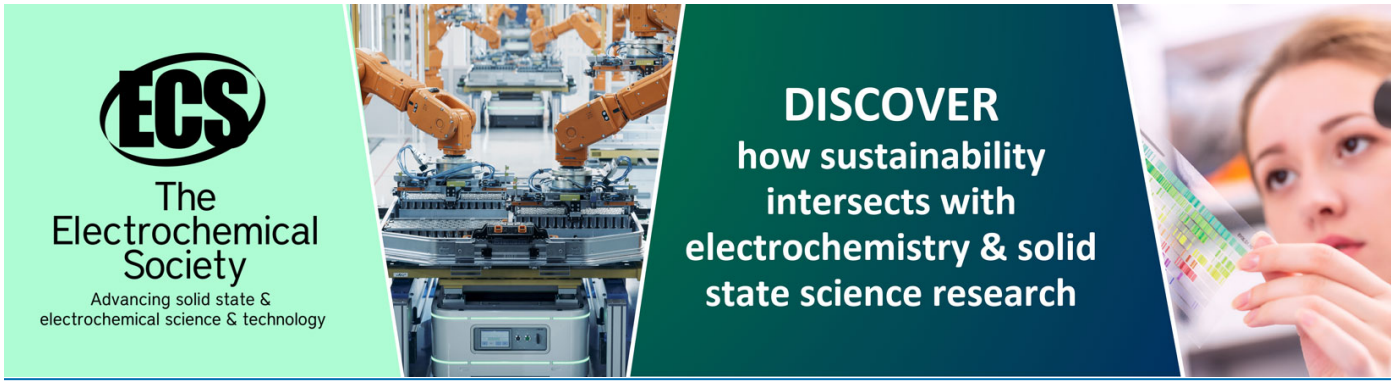
Compression and Rendering of Space Environment Volume Data Based on Improved HVQ Algorithm



Compression and Rendering of Space Environment Volume Data Based on Improved HVQ Algorithm

L L Bao[1,2,*], Y X Cai[1,2], Y M Cui[1,2], S Q Liu[1,2,3] and L Q Shi[1,2,3]

[1] National Space Science Center, Chinese Academy of Sciences, Beijing 100190, China
[2] Key Laboratory of Science and Technology on Environmental Space Situation Awareness, CAS, Beijing 100190, China
[3] University of Chinese Academy of Sciences, Beijing 100049, China

[*]baoll@nssc.ac.cn

Abstract. We propose an improved HVQ approach by combining two detail levels to compress space environment volumes, based on the variation characteristics including smooth variation and significantly positive correlation between variations in the same direction at two different scales. First, the space environment data is divided into 4^3 blocks. Then blocks are decomposed into a two-level hierarchical representation and each block is represented by a mean value and a detail vector. Finally, the detail vectors are encoded by a vector quantizer. During the vector quantization process, PCA-split is applied to compute an initial codebook, and then LBG-algorithm is conducted for codebook refinement and quantization. We take advantage of the codebook-retraining method to speed up the quantization of time series with temporal coherence. Furthermore, we employ the progressive rendering based on GPU for real-time interactive visualization. The results of our experiments prove that the algorithm proposed in this paper can significantly improve the compression rate and decoding speed without sacrificing fidelity in space environment domain.

1. Introduction

Space environment, such as radiation environment, atmosphere environment, magnetopause environment, debris environment, influences ground-based and space-based technological systems that people rely on and effects human life. Volume rendering can illustrate the inner structure characteristics and reveal the overall distribute patterns in an interactive way, and it has been widely applied in space environment domain[1-3]. Because of the development of programmability in graphics hardware, we can achieve real-time volume rendering by the means of volume rendering in hardware[4,5]. However, with the more frequent space missions and more advanced physical models, scientists can simulate many complex space environment phenomena in detail, producing data with tens of variables, hundreds of time steps and several hundred million voxels. Real-time rendering of a complex space environment simulation on GPU is a challenging due to conflict between the limited memory capacity and large data sizes. Compressed Volume Rendering (CVR)[6] can reduce data size effectively by compression and support decompression coupled to rendering. Vector Quantization (VQ)[7] has an asymmetric coding scheme, complex encoding and simple decoding, and is a perfect choice for CVR[6,8]. However, most existing VQ algorithms are concerned on volumes containing blocks with substantial high frequency content[6,9,10], while space environment datasets are gradual.

We describe an improved HVQ algorithm, Hierarchical Vector Quantization With 1 Detail Level (HVQ-1d) that is based on the characteristics of space environment variations.

The paper is organized as follows. In Section 2, we review related work in VQ. In Section 3, we describe an improved HVQ method. In Section 4, we introduce our experiments, then present the compression results and comparison. We draw a conclusion in the Section 5.

2. Representative VQ Algorithms

The representative VQ algorithms include Simple Vector Quantization (SVQ), Hierarchical Vector Quantization (HVQ), Transform Vector Quantization (TVQ) and Residual Vector Quantization (RVQ) [11].

