Inter-Frame Compression of 3D Point Cloud Sequences

Ji-Su Kim, Seonho Lee, Jae-Han Lee, Chang-Su Kim
School of Electrical Engineering, Korea University, Seoul, South Korea
{jisukim, seonholee, jaehanlee}@mcl.korea.ac.kr, changsukim@korea.ac.kr

***Abstract*—In this paper, we propose a 3D point cloud sequence compression algorithm based on inter-frame coding. First, we create an octree structure for the 3D point cloud. For the first frame, we compress geometric and color attributes through intra-frame coding. Then we apply inter-frame coding to the rest of the frames. We compress the geometric attributes using a dual octree and compress the color attributes using the k-means clustering. Both intra-frame coding and inter-frame coding are performed with arithmetic codes. Experimental results demonstrate that the proposed algorithm compresses point cloud sequences more efficiently than the algorithm using only intra-frame coding.**

Keywords—point cloud; dual octree; intra-frame coding; inter-frame coding

I. Introduction

A 3D point cloud is a type of 3D data, which contains location information and color information of numerous 3D points. Nowadays 3D point cloud is widely used for many visualization tasks in various fields such as 3D rendering [1], 3D reconstruction, and autonomous driving. However, since a point cloud consists of many 3D points, it demands a huge amount of storage space, as well as high transmission bandwidth. Therefore, it is essential to develop an efficient 3D point cloud compression techniques.

Various attempts have been made to compress 3D point cloud sequences. For instance, [2] used a kd-tree structure to sub-divide of a point cloud space at different resolution layers. Also, [3] used the octree structure to compress point cloud attributes, such as colors. The octree data structure to decompose the 3D space is an effective means to represent spatial point cloud distribution. As compared with the mesh construction, the octree decomposition is simpler to achieve.

In this paper, we propose a 3D point cloud sequence compression algorithm using a dual octree tree and the k-means clustering technique. Also, we adopt arithmetic codes for achieve a higher coding gain. Experimental results show that the inter-frame coding performs better than the intra-frame coding only.

II. Proposed Algorithm

A. Overview

The number of points constituting each point cloud frame changes every time. This is because, when generating a point cloud sequence, the number of points captured by the camera is different at each frame in general. The fact that the number of points in each frame is different makes it difficult to find geometry and color differences between adjacent frames. In other words, it is difficult to find correspondences between frames. To overcome this problem, [4] suggests the dual octree technique for a geometry coding and the k-means clustering for a color coding.

B. Octree Construction

Each point in a 3D point cloud contains geometry information and color information. We build an octree structure from the input 3D point cloud data based on the geometry information.

First, we decompose the 3D space into a lot of voxels. Each voxel contains 8 child nodes. If there is a point in the child voxel, the corresponding child node value is set to 1, otherwise 0 is assigned. This 0 and 1 sequence forms a bit stream.

C. Geometry Coding

The bit stream contains the spatial information of point cloud. We obtain the spatial variation of the two adjacent frames by comparing the bit stream of each frame. However, since the total numbers of points are different, it is not possible to compute the spatial variation by comparing the bit stream directly. To compute the spatial variation, we employ the dual octree method.

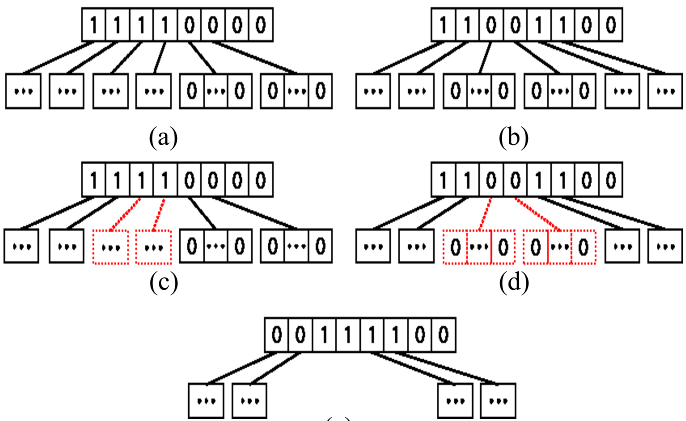

Fig. 1. (a), (b) Reference and target octrees at the first step, (c), (d) reference and target octrees at the second step, and (e) the XOR result octree at the final step.

We build a dual octree in three steps, as illustrated in Fig. 1. We denote the voxel value of the reference octree by v_r, the voxel value of the target octree by v_t, the child node of v_r by n_r and child node of v_t by n_t. At the first step, we compare two octrees obtained from adjacent two frames. In Fig. 1. (a), If v_r is 0 and v_t is 1, we assign 0 to n_r. In Fig. 1. (b), if v_r is 1 and v_t is 0, we assign 0 to n_t. Through this process, the sizes of two octrees become equal. Next, we compare the octrees to obtain a dual octree. In Fig. 1. (c), if v_r is 0 and v_t is 1, we assign 0 to n_r. In Fig. 1. (d), if v_r is 1, and v_t is 0, we remove both n_r and n_t. Thus, we build the dual octree that has the same size as the target octree. Subsequently, we perform the XOR operation using the dual octree to compute the spatial variation of two frames. Fig. 1. (e) shows the final result of the XOR operation.

