A Content-based Classified Hierarchical Vector Quantization Algorithm for Volume Compression

Li-Ping Zhao[1], Guang-Xue Yue[1, 2, 3], De-Gui Xiao[2], Xu Zhou[1], Xiang Yu[1]
[1]School of Mathematics and Information Engineering, Jiaxing University, Jiaxing, Zhejiang, China
[2]School of Computer and Communication Engineering, Hunan University, Changsha, Hunan, China
[3]Department of Computer Science and Technology, Huaihua University, Huaihua, China
Email: zhaoliping_jian@126.com

Fei Yu
Jiangsu Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Soochow, China
hunanyufei@126.com

***Abstract*—An improved volumetric compression algorithm is presented in this paper. Histogram technique is used for analyzing the trait of volume data. The volume data is then partitioned into volume bricks which will be classified into two groups, the blocks with meaningless information as one group(also called empty blocks), and those with meaningful information as the other group(also called object blocks). An efficient hierarchical VQ is applied to compress object blocks while for empty blocks, nothing is saved. Compare with analogous Volume Compression algorithm, experimental results demonstrate the proposed algorithm not only can improve the compression rate significantly on the premise of the good quality of reconstruction image, but also can obtain fast decoding speed.**

***Index Terms*—Vector quantization, volume Classify, Object blocks, Volume compression, GPU**

I. INTRODUCTION

The rapid development of commercial graphics hardware has made it a reality nowadays to render volumetric datasets both in ultra high quality and at interactive frame rates. However, real-time rendering of a large volumetric dataset is still challenging due to the limitations in memory capacity and communication bandwidth (and delay). In the foreseeable future, datasets of further increased scale will be produced from the burst of data acquisition devices and numerical simulations [1]. Neither the memory capacity nor the communication channel, unfortunately, is expected to improve to such a level that the bottlenecks can be cleared.

Accordingly, CVR (Compressed Volume Rendering) [1-7], with the principle of coupling compression to rendering, has been proposed and has shown to be a practical solution for the issue of real-time rendering. Considering that a volumetric dataset has noticeable redundant information, the dataset is firstly compressed before being put into the memory or a GPU (Graphics Processing Unit), and then decompression and rendering can be done simultaneously at the GPU because it is equipped with some computing components. For a GPU, its SIMD (Single Instruction Multiple Data) architecture and parallel execution model, nevertheless, constrain the compression method that can be applied in CVR.

VQ (Vector Quantization) [2-6] is an ideal choice as an asymmetric coding scheme for CVR. In spite of the fact that the encoding of VQ is complex, the decoding is simple as it is essentially a single table look-up procedure. VQ has first been adopted in volume rendering for the purpose of compression by [2]. In the recent years, various VQ approaches have been applied in CVR domain, including HVQ (Hierarchical VQ) [3], CVQ (Classified VQ) [1, 4-5], TVQ (Transform VQ) [4] and their combined algorithms, such as CTVQ (Classified Transform VQ) [4]. However, most existing VQ for CVR are Pixel-based compression algorithm [1-5]. MEPG-4 firstly introduced the content-based video/image compression algorithm. The key idea of MEPG-4 is that a scene can be divided into different AVOs (Audio Visual Object), for example the foreground AVOs and background AVOs, and which of those AVOs use different encoding methods. In this way, the objective of high compression ratio can be obtained on the premise of good quality of reconstruction image.

Inspired by the ideas of MEPG-4, in this paper we present a content-based volumetric compression algorithm which uses classified hierarchical VQ, hereafter abbreviated to CCHVQ (Content-based Classified Hierarchical VQ). This work extends our previous FCHVQ [5]. Compared with FCHVQ, firstly the histogram technique is used for analyzing the traits of volume data; secondly the classification scheme is content-based, in other words, total blocks are divided into two classes, one is empty blocks class (background) and the other is object blocks class (foreground); at last, different classes have different coding strategies. Specifically, our algorithm has the following particular properties:

- Memory efficiency: Note that the empty blocks is the meaningless part of the volume data, we set a flag for those blocks and save nothing for reconstruction. Many experiments show that more than 60% of blocks are empty blocks. Object

doi:10.4304/jsw.6.2.322-330

blocks are then vector quantized to achieve higher compression rate.

- Decoding efficiency: Different classes decode its different ways. Our decode speed of volume data can be faster as we waste no time on empty blocks. When decoding and rendering in GPU in future work acceleration techniques for GPU-based volume rendering, for example, empty space leaping [8, 9] can be well used.
- Fidelity efficiency: The ultimate aim to volume rendering is to gain some meaningful information about the dataset. Our classified strategy let us focus on the object blocks. So, more details of object blocks can be retained to get fidelity efficiency.

In the following, we present related work in the context of compression for volumetric datasets (Section 2) and provide a rather brief introduction in the improved classical hierarchical VQ methods in Section 3. In Section 4, we introduce the examined datasets and present our compression results and Comparison. Finally, we conclude our paper in Section 5.