2.1. SVQ

Ning and Hesselink[7,8,12] has introduced VQ technology in compressed volume rendering. SVQ partitions the original volumetric scalar data into n^3 blocks and drives the quantization for these blocks by an appropriate vector quantizer. SVQ algorithm is simple and able to achieve the high compression rate, however, the quality of reconstruction is poor.

2.2. HVQ

HVQ was proposed by Schneider and Westerman to get better reconstruction quality[9,10,13-15]. Specifically, volumetric scalar data is partitioned into 4^3 blocks initially. The blocks are then represented as three levels, one mean level and two detail levels. By the means of the vector quantizer, appropriate mappings and codebooks are computed for the two detail levels. Compared with SVQ, HVQ can obtain better reconstruction quality, however, lower compression ratio.

2.3. TVQ

Fout and Ma[6] presented TVQ based on a volume transform coding scheme adopting Karhunen-Loeve Transform and partitioned vector quantization. TVQ also incorporates a classification in the transform domain to preserve perceptually important features, such as strong gradients and edges. Fout and Ma integrate the classification with vector partitioning and also propose a partial reprojection method, such that TVQ optimizes the encoding while hiding the complexity from decoder. TVQ is an appropriate choice for volumes containing edges or high-gradient regions, however, the variations of space environment are smooth.

2.4. RVQ

RVQ is an extension to SVQ[16]. For RVQ, a set of residual vectors is constructed after the quantization of original vectors. This set is subjected to yet another vector quantization, giving a second pair of index set and codebook. However, the compression time is long, and we cannot accelerate the algorithm by task-parallelism because of the dependence between original vectors and residual vectors.

VQ approaches have been widely applied in the compression of volumes containing blocks with substantial high frequency content. However, space environment datasets hardly contain large variations and the variations in the same direction at two different scales have a significantly positive correlation. Taking advantage of these characteristics, in this paper we propose an improved HVQ algorithm, HVQ-ld which achieves high compression rate, good fidelity and fast decoding while applied to compress and render space environment volumes.

3. Hierarchical Vector Quantization With 1 Detail Level

3.1. Basic idea

The characteristics of space environment data allow us to improve HVQ by combining two detail levels. Let us begin by describing the relation between hierarchical decomposition in HVQ and Haar wavelet transform.

Schneider and Westerman concluded that HVQ achieves fidelity similar to wavelet based compression[13], according to a series of experiments. Obviously, they did not notice the relation between hierarchical decomposition in HVQ and Haar wavelet transform, neither did the following researchers[17,18]. In fact, the mean values of 4^3 blocks are the same as the low frequency coefficients in transform domain after two-level 3D Haar wavelet transform. The 64- and 8-compontent detail vectors in HVQ can be obtained by applying specific orthogonal transforms to the high-frequency coefficients at the first level and the second level of Haar wavelet decomposition, respectively. This relation is illustrated in figure 1 and proved in appendix (Due to the limit of paper length, we take 4^2 block for instance and the proof can be extended to 4^3 block). Therefore, HVQ achieves the same fidelity as partitioned vector quantization based on Haar wavelet, not just similar to wavelet based compression, although HVQ is much simpler.