D. Color Coding

The sizes of adjacent two frames are different, and it is hard to determine corresponding color differences. Therefore, we reconstruct an estimated color attribute map that has the same size with the target frame using the k-means clustering.

Let q denote a point in the target frame, and N_q denote the set of nearest neighbor points of q. We obtain p which $p \in N_q$ by the Euclidean distance of the 3D geometry information of the target frame $g_t(q)$ and the geometry information of reference frame g_r. We use $k = 3$ as the number of nearest neighbors.

We compute the predicted color of q using the geometry information of the reference frame and the k-means clustering. More specifically, for each q, the predicted color in the target frame $\tilde{c}_t(q)$ can be estimated by computing the average color value of the point p, which is

$$\tilde{c}_t(q) = \frac{1}{|N_q|} \sum_{p \in N_q} c_r(p). \tag{1}$$

Finally, we compute $\Delta c_t(q)$, the residual of the target frame color information and the predicted color information. This removes temporal redundancy in the point cloud attribute. When the computed $\Delta c_t(q)$ is smaller than a threshold θ, we assume that v$\Delta c_t(q)$ as 0, i.e.,

$$\Delta c_t(q) = \begin{cases} \Delta c_t(q), & \text{if } \Delta c\_t\ (q) > \theta \\ 0, & \text{otherwise} \end{cases}. \tag{2}$$

E. Arithmetic Coding

For entropy coding of the geometric and color differences between two adjacent frames, we use the arithmetic codes in [5].

III. Experimental results

We use JPEG Pleno Database: Microsoft Voxelized Upper Bodies – A Voxelized Point Cloud Dataset [6]. We downsample the dataset through voxelization. Each frame consists about 8,300 points. We perform experiments using 10 frames, whose size is about 5.39MB.

The intra-frame coding compresses absolute geometry and color values. On the other hand, the inter-frame coding compresses a residual bit stream for the geometry attribute and residual color values for the color attribute. The proposed algorithm compresses the first frame using the intra-frame coding and the rest frames using the inter-frame coding. We measure the compression ratio using bits per voxel (bpv).

Table I shows the changes in the compression ratio according to the threshold θ. Note that the proposed algorithm achieves better compression ratio than inter-frame coding only. The intra coding compresses a frame with about 24KB. In contrast, the inter coding compress a frame with about 4.7KB. By increasing θ, we can obtain a compacter bit-stream.

Fig. 2. shows a reference frame, a target frame, and a reconstructed frame, encoded with $\theta = 20$. We see that there are no noticeable image distortions.

Table I. Comparison of compression ratio

type	intra-frame coding	Proposed inter-frame coding	
θ	-	10	20
size(KB)	240	87	71.25
bpv	23.1	8.4	6.9

(a) (b) (c)

Fig. 2. (a) Reference frame, (b) target frame, and (c) reconstructed target frame

IV. Conclusion

We proposed an inter-frame coding technique for point cloud sequences. First, we convert the point cloud data into an octree structure. Then, we compute the geometry differences between two adjacent frames using the dual octree, and the color differences using the k-means clustering. Experimental results showed that the proposed algorithm compresses point data more efficiently than the intra-frame coding only.

Acknowledgement

This work was supported partly by 'The Cross-Ministry Giga KOREA Project' grant funded by the Korea government (MSIT) (No. GK18P0200, Development of 4D reconstruction and dynamic deformable action model based hyper-realistic service technology), and partly by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2018R1A2B3003896).

References

[1] Han-Ul Kim, Jin-Hwan Kim, and Chang-Su Kim, "Hybrid representation and rendering of indoor environments using meshes and point clouds," in Proc. URAI 2014, Nov. 2014.

[2] O. Devillers and P.-M. Gandoin, "Geometric compression for interactive transmission," in Proc. IEEE Vis., Oct. 2000, pp. 319–326.

[3] C. Zhang, D. Florêncio, and C. Loop, "Point cloud attribute compression with graph transform," in Proc. IEEE ICIP., Oct. 2014, pp. 2066–2070.

[4] D. Thanou, P. Chou, and P. Frossard, "Graph-based compression of dynamic 3D point cloud sequences," in IEEE Trans. Image Proc., April 2016, vol. 25, no.4, pp. 1765–1778.

[5] D. Marpe, H. Schwarz, and T. Wiegand, "Context-adaptive binary arithmetic coding in the H.264/AVC video compression standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, pp. 620–636, July 2003.

[6] C. Loop, C. Zhang, and Z. Zhang, "Real-time high-resolution sparse voxelization with application to image-based modeling," in Proc. 5th High-Perform. Graph. Conf., 2013, pp. 73–7