VIEW-DEPENDENT GEOMETRY CODING OF 3D SCENES

Frédéric PAYAN, François MERIAUX, Marc ANTONINI, member, IEEE

Laboratoire I3S (UMR 6070 CNRS-Université de Nice - Sophia Antipolis, France)

ABSTRACT

A view-dependent geometry coding of 3D scenes defined by sets of semi-regular meshes is presented. The objective is to reduce the quantity of significant data to store when visualizing static 3D scenes from a specific point of view. The proposed coding scheme combines a segmentation for determining the visible regions, and an allocation process for improving the visual quality of the encoded scene.

Index Terms— 3D scenes, view-dependent coding, segmentation, semi-regular meshes.

1. INTRODUCTION

As soon as we deal with 3D environments, a viewer will not be ubiquitous. When looking at a static 3D scene, a viewer cannot see all the regions of the different objects. So, if the position of the viewer is known during the coding stage, it becomes useless to preserve the regions that the viewer will never see. Also, among the visible parts, some features of objects located in the background of the scene may not be discerned because of the distance from the viewer, contrary to features of close objects. Visually speaking, the same quantization for all the objects unavoidably leads to a suboptimal coding. It seems more relevant to quantize more finely the geometry of objects in the foreground than in the background.

The objective of our study is to reduce the quantity of significant data to store when visualizing static 3D scenes from a specific point of view. This leads to the development of a view-dependent coding [1, 2, 3, 4]. The principle of the presented method is the following. For a given 3D scene, defined in our study by several semi-regular meshes, and a given position for the viewer, we first differentiate the visible parts from the invisible ones. Then, to privilege the reconstruction quality of close objects (in the foreground), a bit allocation process is necessary for optimizing the quantization of the geometry in function of the distance between the objects and the viewer. The proposed approach is a preliminary work for future applications like 3D movies storage.

The rest of the paper is organized as follows. Section 2 describes the technique for removing the invisible parts. Section 3 presents the coding scheme, and the distance-based bit allocation process. Some results are given in Section 4 to validate our contribution, and we conclude in Section 5.

2. DETECTION OF THE VISIBLE PARTS

2.1. Classification of the front-facing regions

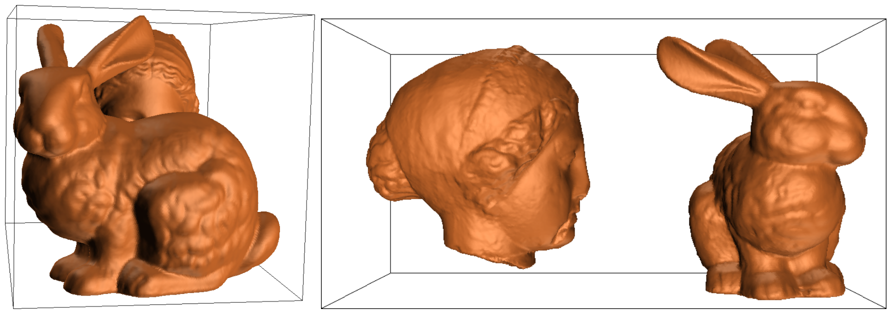
Let us define a 3D scene S composed by a set of N semi-regular meshes: $S = \{M_0, ..., M_m, ..., M_N\}$, and a given position of the viewer. The structure of a semi-regular mesh is a set of base triangles (giving a coarse version of the shape) and several levels of regular vertices added successively by regular subdivision [5]. For the view-dependent coding of this 3D Scene, the first step is to determine which regions can be seen by the viewer. For this purpose, two kinds of visibility tests are usually performed on the meshes: the view-frustum test that detects the part of the 3D-space visible in a rectangular screen [1], and a surface-orientation test that defines which parts of the object face the viewer [1, 2].

The view-frustum test consists in checking which vertices lie inside the view pyramid (its base represents the rectangular screen) [1]. Let us define each side of the pyramid by $a_j x + b_j y + c_j z + d_j = 0 \; for \; j = 1..4$. One vertex $v_i = (v_i^x, v_i^y, v_i^z)$ is in the view-frustum if $a_j v_i^x + b_j v_i^y + c_j v_i^z x + d_j > 0 \; for \; j = 1..4$.

One classical approach to test if one triangle faces the viewer is to compute its normal and then to compare its direction with the view vector. If these two vectors have the same orientation, this triangle faces the viewer. Applying such an approach for a given 3D scene would permit to detect the front-facing triangles. But in a 3D scene, a region can face the viewer, and be eclipsed by another region. So, this test is not efficient for full 3D scenes. Therefore we prefer using a ray/triangle intersection test [6] to detect the invisible regions. This test should determine the back-facing regions, but also the front-facing regions hidden by another part of the scene. Such a test should provide finer results for 3D scenes, but also for objects with highly-changing curves.