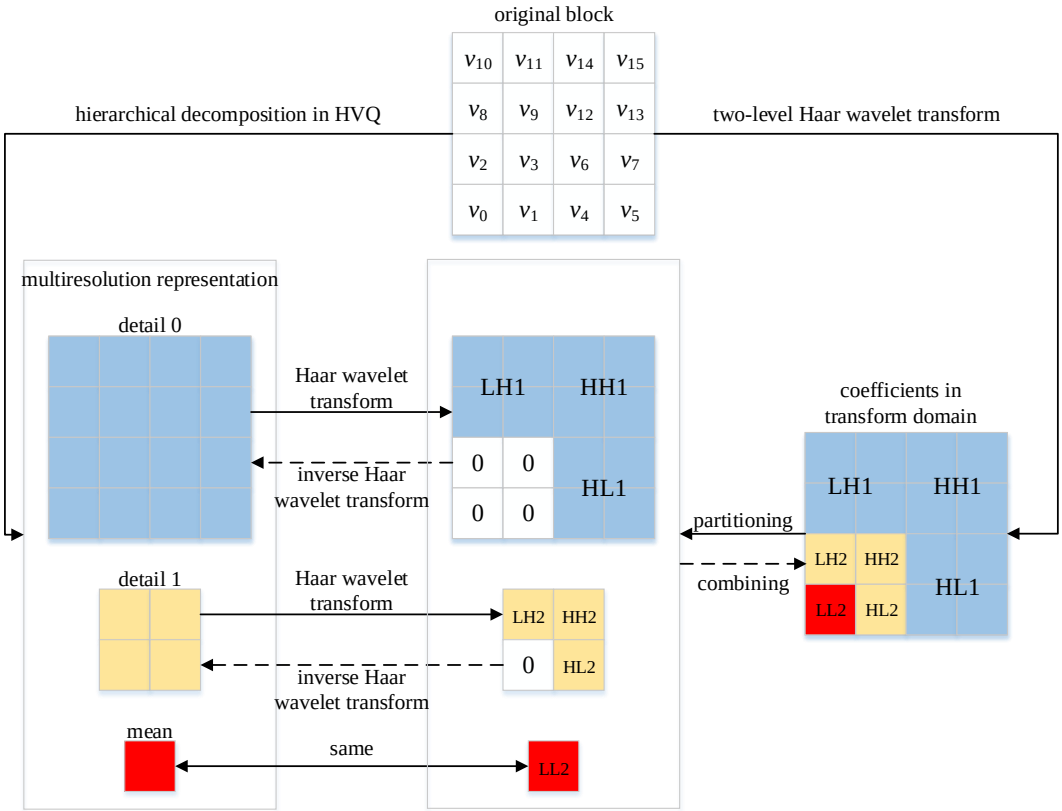


Figure 1. Relation between hierarchical decomposition in HVQ and two-level Haar wavelet transformation (LH1 represents the vertical detail coefficients at the first level of Haar wavelet decomposition, which are obtained by applying low-pass filter and high-pass filter to horizontal axis and vertical axis separately. The definitions of HH1, HL1, LH2, HH2, HL2 and LL2 are similar.)

Partitioned vector quantization in transform domain, integrated with classification, allows us to preserve perceptually important features, such as material boundaries, object surfaces, as well as fluid interfaces[6]. However, space environment data with smooth variations doesn't contain obvious edge features. Furthermore, the space environment variations in the same direction at two different scales have a significantly positive correlation. When we apply partitioned vector quantization based on Haar

wavelet (refer to figure 1) to space environment volumes, two high frequency partitions (blue partition and yellow partition in figure 1) are significantly correlated, so only a very small proportion of all the available codeword combinations is used by reconstruction. Thus in the case of space environment field, two high frequency partitions in vector quantization based on Haar wavelet are completely unnecessary, so are two detail levels in HVQ. We propose HVQ-1d, which combines two detail levels in HVQ, achieving higher compression rate and faster decoding speed on the premise of good fidelity, and also avoiding complicated wavelet transform. In addition, some advanced approaches in TVQ (vector quantization based on wavelet transform), such as Level of Detail (LOD) approach[6, 17], can be incorporated in HVQ and HVQ-1d based on the relation between hierarchical decomposition in HVQ and Haar wavelet transform.

3.2. Framework

The framework of HVQ-1d is illustrated in figure 2. Starting with an original space environment field, the data is divided into 4^3 blocks initially. Blocks are decomposed into a two-level hierarchical representation and each block is represented by a mean value and a detail vector of length 64. The detail vector stores the difference between the original samples and the mean value. Finally, the detail vectors are encoded by a vector quantizer. During the vector quantization process, PCA-split is applied to compute an initial codebook, and then LBG-algorithm is conducted for codebook refinement and quantization[13]. Most space environment fields are non-stationary and for the quantization of time-varying sequence, we employ a codebook-retraining algorithm which speeds up the LBG-relaxations by reusing the codebook of previous time[13].

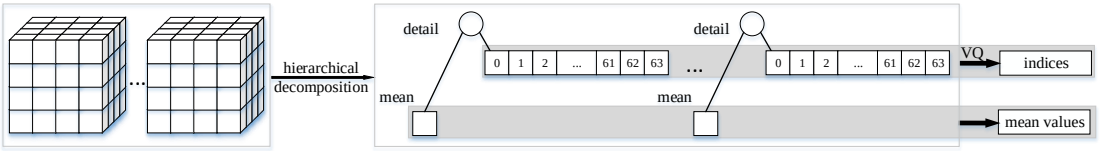


Figure 2. The framework of HVQ-1d

3.3. Compression Rate

For a $N \times M \times K$ volume, we assume that each voxel holds B byte, the mean value and the index in each 4^3 block need 1 byte respectively, the size of codebook is C, and each component in codeword holds 1 byte. The compression rate of HVQ-1d can be computed by:

$$Rate = \frac{N \times M \times K \times B}{2 \times N \times M \times K / 4^3 + C \times 4^3} \tag{1}$$

The compression rate of our approach is about 0.5 times higher than that of HVQ and TVQ, and also slightly higher than RVQ (for the compression rate of HVQ, TVQ, etc., refer to [14,15]).

3.4. Decompression and rendering

Decompression is coupled with rendering on GPU. To decode a particular block, we have to add its mean and the detail information which is achieved by the lookup in the codebook according to its index. Progressive rendering is supported by combining our approach with LOD approach and is advantageous to real-time interactive visualization, especially for quick browsing and for extremely large data sets. During the interaction, only mean values are used to reconstruct low-resolution voxels (low-level model). After the interaction, the detail information is added to the mean values to reconstruct high-resolution voxels (high-level model). This strategy can obtain satisfied reconstruction quality while ensuring interactive rendering frame rate.

