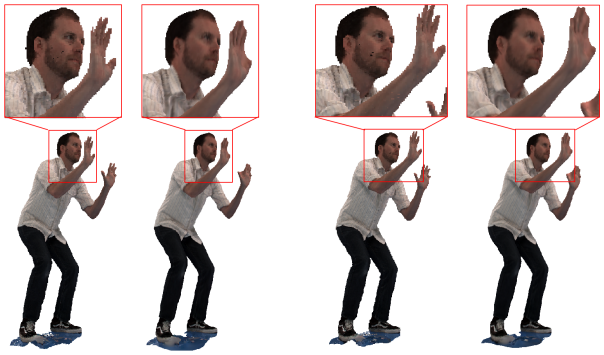

Fig. 4: Original point clouds (left) and obtained mesh representations (right) for (a) reference frame 31 and (b) predicted frame 32.

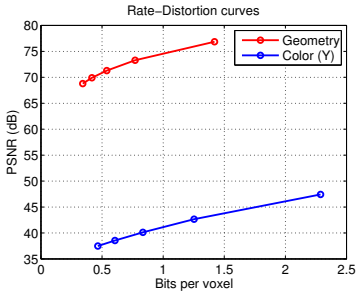

Fig. 5: Rate-distortion curve for geometry and color (Y-component) for the Man sequence.

which may impose a slight penalty on the compression efficiency.

We also perform preliminary compression experiments on this sequence. We compress the vertex positions and colors of the estimated subdivisional meshes using *GraphBior* (6,6) filterbanks [17] and the RLGR coder for different quantization step sizes. Figure 5 shows the rate distortion curves obtained. Note that the distortion is computed between the estimated mesh and its transformed, quantized version. Since our intermediate mesh representation is a resampling of the original point cloud, comparing the quality of the fit mesh, or the decoded output against the input point cloud, in terms of distortion, is not straightforward. This requires defining a metric that measures the distance between two point clouds, while taking into account resampling. We plan to investigate this as part of future work, and provide comparisons with existing compression schemes.

4. CONCLUSION

In this paper, we provide a framework for compression of 3D point cloud sequences. Our approach involves representing sets of frames by a consistently-evolving high-resolution subdivisional triangular mesh. This representation helps us facilitate efficient implementations of motion estimation and graph wavelet transforms. The subdivisional structure plays a crucial role in designing a simple hierarchical method for efficiently estimating these meshes, and the application of Biorthogonal Graph Wavelet Filterbanks for compression. Preliminary experimental results show promising performances of both the estimation and the compression steps, and we believe this work shall open new avenues of research in this emerging field.

5. REFERENCES

[1] C. Loop, C. Zhang, and Z. Zhang, "Real-time high-resolution sparse voxelization with application to image-based modeling," in *Proceedings of the 5th High-Performance Graphics Conference*, New York, NY, USA, 2013, HPG '13, pp. 73–79, ACM.

[2] R. Schnabel and R. Klein, "Octree-based point-cloud compression," in *Symposium on Point-Based Graphics 2006*, M. Botsch and B. Chen, Eds. July 2006, Eurographics.

[3] Y. Huang, J. Peng, C.-C.J. Kuo, and M. Gopi, "A generic scheme for progressive point cloud coding," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 2, pp. 440–453, March 2008.

[4] J. Kammerl, N. Blodow, R.B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, May 2012, pp. 778–785.

[5] Fan Y., Y. Huang, and J. Peng, "Point cloud compression based on hierarchical point clustering," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2013 Asia-Pacific*, Oct 2013, pp. 1–7.

[6] C. Zhang, D. Florencio, and C. Loop, "Point cloud attribute compression with graph transform," in *Image Processing (ICIP), 2014 IEEE International Conference on*, Oct 2014, pp. 2066–2070.

[7] D. Thanou, P. A. Chou, and P. Frossard, "Point cloud attribute compression with graph transform," in *Image Processing (ICIP), 2015 IEEE International Conference on*, Sep 2015.

[8] D. Thanou, P. A. Chou, and P. Frossard, "Graph-based compression of dynamic 3D point cloud sequences," *CoRR*, vol. abs/1506.06096, 2015.

[9] G.K.L. Tam, Z.-Q. Cheng, Y.-K. Lai, F.C. Langbein, Y. Liu, D. Marshall, R.R. Martin, Xian-Fang Sun, and P.L. Rosin, "Registration of 3D point clouds and meshes: A survey from rigid to nonrigid," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 7, pp. 1199–1217, July 2013.

[10] J. Peng, C.-S. Kim, and C.-C.J. Kuo, "Technologies for 3D mesh compression: A survey," *Journal of Visual Communication and Image Representation*, vol. 16, no. 6, pp. 688 – 733, 2005.

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

[12] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, March 2003, vol. 3, pp. 1848–1857 vol.3.

[13] R.R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 53 – 94, 2006, Special Issue: Diffusion Maps and Wavelets.

[14] W. Wang and K. Ramchandran, "Random multiresolution representations for arbitrary sensor network graphs," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, May 2006, vol. 4, pp. IV–IV.

[15] 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.

[16] S.K. Narang and A. Ortega, "Perfect reconstruction two-channel wavelet filter banks for graph structured data," *Signal Processing, IEEE Transactions on*, vol. 60, no. 6, pp. 2786–2799, June 2012.

[17] S.K. Narang and A. Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *Signal Processing, IEEE Transactions on*, vol. 61, no. 19, pp. 4673–4685, Oct 2013.

[18] H.Q. Nguyen, P.A. Chou, and Y. Chen, "Compression of human body sequences using graph wavelet filter banks," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 6152–6156.

[19] C. Loop, "Smooth subdivision surfaces based on triangles," M.S. thesis, University of Utah, 1987.

[20] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, Aire-la-Ville, Switzerland, Switzerland, 2006, SGP '06, pp. 61–70, Eurographics Association.

[21] 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, May 2013.

[22] H.S. Malvar, "Adaptive run-length/golomb-rice encoding of quantized generalized gaussian sources with unknown statistics," in *Data Compression Conference, 2006. DCC 2006. Proceedings*, March 2006, pp. 23–32.