COMPRESSION OF DEPTH INFORMATION FOR 3D RENDERING

Pietro Zanuttigh, Guido M. Cortelazzo

University of Padova, Italy

ABSTRACT

This paper presents a novel strategy for the compression of depth maps. The proposed scheme starts with a segmentation step which identifies and extracts edges and main objects, then it introduces an efficient compression strategy for the segmented regions' shape. In the subsequent step a novel algorithm is used to predict the surface shape from the segmented regions and a set of regularly spaced samples. Finally the few prediction residuals are efficiently compressed using standard image compression techniques. Experimental results show that the proposed scheme not only offers a significant gain over JPEG2000 on various types of depth maps but also produces depth maps without edge artifacts particularly suited to 3D warping and free viewpoint video applications.

Index Terms— Image coding, Image segmentation

1. INTRODUCTION

3DTV systems require efficient ways to compress and transmit depth maps, as such data are both produced by many 3D scanners and passive reconstruction algorithms and needed in 3D display devices and visualization algorithms. The first straightforward approach to depth maps' compression is to treat them as standard images and to use standard image or video compression tools like JPEG2000 or H.264. However these standards aim at the minimization of image visual quality and do not exploit the peculiarities of depth maps, therefore being not optimal. Some authors have shown that by reshaping the dynamic range and by properly tuning compression parameters it is possible to improve the performance of image compression tools. In particular Krishnamurty et al.[1] have shown that the Region Of Interest (ROI) feature of JPEG2000 driven by an edge detector can be used to avoid artifacts on the edges. The careful handling of edges is one of the key points in depth maps' compression, in fact they are usually made of smooth regions divided by sharp edges. By exploiting this structure it is possible to achieve good compression performance [2]. Another family of approaches is based on the conversion of the depth map to a 3D mesh [3]. In free viewpoint video applications another interesting field of research is how to exploit the redundancy between the various depth maps corresponding to different views of the same scene for an efficient compression [4].

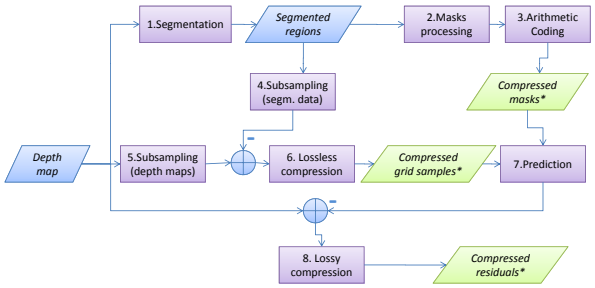

Fig. 1. Architecture of the compression system

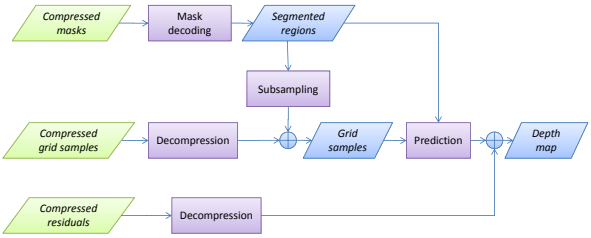

Fig. 2. Architecture of the decompression system

2. OVERVIEW OF THE PROPOSED COMPRESSION SCHEME

The proposed compression scheme rests on the common assumption that depth maps are usually made by a set of quite smooth surfaces separated by sharp edges between them.

The edges represent the key issue in depth map compression. Small errors in their positions or blurring of them lead to huge artifacts in typical uses of the depth information such as novel views' reconstruction from different viewpoints. They also represent one of the biggest issues when directly applying image compression tools on depth data.

The compression system we propose is based on the architecture shown in Fig.1, where the rectangular boxes represent processing steps and the parallelograms indicate data. In particular the data of the output compressed bitstream is marked by an asterisk. The first task is the segmentation of the depth map (*rectangular box 1*). Segmented data are then compressed exploiting the approach presented in Section 2.1 (*box 2 and 3*). In the second task both the original depth map

and its segmented version are subsampled and the difference is compressed as a JPEG2000 lossless image (*box 4,5 and 6*). The next task of the algorithm is the construction of a prediction of the original depth map from its subsampled version and the segmented data exploiting 3D clues (*box 7*). Finally the residuals between the prediction and the actual depth map are computed and lossy compressed in JPEG2000 (*box 8*). The decompression stage (Fig.2) follows the same steps in the reversed order: the subsampled depth map and the segmentation data are decompressed and used to predict the depth map. Then compression residuals are added to the prediction.