4. Results and Comparison

In order to evaluate our work, we compare HVQ-1d with analogous implementations of HVQ, SVQ, TVQ and RVQ. We test these methods on three time-varying space environment volumes, which are *High_Energy_Electron_Flux* (each time step of size 256^3, byte), *Atmospheric_Density* (each time step of size 360×180×100, byte) and *Magnetic_Intensity* (each time step of size 220×160×148, byte). All our experiments were compiled and run under Windows 10 on a 1.80GHz Intel(R) Core(TM) i7-8550U CPU with 20.0GB main memory and an NVIDIA Quadro P500.

4.1. Compression results and comparison

For instance, we take the time step 1st May 2019 at 0:00 UTC, 1st May 2019 at 12:00 UTC and 5th September 2017 at 0:00 UTC out of *High_Energy_Electron_Flux*, *Atmospheric_Density* and *Magnetic_Intensity* respectively, other time steps are similar. The length of each codebook is 256. Peak Signal-to-Noise Ratio (PSNR) is used to evaluate the fidelity. We exclude zero blocks in the calculation to avoid rate inflation and facilitate comparisons. Table 1-3 are compression rate, PNSR and compression time results respectively. Renderings of compressed volumes are illustrated in figure 3-5.

Table 1. Compression rates of HVQ-1d algorithm and four representative VQ algorithms

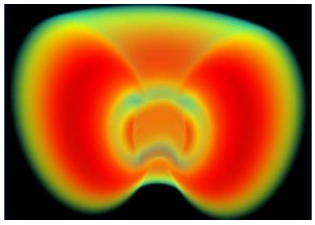
Dataset	HVQ	HVQ-1d	SVQ	TVQ	RVQ
High_Energy_Electron_Flux	19.50 : 1	30.12 : 1	60.24 : 1	17.07 : 1	29.26 : 1
Atmospheric_Density	17.17 : 1	27.54 : 1	55.09 : 1	12.95 : 1	25.75 : 1
Magnetic_Intensity	16.39 : 1	26.64 : 1	53.28 : 1	11.82 : 1	24.58 : 1

Table 2. PSNR values of HVQ-1d algorithm and four representative VQ algorithms (Unit: dB)

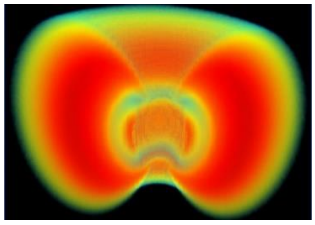
Dataset	HVQ	HVQ-1d	SVQ	TVQ	RVQ
High_Energy_Electron_Flux	33.68	33.10	31.29	32.21	34.91
Atmospheric_Density	56.16	57.02	49.01	49.19	53.97
Magnetic_Intensity	46.18	45.53	42.19	43.11	45.79

Table 3. Compression time of HVQ-1d algorithm and four representative VQ algorithms (Unit: s)

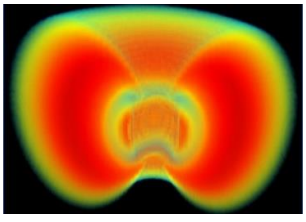
Dataset	HVQ	HVQ-1d	SVQ	TVQ	RVQ
High_Energy_Electron_Flux	93.31	94.89	96.44	81.13	170.84
Atmospheric_Density	94.56	94.07	114.75	78.57	207.93
Magnetic_Intensity	77.62	67.54	87.44	61.57	142.39



(a) Original Data



(b) HVQ



(c) HVQ-1d

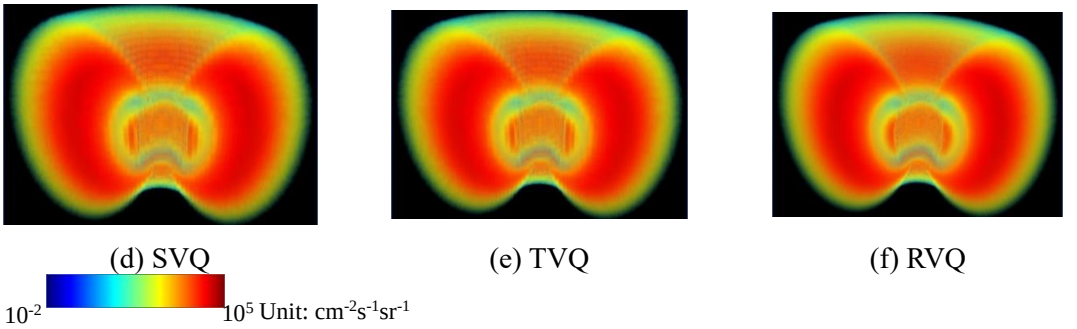


Figure 3. Compression quality comparison of rendering *High_Energy_Electron_Flux* using HVQ-1d algorithm and four representative VQ algorithms

