A Basic Geometry Driven Mesh Coding Scheme with Surface Simplification for 3DTI

Rufael Mekuria, Pablo Cesar
Centrum Wiskunde Informatica, Netherlands
Rufael.mekuria@cwi.nl p.s.cesar@cwi.nl

1. Introduction

With the rise of consumer grade depth camera's and available computational power, the computer vision and image processing communities are developing systems that can reconstruct full 3D Mesh geometry consisting of vertex and connectivity data representing a human or animal user on-the-fly. To enable interactive communication (i.e. 3D Tele-immersion: 3DTI) with this representation, efficient transmission and compression schemes are needed. While much research on the transport/compression of 3D representations focuses on multi-view video (MVV) based on multiple image views, the 3D Mesh representation has benefits for 3D immersive communications. Firstly, 3D Mesh **rendering** is directly supported in modern OpenGL compliant graphics cards. Secondly, **composite rendering (composition)** with synthetic 3D content in games and virtual worlds is much easier compared to image based MVV representations. Also, advanced **3D audio and lighting rendering** schemes can take advantage of the explicit surface position of the mesh. Thirdly, the mesh representation is common in **video games** and **virtual worlds** where controlled agents can use the geometric information to control their **AI**. Lastly, **compression efficiency** could be an advantage, as especially time-consistent 3D dynamic meshes can be compressed very efficiently, possibly reducing the bandwidth requirements over image based MVV.

Nevertheless, existing mesh transmission and compression techniques have not been designed for real-time streaming and transmission of live reconstructed content. Instead, they focus on traditional synthetic graphics and animation content, which has different requirements as all content is authored offline. Based on the large industrial potential of live 3D Tele-Immersive mesh coding, the MPEG AhG on 3D Graphics is currently exploring this new direction, gathering industry requirements and benchmarking existing technologies. In this paper we contribute by introducing a full real-time geometry driven mesh codec focusing on geometry guided connectivity coding. In Section 2 we first present related work. In section 3 we introduce the specific challenges and requirements for 3DTI. We discuss the implementation based on octree composition, surface simplification and connectivity coding in section 4. In Section 5 we present an evaluation and rate-distortion comparison of the codec to the MPEG-4 standard.

2. Related Work

Dynamic time-consistent geometry, i.e. sequences of meshes where only the geometric coordinates change (like animations), can generally be compressed well as the fixed connectivity over time allows direct coding of frame by frame vertex differences (for example the MPEG-4 FAMC [1]). In the case of live 3D capturing systems that reconstruct 3D meshes from multiple depth images, connectivity can also change over time, introducing the need for time-varying mesh compression (TVM). A standardization contribution to MPEG-3DG in [2] proposes a system for time-varying mesh compression based on the inter-frame prediction between MPEG-4 [3] static mesh coded frames. The method utilizes separately acquired skeleton motion and skinning for the prediction. Their results show some improvements compared to using the MPEG-4 codec at lower QP's. A limitation of the approach in [2], is that temporal redundancy of the geometry information over time cannot be exploited directly, as the geometry coding is guided by the connectivity. Also, based on the single rate MPEG-4 codec it is harder to implement spatial scalability in the bit-stream. To overcome these limitations, we propose a purely geometry driven approach where connectivity coding is guided by the geometric data. The geometry coding is based on an octree representation comparable to [4] which has been considered favorable for the densely sampled surfaces that TVM's represent. We further exploit the octree structure to enable basic scalability and mesh surface simplification. Also, we envision that the octree structure can in a next stage be used to exploit temporal redundancies better. In the current implementation we introduce the geometry-guided connectivity coding scheme that is simpler compared to the method from [4], making it more suitable for 3DTI. Together with an existing point cloud codec it is integrated into a complete geometry driven 3D Mesh Codec.