The principle is the following. For each vertex of a given 3D scene, we perform the ray/triangle intersection between the observation line and the set of triangles. Among the intersected triangles, we select the ones which are not in the first-neighborhood of the tested vertex. If one of those triangles is closer to the viewer than the vertex, then it hides this vertex.

Figure 2 illustrates the results of this classification method. The input data is the synthetic 3D scene named BUNNYANDVENUS (Figure 1), where the Bunny hides a large part of

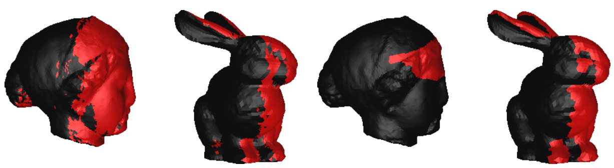


(a) Front view.

(b) side view.

Fig. 1. A 3D scene composed by two semi-regular meshes.

the Venus head. We observe a significant difference between the two tests. The classification based on this ray/triangle intersection test is more selective than the classification based on the surface normals.



(a) Using the surface normals.

(b) Using the ray/triangle intersection test.

Fig. 2. Results of the two tested classifications.

2.2. Segmentation of the visible parts

Once the classification is done, the significant parts of the semi-regular meshes that contain all the highlighted triangles have to be segmented. To preserve the specific structure of the semi-regular meshes which makes the connectivity implicit excepted the list of base triangles (relevant for an efficient coding), the boundaries of the visible regions must be projected on the base triangles (see Figure 3). The resulting set of base triangles and all the associated vertices until the finest resolution level determine the output of the proposed segmentation, *i.e.*, the regions visible from the given position.

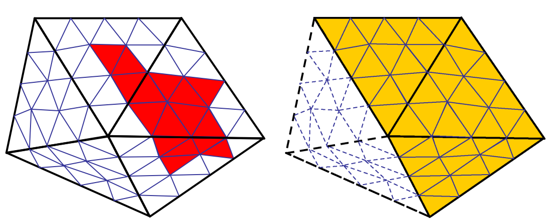


Fig. 3. The red triangles (on the left) represent the visible regions determined by classification. The orange region (on the right) represents the segmentation output.

Figure 4 shows one result of the proposed view-dependent segmentation. This method ensures that the viewer cannot see any visual difference with the original scene, while reducing strongly the quantity of information needed to represent perfectly the 3D scene.

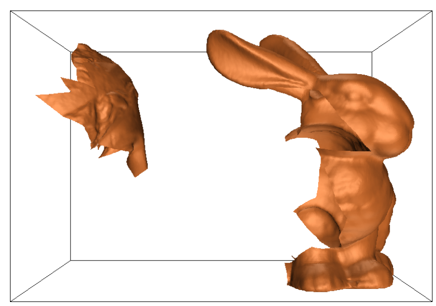


Fig. 4. View-dependent segmentation (side view).

3. DISTANCE-BASED CODING SCHEME

Once the given 3D scene is split into visible/invisible parts, a coding scheme specific to 3D scenes is needed to take into account the positions of the different objects in the visible 3D-space. Indeed, when looking at a 3D-scene, the details of objects are much more visible in the foreground than in the background. Consequently, a given quantization error on objects far (respectively close) from the viewer leads to lower (respectively higher) visual loss.

A bit allocation is a relevant process to determine the level of quantization in function of the distance between the objects and the point of view. The goal of such a process is to minimize the trade-off between rate and distortion relative to the data quantization. Globally, this constrained allocation problem can be solved by minimizing the criterion

$$J_\lambda(Q) = D(Q) + \lambda(R(Q) - R_{target}), \quad (1)$$

with λ the lagrangian operator, $D(Q)$ and $R(Q)$ respectively the distortion and the bitrate relative to the quantizer Q of the

3D scene geometry, and R_{target} a user-given bitrate. Computing the quantizer Q^* minimizing this criterion will permit to obtain the optimal level of quantization for each object, in function of the user-given birate, and the position of the viewer. To take into account the position of the different objects in function of the viewer, we propose to estimate the global distortion with a weighted sum of the MSE D_m computed independently on the M different objects:

$$D(Q) = \sum_{m=1}^{M} \Pi_m D_m(Q_m).$$

$\{\Pi_m\}$ represent the weights used to take into account the distance between the visible regions and the observation position. The weights are given by

$$\Pi_m = \frac{1}{\bar{d}_m^2},$$

with $\bar{d}_m$ the mean distance between the observation point and all the vertices of the visible part (given by the segmentation) of the mesh m. This formulation ensures that the weights are inversely proportional to the associated distances. The distortion of far objects will be attenuated by the weighting, and the quantization will be consequently finer for the close objects. This expression is a simple technique to improve the perceived visual quality of quantized 3D scenes, directly from the purely geometric MSE.