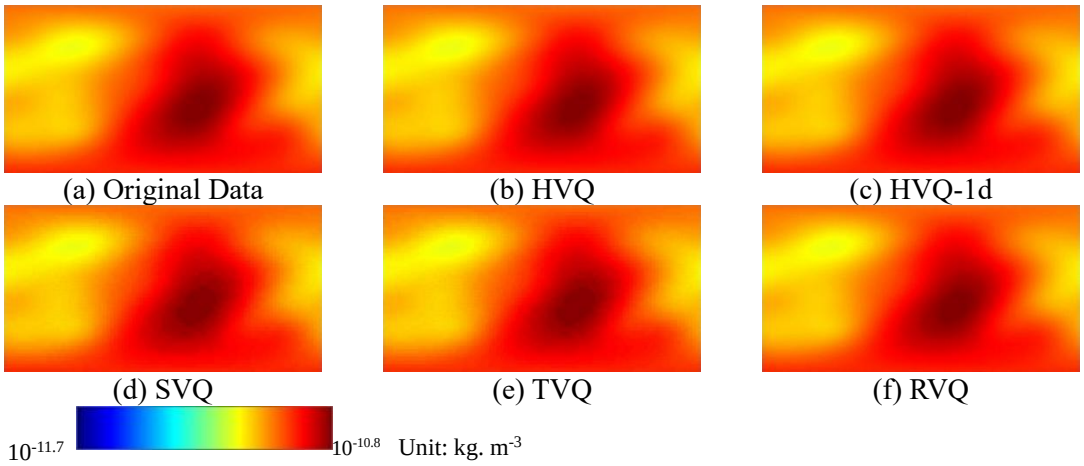
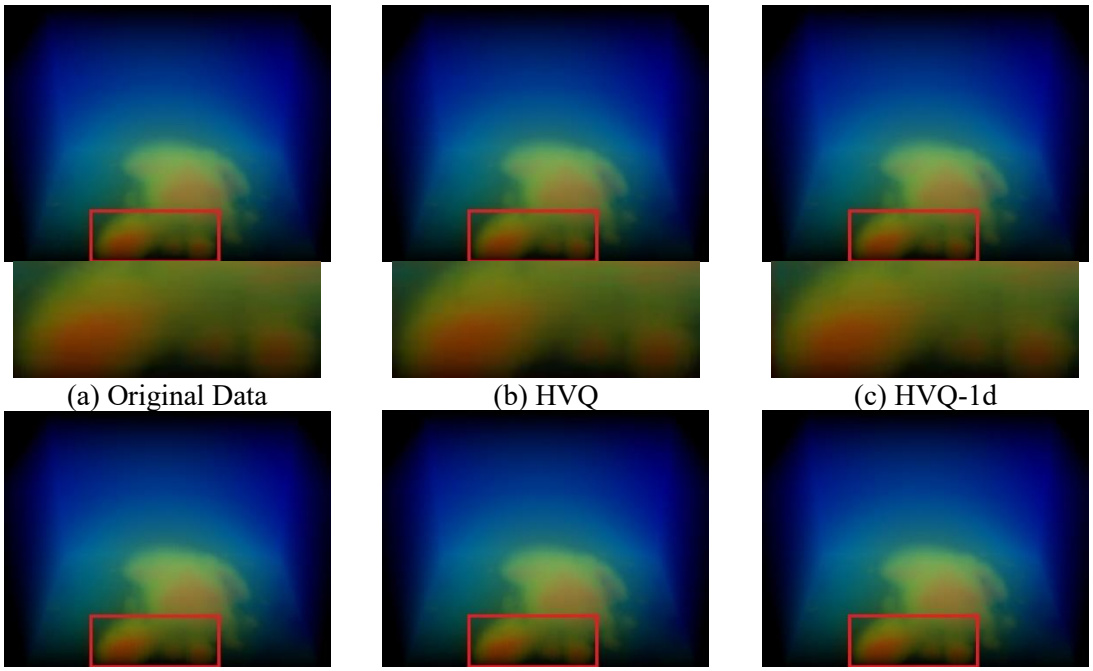


Figure 4. Compression quality comparison of rendering *Atmospheric_Density* using HVQ-1d algorithm and four representative VQ algorithms (The slices at 300km are illustrated)



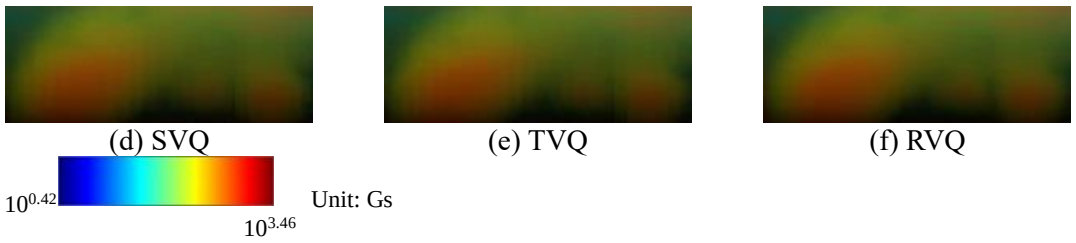


Figure 5. Compression quality comparison of rendering *Magnetic_Intensity* using HVQ-1d algorithm and four representative VQ algorithms (Figures below are the enlarged view of the red framed in the upper figures)