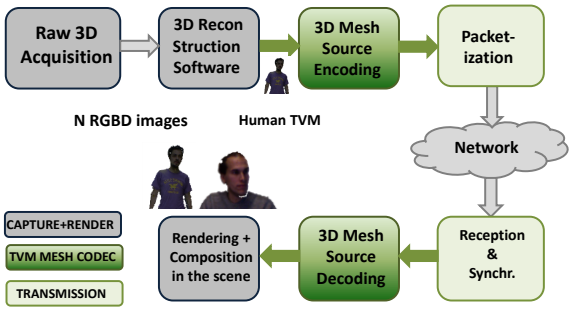

Figure 1 Mesh Based 3DTI Pipeline Example

3. 3DTI Based on Reconstructed Geometry Requirements

In Figure 1 we show a typical 3D tele-immersive pipeline based on live reconstructed geometry. The key difference compared to image based MVV is that instead of sending the raw captured 3D depth/image data the 3D reconstruction software reconstructs the full 3D surface represented as a 3D mesh $M(V,C)$ with vertex data V and connectivity data C. So instead, $M(V,C)$ is subsequently compressed and transmitted. Specifically challenging in this case compared to traditional 3D Geometry compression/streaming systems for 3D Graphics is that low encoding/decoding complexity is needed and that imperfect input data should be handled gracefully. To establish a common understanding of the requirements in this use case that is not just about compression efficiency, the MPEG AhG on 3DG has developed a document describing the use case and coding system requirements for 3DTI [5] .

4. Geometry Driven Mesh Codec

Figure 2 illustrates the geometry driven mesh coding scheme. First we encode the geometry (vertex information, possibly with colors) with an existing octree based point cloud codec that encodes only vertex/color data and is available in PCL [6] and described in [7]. This codec partitions all vertex data in voxels in a 3D bounding box that can be indexed with discrete positional indices $v_{octree}\{x,y,z\}$, we call this voxel space V_{octree}. The octree composition is also used to simplify the dense input mesh by down sampling on a per voxel basis. We only recover the voxel leaf center at the receiver instead of all enclosed vertices. This simplification allows a large data reduction with fine-grained quality control by tuning the voxel size depending on quality/bandwidth requirements.

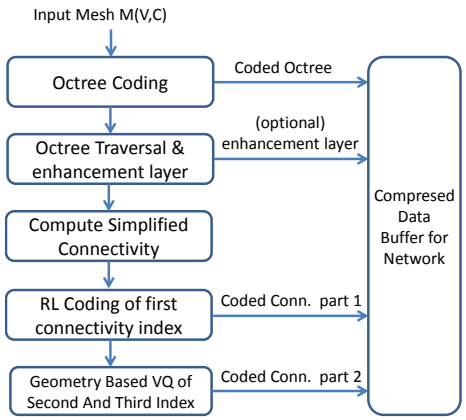

Figure 2 Basic Geometry Driven Mesh Codec

In the second stage, we traverse the octree, storing the indices I_{simp} that index a voxel leaf with a single integer. I_{simp} contains all vertex indices for the simplified connectivity. During this traversal we optionally compute an enhancement layer that refines the position of the voxel center based on the original vertex points based on a weighted average. This layer allows an increase in received mesh quality when transmitted and received properly (e.g. when enough bandwidth is available). In this traversal step we also obtain the mapping m from the original vertex indices I_{or} to the simplified leaf voxel indices I_{simp} and the mapping l between I_{simp} and the 3D voxel indices $v_{octree}\{x,y,z\}$ in the grid V_{octree} .

$$m: I_{or} \rightarrow I_{simplified}\{i_{or} \in I_{or}, m(i_{or}) \in I_{simplified}\}$$

$$l: I_{simplified} \rightarrow V_{octree}\{i_{simp} \in I_{simplified}, l(i_{simp}) \in V_{octree}\}$$

Based on the original connectivity $C_{original}$ provided by the reconstruction software where the triangles $T_{\text{unordered}}$ are provided without ordering we compute $C_{simplified}$. $C_{simplified}$ is composed by triangles $T_{ordered}$ where the vertex indices are ordered.

$$T_{unordered} = \{i_1, i_2, i_3\} : \{i_1, i_2, i_3 \in I_{original}, i_1 \neq i_2 \neq i_3\}$$

$$T_{ordered} = \{i_1, i_2, i_3\} : \{i_1, i_2, i_3 \in I_{simplified}, i_1 < i_2 < i_3\}$$