The coding scheme used during this work is based on wavelet filtering such as in [7]. The wavelet filtering transforms the M semi-regular meshes of the 3D scene S in M coarse meshes (low frequency signal) and M sets of subbands of wavelet coefficients (high frequency details). With such a structure, the MSE D_m and the bitrate R_m of the m^{th} mesh can be estimated respectively by

$$D_m(Q_m) = \sum_{i=1}^{nres} w_i D_{m,i}(Q_{m,i}),$$

and

$$R_m(Q_m) = \sum_{i=1}^{nres} a_i R_{m,i}(Q_{m,i}).$$

$D_{m,i}(Q_{m,i})$ and $R_{m,i}(Q_{m,i})$ are respectively the MSE and the bitrate due to the quantization $Q_{m,i}$ of the $(m,i)^{th}$ subband. $\{w_i\}$ and $\{a_{m,i}\}$ are respectively the weights relative to the non-orthogonality and the subsampling effect of the wavelet filtering used [7]. $\{a_{m,i}\}$ represents the ratio between the size of the $(m,i)^{th}$ subband of the mesh m and the number of vertices in the 3D scene. The criterion (1) becomes

$$J_\lambda(\{Q_{m,i}\}) = \sum_{m=1}^{M} \Pi_m \sum_{i=1}^{nres} w_i D_{m,i}(Q_{m,i}) + \lambda(\sum_{m=1}^{M} \sum_{i=1}^{nres} a_{m,i} R_{m,i}(Q_{m,i}) - R_{target}), \quad (2)$$

The set of optimal quantizers $\{Q^*_{m,i}\}$ are finally obtained by solving the following system:

$$\begin{cases} \frac{\partial J_\lambda(\{Q_{m,i}\})}{\partial Q_{m,i}} = 0 \\ \frac{\partial J_\lambda(\{Q_{m,i}\})}{\partial \lambda} = 0 \end{cases}. \quad (3)$$

The solution of this system, as well as the resulting implementation of the whole coding scheme, are explained in details in [7]. Pleaser refer for more explanations.

4. EXPERIMENTAL RESULTS

Figure 4 shows the Rate-PSNR curves obtained with the global approach [7] and the proposed one. The PSNR depends on a RMSE based on the popular Hausdorff distance which measures the distance between two surfaces (computed with MESH [8]). The rate is reported in bits/vertex, according to the number of vertices of the original 3D scene. This curve shows the amount of unnecessary information when applying the global approach. For a same PSNR, the view-dependent coder strongly reduces the size of compressed data, whatever the target bitrate.

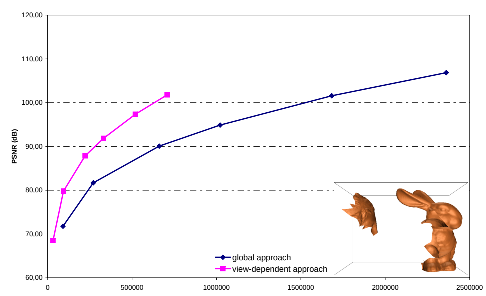
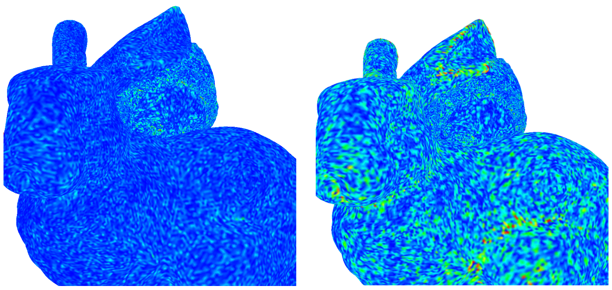


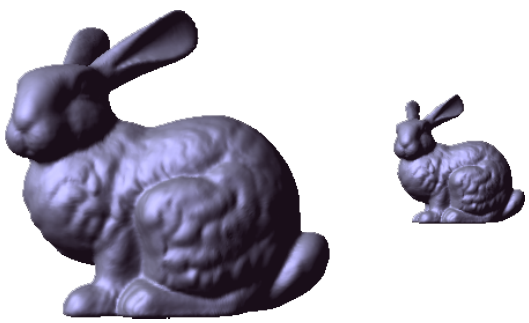
Fig. 5. PSNR curves obtained with the global approach [7] and the proposed one.