As in table 1, 2 and figure 3-5, HVQ-1d generates a significantly higher compression rate compared to HVQ, while yielding a similar reconstruction quality. As we expect, HVQ loses its quality advantage, because the characteristics of space environment variations result in the low utilization rate of codeword combinations in HVQ, only 2.96%, 6.77% and 4.11% for *High_Energy_Electron_Flux*, *Atmospheric_Density* and *Magnetic_Intensity* respectively. SVQ obtains the highest compression rate, however, at the cost of worst reconstruction quality. TVQ reaches bad reconstruction quality, just better than SVQ, and lowest compression rate. Compared to RVQ, HVQ-1d yields better fidelity of *Atmospheric_Density*, while similar fidelity of *Magnetic_Intensity* and worse fidelity of *High_Energy_Electron_Flux*. However, HVQ-1d achieves higher compression rate and faster encoding speed.

We adopt codebook-retraining algorithm to speed up the quantization of the time step 1st May 2019 at 0:30 UTC of *High_Energy_Electron_Flux* for instance. When this time step is encoded separately. The encoding time is 97.97s and the PSNR is 33.09dB. By retraining the codebook from the time step 1st May 2019 at 0:00 UTC, encoding time decreases to 23.61s and PNSR slightly increases to 33.11dB. Thus codebook-retraining can improve the encoding speed while maintaining the fidelity for a time-series which just contain small variations between two adjacent time steps.

4.2. Decompression and rendering results and comparison

We apply progressive rendering based on GPU to display the volumes compressed by the above VQ algorithms, expect SVQ. The low-level models of HVQ-1d, HVQ, TVQ and RVQ are reconstructed by the first levels (low-frequency levels) and represented as HVQ-1d (L), HVQ (L), TVQ (L) and RVQ (L) respectively. The high-level models of HVQ-1d, HVQ, TVQ and RVQ are reconstructed by all levels and represented as HVQ-1d (H), HVQ (H), TVQ (H) and RVQ (H) respectively. SVQ with single level cannot support progressive rendering. All our examples are rendered using nearest neighbor interpolation, because linear interpolation within codebook is not possible. These results in terms of decoding speed are shown in figure 6-8.

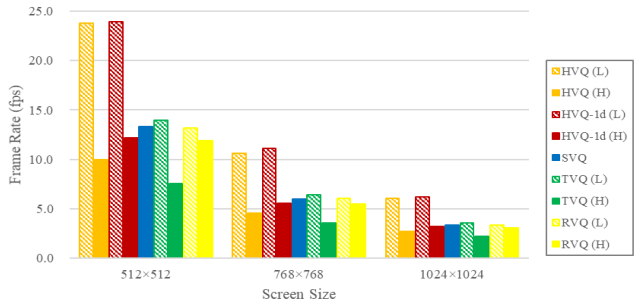


Figure 6. Efficiency comparison of rendering *High_Energy_Electron_Flux* directly from volumes compressed by HVQ-1d algorithm and four representative VQ algorithms

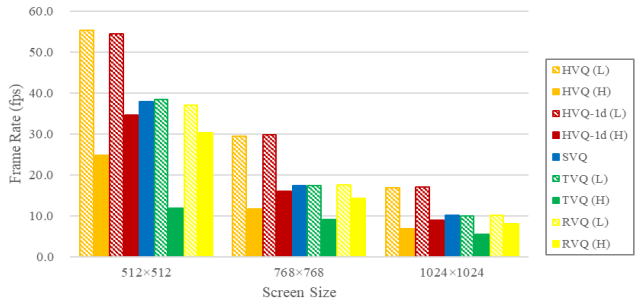


Figure 7. Efficiency comparison of rendering *Atmospheric_Density* directly from volumes compressed by HVQ-1d algorithm and four representative VQ algorithms

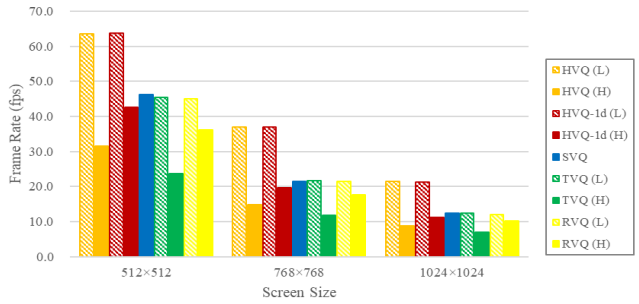


Figure 8. Efficiency comparison of rendering *Magnetic_Intensity* directly from volumes compressed by HVQ-1d algorithm and four representative VQ algorithms

Compared to high-level models, low-level models achieve higher frame rates due to lower complexity. HVQ-1d (L) and HVQ (L) obtain similar decompression and rendering speed, and significantly outperform TVQ (L) and RVQ (L), because both HVQ-1d (L) and HVQ (L) use the mean values directly without the need to look up in a codebook. In all high-level models, HVQ-1d (H) obtains the highest decompression and rendering efficiency because of the lowest dependent codebook reads. The decompression and rendering speed of SVQ is a little faster than that of HVQ-1d (H), however, significantly slower than that of HVQ-1d (L).