2.1. Segmentation and compression of the segmented data

In order to extract the different objects in the scenes and to identify the edge positions, we used the pyramidal segmentation algorithm implemented in the OpenCV libraries(*box 1*).

If the source image has been divided into n regions $R_k, k = 1, 2, ..., n$ by the segmentation algorithm, the segmentation output can be represented by a set of n binary masks $m_k(x)$, one for each segmented region. The k_{th} mask has a bit for each pixel indicating if the corresponding pixel belongs to that region, i.e., $m_k(x) = 1$ if $x \in R_k$ and $m_k(x) = 0$ otherwise. The proposed compression scheme for the segmentation output works as follows:

1. The average depth value a_k for each segmented region R_k is computed and stored. The n regions are then sorted on the basis of the corresponding average a_k and an index is associated to each mask (denoted with i).

2. A new set of $\lceil \log n \rceil$ masks M_j is built (*box2*). Each of the M_j corresponds to a bitplane in the mask index representation: If pixel x belongs to mask m_i then the masks corresponding to the bits set to one in the binary representation of i are set to one for that pixel. In the decompression stage the index of the pixel segment can be computed simply as $i(x) = \sum_{j=0}^{\lceil \log n \rceil - 1} M_j(x) 2^j$. This approach reduces the number of masks that need to be compressed and the size of the corresponding compressed data (in spite they generally have more complex shapes experimental result shows an average improvement of about 20% in the compressed mask data).

3. Each of the M_j is lossless compressed using an arithmetic coder (*box 3*). Lossless coding is needed since the masks contain the key information about edge positions. We used the JBIG coding scheme based on IBM's Q-coder. This solution offers a very good compression performance and also a multi-layer feature that can be used to handle the set of masks.

The next task (*box 5*) concerns the subsampling of the depth data. Firstly the depth map is subsampled on a regular grid as shown by the yellow dots in Fig 3b. Let's denote with Δx the grid squares' size, values of 8x8 or 16x16 provided the best performance. The idea is to take regularly spaced samples of the 3D surface represented by the depth map, without any filtering in order to avoid edges blurring. If W and H are the dimensions of the original depth map, the subsampled depth values are basically a reduced resolution depth map of size $(W/\Delta x)$ by $(H/\Delta x)$. It can be compressed using JPEG2000 exploiting the lossless compression mode (*box 6*). Lossless compression is necessary to avoid blurring on the edges and averaging between close samples belonging to different regions which would make useless the benefits of the segmentation step. However the low resolution allows to obtain small file sizes even in lossless compression. It is also possible to improve the compression performance by subtracting the average depth value of the corresponding segment to each sample and by compressing the difference between the two values (*box 4* of Fig.1 refers to this solution). In this way sharp edges are removed from the subsampled depth map and the size of the corresponding compressed file is reduced.

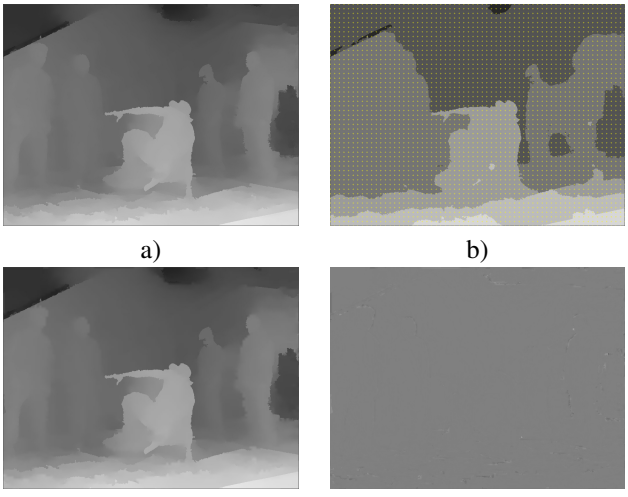

Fig. 3. a) Depth map; b) Segmented depth map (yellow dots represent the position of the grid samples); c) Predicted depth map; d) Difference between predicted and actual depth map

2.2. Prediction of the depth map

One of the key steps in both the coding and decoding procedure is the estimation of the depth map (or the surface shape) from the subsampled depth map and the full resolution segmentation output (*box 7*).