Second, to show the interest of the distance-based allocation process, the Hausdorf distance between original and quantized meshes is displayed locally on the different objects composing the 3D scenes BUNNYANDVENUS (figure 6), and another 3D scene DOUBLEBUNNY composed with two Bunny models (figure 7). Note that the RMSE based on the Hausdorf distance cannot be used here to evaluate the efficiency of the allocation process, which tends to improve the perceived visual quality and not only the geometric quality. Figure 7(b) shows that the close bunny in the scene DOUBLEBUNNY is quantized more finely than the far one, whereas the previous global approach produces exactly the same quantization errors on the two bunnies (Figure 7(c)). It proves that the distance-based allocation process considers the point of view. The local details are better preserved in the foreground than in the background which is coherent with a perceived visual quality. Similar results are obtained with the scene BUNNYANDVENUS (see Figure 6).

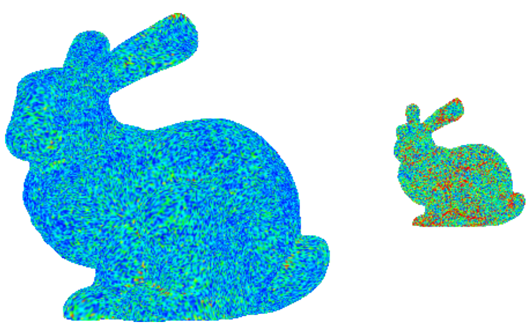


(a) Distance-based allocation. (b) Classical allocation [7].

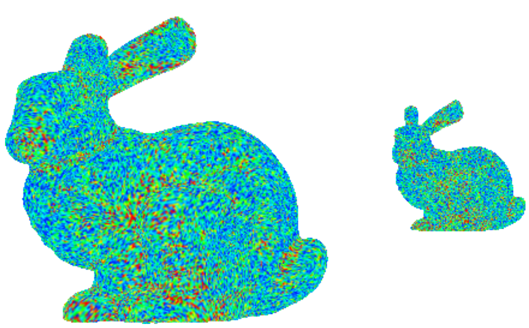
Fig. 6. Visual results depending on the allocation process. The colours depends on the magnitude of the local errors, from the minimal (blue) to the maximal value (red).



(a) the 3D scene DOUBLEBUNNY.



(b) Distance-based allocation.



(c) Classical allocation [7].

Fig. 7. Visual results depending on the allocation process.

5. CONCLUSION AND FUTURE WORKS

The objective was to propose a view-dependent geometry coding for static 3D scenes. The first contribution is the segmentation which determines the regions visible from one specific point of view in order to reduce the size of relevant data. The second contribution is the distance-based allocation process which improves the perceived visual quality of the encoded 3D scene.

One future work is to design a more complex distortion model which will further consider visual perception features of 3D scenes. For instance, quantization errors are not visually perceived by the same manner on the silhouettes or on the front-facing regions of 3D objects [9]. Also, the visual impact of quantization errors depends on the illumination of rendered 3D objects [4]. Considering these features during the allocation is a promising way to further improve the visual quality of complex 3D scenes. In parallel, one major objective is to extend our coding scheme to the animated 3D scenes, by considering the temporal evolution.

6. REFERENCES

[1] H. Hoppe, "View-dependent refinement of progressive meshes," *SIGGRAPH'97, Computer Graphics Annual Conferences Series*, pp. 189–198, 1997.

[2] D. Luebke and C. Erikson, "View-dependent simplification of arbitrary polygonal environments," in *Proceedings of ACM SIGGRAPH'97*, 1997, pp. 199–208.

[3] J. Sim, C. Kim, C. Kuo, and S. Lee, "Rate-distortion optimized compression and view-dependent transmission of 3D normal meshes," *IEEE TCSVT*, vol. 14, July 2004.

[4] W. Guan, J. Zhang, J. Cai, and J. Zheng, "Illumination and view dependent progressive transmission of semi-regular meshes with subdivision connectivity," in *In Computer graphics and interactive techniques in Australia and South East Asia*, 2006, pp. 219–226.

[5] A.W.F Lee, W. Sweldens, P. Schröder, P. Cowsar, and D. Dobkin, "Maps: Multiresolution adaptive parameterization of surfaces," *SIGGRAPH*, 1998.

[6] T. Möller and B. Trumbore, "Fast, minimum storage ray-triangle intersection," *Journal of Graphics Tools*, 1997.

[7] F. Payan and M. Antonini, "An efficient bit allocation for compressing normal meshes with an error-driven quantization," *Computer Aided Geometric Design*, 2005.

[8] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "Mesh: Measuring errors between surfaces using the hausdorff distance," in *Proc. of IEEE ICME*, 2002.

[9] D. Luebke and B. Hallen, "Perceptually driven simplification for interactive rendering," in *Proceedings of Eurographics Workshop on Rendering*, 2001.