5. Conclusion

In this paper we have presented an improved volume compression method by combining two detail levels in HVQ, based on the variation characteristics of space environment, including smooth variation and the significantly positive correlation between variations in the same direction at two different scales. With regard to performance, this method improves compression rate and decoding speed significantly on the premise of good fidelity. For a space environment time-series with smooth variations between two adjacent time steps, we have employed codebook-retraining to speed up the quantization. Furthermore, we have described a progressive rendering method to decode and display the compressed volumes directly on GPU and this method is advantageous for real-time interactive visualization.

6. Appendix

Hierarchical decomposition in HVQ and Haar wavelet transform have the following relation (refer to figure 1): the mean values of 4^3 blocks are the same as the low frequency coefficients in transform domain after two-level 3D Haar wavelet transform. The 64- and 8-compontent detail vectors in HVQ can be obtained by applying specific orthogonal transforms to the high-frequency coefficients at the first level and the second level of Haar wavelet decomposition, respectively. Due to the limit of paper length, we take 4^2 block for instance and the proof can be extended to 4^3 block. The proof is presented below:

The original data is stored in a vector **v** (refer to figure 1):

$$\boldsymbol{v} = \left[v_0, v_1, v_2, v_3, \ldots\right]^T \tag{2}$$

The detail0 vector $\boldsymbol{d_1}$, detail1 vector $\boldsymbol{d_2}$ and mean value can be represented by equation 3, 4 and 5 respectively:

$$\boldsymbol{d_1} = \left[v_0 - \frac{\sum_{i=0}^{3} v_i}{2^2}, v_1 - \frac{\sum_{i=0}^{3} v_i}{2^2}, v_2 - \frac{\sum_{i=0}^{3} v_i}{2^2}, v_3 - \frac{\sum_{i=0}^{3} v_i}{2^2}, \ldots \right]^T \tag{3}$$

$$\boldsymbol{d_2} = [\frac{\sum_{i=0}^{3} v_i}{2^2} - \frac{\sum_{i=0}^{15} v_i}{4^2}, \frac{\sum_{i=4}^{7} v_i}{2^2} - \frac{\sum_{i=0}^{15} v_i}{4^2}, \frac{\sum_{i=8}^{11} v_i}{2^2} - \frac{\sum_{i=0}^{15} v_i}{4^2}, \frac{\sum_{i=12}^{15} v_i}{2^2} - \frac{\sum_{i=0}^{15} v_i}{4^2}]^T \tag{4}$$

$$mean = \frac{\sum_{i=0}^{15} v_i}{4^2} \tag{5}$$

$\boldsymbol{d_{1\_Haar}}$ and $\boldsymbol{d_{2\_Haar}}$ are derived by applying Haar wavelet transform to detail0 and detail1 respectively:

$$\boldsymbol{d_{1\_Haar}} = \left[0,0,0,0, \frac{v_0 - v_1 + v_2 - v_3}{2^2}, \ldots, \frac{v_0 + v_1 - v_2 - v_3}{2^2}, \ldots, \frac{v_0 - v_1 - v_2 + v_3}{2^2}, \ldots \right]^T \tag{6}$$

$$\begin{aligned} \boldsymbol{d_{2\_Haar}} = [0, & \frac{\sum_{i=0}^{3} v_i - \sum_{i=4}^{7} v_i + \sum_{i=8}^{11} v_i - \sum_{i=12}^{15} v_i}{4^2}, \\ & \frac{\sum_{i=0}^{3} v_i + \sum_{i=4}^{7} v_i - \sum_{i=8}^{11} v_i - \sum_{i=12}^{15} v_i}{4^2}, \frac{\sum_{i=0}^{3} v_i - \sum_{i=4}^{7} v_i - \sum_{i=8}^{11} v_i + \sum_{i=12}^{15} v_i}{4^2}]^T \end{aligned} \tag{7}$$

$\boldsymbol{v_{Haar}}$ is derived by applying two-level Haar wavelet transform to original data block and represented as:

$$\begin{aligned} \boldsymbol{v_{Haar}} = [& \frac{\sum_{i=0}^{15} v_i}{4^2}, \frac{\sum_{i=0}^{3} v_i - \sum_{i=4}^{7} v_i + \sum_{i=8}^{11} v_i - \sum_{i=12}^{15} v_i}{4^2}, \\ & \frac{\sum_{i=0}^{3} v_i + \sum_{i=4}^{7} v_i - \sum_{i=8}^{11} v_i - \sum_{i=12}^{15} v_i}{4^2}, \frac{\sum_{i=0}^{3} v_i - \sum_{i=4}^{7} v_i - \sum_{i=8}^{11} v_i + \sum_{i=12}^{15} v_i}{4^2}, \\ & \frac{v_0 - v_1 + v_2 - v_3}{2^2}, \ldots, \frac{v_0 + v_1 - v_2 - v_3}{2^2}, \ldots, \frac{v_0 - v_1 - v_2 + v_3}{2^2}, \ldots]^T \end{aligned} \tag{8}$$

where $\frac{\sum_{i=0}^{15} v_i}{4^2}$ is low-frequency coefficient, which is equal to *mean*.