The idea is that the depth map is made of a set of smooth surface regions represented by the segmented data. For each region a set of samples is available (represented by the values of the corresponding subsampled depth map pixels). The proposed prediction scheme uses the pixels of the sparse grid corresponding to the subsampled depth map (the yellow dots

in Fig. 3b) and the surface regions' shape information derived from the segmentation. Each pixel of the depth map can therefore be surrounded by up to 4 samples of the grid that belong to the same region (shown in yellow in Fig. 4). The value of each pixel can be estimated by interpolating the values of the grid samples that belong to the same region. There are 5 possible cases (excluding the trivial case of samples that correspond to grid points). Figure 4 shows an example of each of them but other symmetrical or rotated configurations are possible. The proposed surface estimation approach is based on the assumption that each segmented region ideally represents a single object in the 3D scene and it can be approximated by a set of planar patches. The algorithm works in the following way for the 5 different configurations:

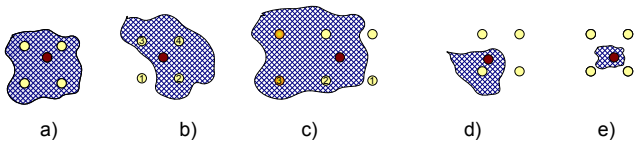

Fig. 4. Grid samples: the 5 possible configurations

a) All the 4 samples surrounding the pixel to be predicted (shown in red in Fig. 4) are inside the same region. In this case the pixel value is obtained from the value of the 4 closest grid samples by bilinear interpolation[1].

b) The pixel to be predicted is surrounded by 3 samples belonging to the same region but the fourth is out of the region. In this case we firstly estimate the value that the fourth grid sample would have if it was in the region by assuming that it is on the plane defined by the other three points. Referring to the pixel position numbering of Fig. 4b, this corresponds to estimate the value of P_1 as $P_1 = P_2 + P_3 - P_4$. Then we compute the pixel value using bilinear interpolation (with the 3 samples in the region and the prediction of the fourth) as in the previous case.

c) The pixel to be predicted has 2 neighbors in the region and two outside (this typically it happens when it is close to an edge). The two missing points are predicted by assuming that each of them lies on the line passing through the closest available point and the symmetrical point with respect to the available one (shown in orange in Fig 4c). With the notation of Fig. 4c, for example, we obtain $P_1 = 2P_2 - P_3$. If the symmetrical point is also outside the region the closest point value is chosen as prediction. As in the previous case once the two missing grid samples are obtained the pixel to be predicted is computed by bilinear interpolation.

d) The pixel to be predicted has just one neighbor grid sample in the same region: the value of this sample is taken as its estimate.

e) The pixel to be predicted is inside a very small region and none of the 4 closest grid samples are inside it (quite uncommon but possible). The average depth value of the region a_k is taken as the pixel prediction value.

A very important aspect of the proposed procedure is that the prediction is always built from grid samples in the same region of the pixel to be evaluated, while grid samples outside the region are never used. This ensures that steep edges do not get blurred unless they are not detected from the segmentation(see Fig. 3c).

2.3. Residual compression

Finally the difference between estimated and actual depth map is computed and lossy compressed by JPEG2000 (*box 8*). As it is possible to see from Fig.3d the depth prediction is quite similar to the actual depth and the residual image has very small values only. In particular it does not contain large transitions and usually the biggest information content of the residual difference is related to areas where the segmentation has not been able to divide the image properly. In order to achieve nearly-lossless compression the residuals typically correspond to the largest block of data in the compressed file. However residual is compressed in JPEG2000 and it is possible to exploit all the scalability features of this standard to compress and transmit them in a progressive manner.

The decompression step basically follows the compression procedure in the reversed order. Firstly the JPEG2000 image with the grid samples and the JBIG file with the masks are decompressed. Then the masks are recombined using the technique described in Section 2.1 and the prediction of the depth map is built from the segmentation regions and the grid samples. Finally compression residuals are decompressed and added to the estimated depth map.

3. EXPERIMENTAL RESULTS

The first example shows the performance of the proposed scheme on Microsoft's *breakdance* sequence. In this example for the first frame (that has a resolution of 1024x768) the segmentation data requires 5834 bytes (the proposed scheme does not require the creation of too many segmented regions, it is necessary just to extract the main scene elements). The mean values of the 7 segmented regions require just one byte for each of them, the grid samples 1758 bytes ($\Delta x = 16$) or 4629 bytes ($\Delta x = 8$) and the remaining part of the bitrate is allocated to the residuals (from 1 to 50 KBytes). Plot 5

[1]This approach is consistent with the assumption that most surface patches are planar. Experimental proofs with splines and other cubic interpolation schemes has shown that in most practical cases these techniques do not provide any improvement on the surface reconstruction accuracy.