Ordering of the vertex indices i_{simp} in $C_{simplified}$ helps to preserve more of the spatial correlation that was introduced by the structured octree traversal. $C_{simplified}$ is computed by traversing the original connectivity $C_{original}$, and when distinct $m(i_{or1})$, $m(i_{or2})$, $m(i_{or3})$ are found in an original triangle $t_{\text{unordered}}$ they are stored as an ordered triangle in the new $C_{simplified}$ (note that in $T_{ordered}$ $i_1 \neq i_2 \neq i_3$ as $i_1 < i_2 < i_3$). So, $C_{simplified}$ can be described as a set of $T_{ordered}$ conditioned as:

$$C_{simplified} = \{t_{ordered}(i_1, i_2, i_3) \mid \exists t_{unord.}(i_{or1}, i_{or2}, i_{or3}) :$$

$$t_{unord} \in C_{original}, m(i_{or1}) = i_1, m(i_{or2}) = i_2, m(i_{or3}) = i_3\}$$

Next, the set $C_{simplified}$ is ordered in ascending order of i_1 and i_1 is coded via a run length coding scheme that codes the number of repetitions and increments in i_1. This is part 1 of the coded connectivity in Figure 2. Next, we construct geometry based representations T_{VQ} of each triangle $t_{ordered}$ in $C_{simplified}$ to code the second and third indices i_2, i_3. Every t_{VQ} represents $t_{ordered}$ where i_2 , i_3 are represented by a 3D shift in the octree voxel grid V_{octree} away from connected voxels $v_1 = l(i_1)$ and $v_2 = l(i_2)$. Hence, T_{VQ} is represented as:

$$T_{VQ} = \{i_1, z_2, z_3\} : \{i_1 \in I_{simplified}, z_2 \in Z^3, z_3 \in Z^3\}$$

Where z_2 and z_3 are signed discrete 3D vectors representing a shift in the octree voxel grid V_{octree}. The T_{vq} representations are obtained from T_{ordered} by F_{vq}:

$$F_{vq} : T_{ordered} \rightarrow T_{VQ} = \{i_1, l(i_2) - l(i_1), l(i_3) - l(i_2)\}$$

In each t_{vq} in T_{vq}, i_1 was already coded in part 1 and only z_2 and z_3 need to be encoded. By structuring the connectivity information in $T_{ordered}$ and F_{vq} we achieved a structure where z_2 and z_3 can be coded very efficiently via vector quantization. The prototype vectors *[-1,0,0]*, *[0,-1,0]* and *[0,0,-1]* occur over 75% of the cases z_2 and z_3 while around 15 % of the other cases are covered by the vectors *[-1,1,0]*, *[0,-1,1]*, *[1,-1,0]*. The remaining

signed binary vectors represent the last 7% of the cases. We developed an efficient VLC scheme to code these vectors with 2, 4 and 8 bit code words. In the exceptional other cases we store all components of z_2 and/or z_3. The decoder executes inverse operations, i.e the octree voxel structure is decoded with [6][7] and instead of only l we also compute the inverse l^{-1} that relates the 3D octree voxel index v_{octree} to $I_{simplified}$.i.e:

$$l^{-1}: V_{octree} \rightarrow I_{simplified}\{v_{octree} \in V_{octree}, l^{-1}(v_{octree}) \in I_{simplified}\}$$

This mapping is used to decode all $t_{ordered}$ recovering $C_{simplified}$ from all t_{vq} based on the relation F^{-1}_{vq}:

$$F^{-1}_{vq}: T_{VQ} \rightarrow T_{ordered} = \{i_1, l^{-1}(l(i_1)+z_2), l^{-1}(l(i_2)+z_3)\}$$

Where i_1 was already recovered from part 1 of the compressed connectivity data. Therefore, by recovering i_2 followed by i_3 for each t_{vq} $C_{simplified}$ is recovered.