$\frac{\sum_{i=0}^{3} v_i - \sum_{i=4}^{7} v_i + \sum_{i=8}^{11} v_i - \sum_{i=12}^{15} v_i}{4^2}$, $\frac{\sum_{i=0}^{3} v_i + \sum_{i=4}^{7} v_i - \sum_{i=8}^{11} v_i - \sum_{i=12}^{15} v_i}{4^2}$, $\frac{\sum_{i=0}^{3} v_i - \sum_{i=4}^{7} v_i - \sum_{i=8}^{11} v_i + \sum_{i=12}^{15} v_i}{4^2}$ are high-frequency coefficients at the second level of Haar wavelet decomposition, which are equal to high-frequency coefficients (non-zero coefficients) in $\boldsymbol{d_{2\_Haar}}$. $\frac{v_0 - v_1 + v_2 - v_3}{2^2}, \ldots, \frac{v_0 + v_1 - v_2 - v_3}{2^2}, \ldots, \frac{v_0 - v_1 - v_2 + v_3}{2^2}, \ldots$ are high-frequency coefficients at the first level of Haar wavelet decomposition, which are equal to high-frequency coefficients (non-zero coefficients) in $\boldsymbol{d_{1\_Haar}}$. Proved.

References

[1] Bock A, Pembroke A, Mays M L, et al. Visual verification of space weather ensemble simulations 2015 *Proc. IEEE Scientific Visualization Conference (SciVis)* (Piscataway, NJ) pp 17-24
[2] Hu Y S, Song J J, Shi P, et al. 2016 Study on 3D subdivision mode and encoding in heliocentric coordination system *Chin. J. Space Sci.* **36**(1) 106-16
[3] Pirotti F, Brovelli M A, Prestifilippo G, et al. 2017 An open source virtual globe rendering engine for 3D application: NASA world wind *Open Geospatial Data, Software and Standards* **2** 4
[4] Yang J Z, Zhao D Z, Li W et al. 2010 The research volume rendering algorithm based on GPU *Acta Electronic Sinica* **38**(2A) 202-206
[5] Wang Song, Wang Haiyang, Wu Yadong, et al. 2016 Improved VolumeLIC rendering technology of 3D vector field based on GPU acceleration *J. Comput-Aid. Desig. Comput. Graph.* **28**(5) 723-732
[6] Fout N and Ma K L 2007 Transform coding for hardware-accelerated volume rendering *IEEE Trans. Vis. Comput. Graph.* **13**(6) 1600-07
[7] Wu Z B and Yu J Q 2019 Vector quantization: a review *Front. Inform. Tech. El.* **20**(4) 507-524
[8] Ning P, Hesselink L. Vector quantization for volume rendering 1992 *Proc. 1992 workshop on Volume visualization* (New York, NY) pp 69-74
[9] Ding Z Y, Tan J G, Wu X Y, et al. 2015 A near lossless compression domain volume rendering algorithm for floating-point time-varying volume data *J. Vis.* **18** 147-57
[10] Yu S, Zhang S, Wang K, et al. 2017 An efficient and fast GPU based algorithm for visualizing large volume of 4D data from virtual heart simulations *Biomed. Signal Proces.* **35**:8-18
[11] Wu X Y, Zhou Y Q and Ji Z P. 2015 A survey on volume data compression *J. Comput-Aid. Desig. Comput. Graph.* **27**(1) 26-35
[12] Ning P, Hesselink L. Fast volume rendering of compressed data 1993 *Proc. Visualization'93* (Piscataway, NJ) pp 11-18
[13] Schneider J, Westermann R. Compression domain volume rendering 2003 *Proc. IEEE Visualization, 2003. VIS 2003*. (Piscataway, NJ) pp 293-300
[14] Zhao L P, Xiao D G, Li K L, et al. 2009 An efficient algorithm for large-scale volume data compression and its application in seismic data processing *J. Comput-Aid. Desig. Comput. Graph.* **21**(11) 1606-11
[15] Zhao L P, Yue G X, Xiao D G, et al. 2011 A content-based classified hierarchical vector quantization algorithm for volume compression *Journal of Software* **6**(2) 322-330
[16] Parys R and Knittel G 2009 Giga-voxel rendering from compressed data on a display wall *Journal of WSCG* **17**(1-3) 73-80
[17] Xue J J, Zhao G and Xiao W L 2016 Compression and multi-resolution rendering of sparse voxels based on wavelet *J. Comput-Aid. Desig. Comput. Graph.* **28**(8) 1350-57
[18] Xiao D G, Li L and Zhang Y 2011 Compressed volume rendering for large-scale volume data *Journal of Hunan University (Natural Sciences)* **38**(7) 73-7