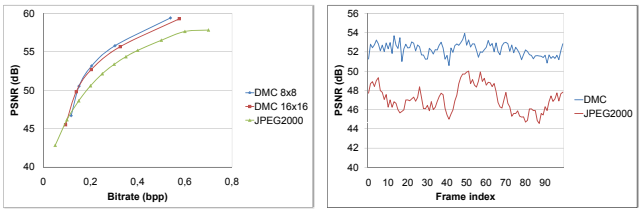

Fig. 5. Experimental results: *breakdance* depth map (first frame and complete sequence at 0.2bpp)

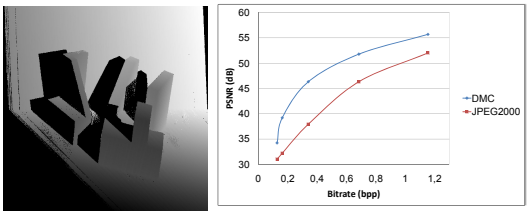

Fig. 6. Experimental results: *ABW* depth map

compares the proposed approach[2] (denoted with "DMC") and JPEG2000. The left plot shows how our approach achieves a PSNR gain of around 3 dB for bitrates bigger than 0.15 bpp (JPEG2000 compression is performed with the *Kakadu* coder). At very low bitrates segmentation data become predominant and the performance is lower. It also shows that for this image the 8x8 grid is the optimal choice (unless very low bitrates are needed). As expected, finer grids require more data for the samples but residuals are smaller at comparable PSNR. In general larger grids work well for simple depth maps with large smooth surfaces, while more detailed depth maps require finer grids. The right plot of Fig. 5 shows the performance on all the 100 frames of the *breakdance* sequence at 0.2 bpp. The PSNR average of the proposed approach on the complete sequence is around 52.2 dB, about 4 dB better than JPEG2000 and comparable to the performance of H.264 (51.6 dB at 0.19 bpp). It is worth noting that H.264 uses also motion prediction. Therefore the performance of the proposed scheme combined with a motion prediction algorithm is likely to outperform H.264 (we are currently implementing a system of this kind).

Fig.6 shows an example from the ABW [5] range image dataset representing a typical example of the output of 3D scanning systems. In this case (Fig. 6) the proposed approach has an average gain of 5dB over JPEG2000 , showing its effectiveness in compressing the data produced by actual range scanners. The impressive results on this image are due both to the presence of many planar surfaces, well handled by the surface prediction scheme of Section 2.2 and by the effectiveness of the segmentation step in handling the many isolated bright points (that are critical for standard image compression tools working in the frequency domain).

The preliminary experimental results performed by including the proposed approach into the framework of [6] have shown that it produces depth maps particularly suited to 3D warping applications. This is mainly due to the precise representation of edges which prevents the artifacts typical of other compression schemes in 3D warping procedures. The proposed algorithm also does not require complex computations (the largest part of the computation time is allocated to the segmentation and JPEG2000 residual compression).

[2]A visual comparison of the compressed images is available at http://lttm.dei.unipd.it/nuovo/research/depth-compression.html

4. CONCLUSIONS

This paper introduces a novel depth maps compression scheme which allows to compress in a very efficient way depth data preserving the sharp transitions on the objects' edges and at the same time to achieve high compression rates on smooth regular surfaces. Experiments show a gain over standard JPEG2000 compression of about 3-5 dB for medium-high bitrates. Segmentation data limit performance at very low bitrates but further work will be devoted at improving low-bitrate coding. We are currently working on a motion prediction stage in order to fully exploit the proposed method in free viewpoint video and 3DTV coding applications. Finally the proposed compression strategy will also be included into the remote visualization framework of [6] in order to improve the geometry compression module.

5. REFERENCES

[1] R. Krishnamurthy, B.B. Chai, H. Tao, and S. Sethuraman, "Compression and transmission of depth maps for image-based rendering," in *Proc. ICIP 2001*, 2001, pp. 828–831.

[2] Y. Morvan, D. Farin, and P. de With, "Depth-image compression based on an r-d optimized quadtree decomposition for the transmission of multiview images," in *Proc. ICIP 2007*, 2007.

[3] S.K. Penta and P. J. Narayanan, "Compression of multiple depth maps for ibr," *The Visual Computer*, vol. 21, no. 8-10, pp. 611–618, 2005.

[4] Y. Morvan, D. Farin, and P. de With, "Predictive coding of depth images across multiple views," in *Proceedings of SPIE, Stereoscopic Displays and Applications*, 2007.

[5] A. Hoover et al., "An experimental comparison of range image segmentation algorithms," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 673–689, 1996.

[6] P. Zanuttigh, N. Brusco, D. Taubman, and G.M. Cortelazzo, "A novel framework for the interactive transmission of 3d scenes," *Signal Processing: Image Communication*, vol. 21, no. 9, pp. 787–811, October 2006.