5. Codec Evaluation and Comparison

The proposed scheme enables coding 3D connectivity and an enhancement layer alongside with the simplified vertex information implementing a full 3D Mesh codec. We compared the codec without enhancement to the mesh codec available in MPEG-4 presented in [3] configured for 8 bit coordinate and 4 bits color quantization (we also use 4 bits color coding in the proposed scheme) and differential prediction. We compared the rate-distortion trade-off in our scheme with the MPEG-4 codec applied on the completely enhanced simplified mesh. We use the datasets from [8] which contain dense photorealistic meshes reconstructed with Kinect based reconstruction software which are currently also used for evaluation in the MPEG-3DG group. We use the metro tool developed in [9] to compute the symmetric mean squared error metric for distortion evaluation. We tune the voxel size of the octree for surface simplification to control the rate. The results show that our method achieves only slightly higher rate-distortion compared to the MPEG-4 codec applied to the simplified mesh i.e., at very low bit-rates the mpeg-codec outperforms the scheme due to the fact that the voxel down-sampling becomes very coarse compared to the enhanced simplified mesh). Nevertheless, we believe that by future enhancements based on inter-frame prediction between the voxel data and better color coding we can achieve much better results very soon.

References

[1] K. Mamou, T. Zaharia, F. Preteux, "FAMC: The MPEG-4 standard for Animated Mesh Compression," in Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, San Diego, 2008, pp. 2676 - 2679.

[2] A. Doumanoglou, D. Alexiadis, D. Zarpalas, P. Daras, "Towards Real-Time and Efficient Compression of Human Time-Varying-Meshes," input document m30537, ISO iec sc29wg11 (MPEG), Vienna July 2013.

[3] K. Mamou T. Zaharia, and F. Preteux, "TFAN a low complexity 3D Mesh Compression algorithm," in Computer Animations and Virtual Worlds (CASA'09) pp. 343-354, 2009.

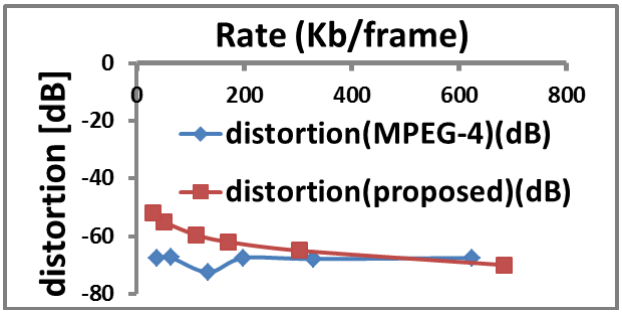

Figure 3 rate-distortion of proposed scheme

[4] J. Peng and C.-C. Jay Kuo, "Geometry-guided progressive lossless 3D mesh coding with octree (OT) decomposition," ACM Trans. Graph, vol. 24, no. 3, pp. 609-616, July 2005.

[5] R. Mekuria, P. Cesar. P, L. Bivolarsky, M. Preda, ISO MPEG N14197 Tele-immersion use case and associated draft requirements," San Jose, California, 2014.

[6] R.B. Rusu, "3D is here: Point Cloud Library (PCL)," in IEEE International Conference on Robotics and Automation, Shanghai, 2011, pp. 1-4.

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

[8] D. Alexiadis, D. Zarpalas, P. Daras, "Real-time, Realistic Full-body 3D Reconstruction and Texture Mapping from Multiple Kinects", 11th IEEE IVMSP Workshop: 3D Image/Video Technologies and Applications, Yonsei University, Seoul, Korea, 10-12 June 2013

[9] P. Cignoni, C. Rocchini and R. Scopigno (1998) Metro: measuring error on simplified surfaces *Computer Graphics Forum*, Blackwell Publishers, vol. 17(2), June 1998, pp 167-174

Rufael Mekuria received a Bachelor's and Master's Degree in Electrical Engineering from Delft University of Technology. He is currently with Centrum Wiskunde & Informatica in Amsterdam, the Netherlands. Since April 2013 he has been contributing to MPEG 3D Graphics group (3DG), where he is currently a co-chair of the AhG on 3DG .

Pablo Cesar leads the Distributed and Interactive Systems group at Centrum Wiskunde & Informatica: CWI. He received his PhD from the Helsinki University of Technology in 2006.