II. RELATED WORKS

VQ maps every *k*-dimensional input vector *x* to some reproduction vector x_i selected from a finite codebook of *M* candidate vectors (i.e., code words), and encodes *x* by the index *i* of x_i. If M is small, then each index only requires a few (i.e., log_2M) bits, thereby achieving the goal of compression. Dividing the size of original data by the size of compressed data, the result gotten is called the compression rate. Note that the size of compressed data is comprised of two parts, the size of indices (denoted by C_{index}), and the size of a codebook (denoted by $C_{codebook}$). For example, given a volumetric dataset of $N \times M \times K$ points, assuming that each point holds *B* bytes, the compression rate of VQ can be calculated by Eq. (1).

$$VQ\_Rate = \frac{N \times M \times K \times B}{C_{index} + C_{codebook}} \quad (1)$$

Existing works on VQ mainly focus on increasing the compression rate, reducing distortion, and reducing the algorithm complexity [10]. For most existing algorithms, improvement on the compression rate tends to result in worse quality of the reconstructed image. In other words, better reconstructed image is often at the cost of the compression rate [3] and the algorithm complexity [4]. Detailed literature studies on VQ will be presented in the following.

A.Hierarchical VQ

Hierarchical VQ was described by Schneider et al [3] to get better image reconstruction quality. Specifically, starting with the original scalar field, the data is initially partitioned into disjoint blocks of size 4^3. Each block is decomposed into a multi-resolution representation, which essentially splits the data into three different triadic frequency bands. Therefore, each block is down-sampled by a factor of two by averaging disjoint sets of 2^3 voxels each. The difference between the original data samples and the respective down-sampled value is stored in a 64-component vector. The same process is applied to the down-sampled version, producing one single value that represents the mean value of the entire block. The 2^3 difference values carrying the information that is lost when going from 2^3 mean values to the final one are stored in an 8-component vector. Finally, a 1-component vector stores the mean of the entire block. In performing this task, the data is decomposed into three vectors of length 64, 8, and 1, respectively, which hierarchically encode the data samples in one block. In this way, HVQ, can reach good fidelity, however, much lower compression rate because each block should store the mean value of the block and two indices in order to reconstruct the whole block value.

Also, consider a volume with $N \times M \times K$ data points and each point holds *B* bytes. The block size is $n \times n \times n$, and the down-sample block size is $(n/2) \times (n/2) \times n/2$, the compression rate of HVQ can be computed by (2). Because the codebook capacity is the same in HVQ 、 FCHVQ and CCHVQ, so we just note by $C_{codebook}$. For compassion convenience, size of codebook is 256.So each index need 1 byte (log_2256 ,8 bits).

$$HVQ\_Rate = \frac{N \times M \times K \times B}{3 \times N \times M \times K/(n \times n \times n) + C_{codebook}} \quad (2)$$

B. Flag Based Classified Hierarchical VQ

Regarding image compression, it should be pointed out that some of the blocks contribute more to the visual quality of the final image, while the others make little contributions [4]. To address this issue, all the blocks need to be classified into different groups, with each group being quantized separately, thereby guaranteeing their presence in the code vector population. Classification of blocks is more important in the case of volumetric compression due to arbitrary nonlinear mapping of the transfer function. In addition, classification of blocks allows the encoder to adapt to a volumetric dataset containing various perceptually important features (e.g., surfaces, boundaries, etc.). Therefore, most literature works adopted the idea of classification for the goal of obtaining better performance[2,4].

FCHVQ (Flag based classified hierarchical VQ) [5] was described by us to get higher compression rate and faster decoding speed in addition to good fidelity. In particular, the volume data set is first divided into smaller regular blocks and then divided into two classes according to whether its average gradient value is zero or not. We focus on the Classification strategy of FCHVQ algorithm. Clearly, if the block with data that they have the same values, it will belongs to the zero average gradient class. While the block with data that they have much different value, it will belongs to the non-zero average gradient class. Suppose the block size is $2 \times 2 \times 2$, then there are 8 data values in a block. See Fig.I for an illustration of the classification strategy.

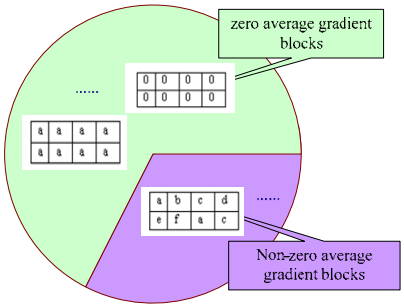


Fig.I the classification strategy of FCHVQ algorithm

Many experiments show that the gradient values of non-zero blocks always occupy a much more proportion of the total blocks. Accordingly, suppose the number of blocks that hold the non-zero gradient value notes $N_{g!=0}$, and the number of blocks that hold the zero gradient value notes $N_{g=0}$, so in fig. I, the area of $N_{g=0}$ is larger than that of $N_{g!=0}$. With the same preconditions as HVQ, suppose a volume with $N \times M \times K$ data points and each point holds *B* bytes. The block size is $n \times n \times n$, the down-sample block size is $(n/2) \times (n/2) \times n/2$, the compression rate of FCHVQ can be computed by (3).

$$FCHVQ\_Rate_{comp} = \frac{N \times M \times K \times B}{9 \times N \times M \times K / (2n \times 2n \times 2n) + 2 \times N_{g!=0} + C_{codebook}} \quad (3)$$

In the context of FCHVQ, different compression methods to different classes were then introduced in order to make FCHVQ algorithm efficient. However, noticing that the blocks that hold the zero gradient value have two situation: one is the blocks that hold the same value *a(a!=0)*, the other is the blocks that hold the same value of 0. So, FCHVQ is also Pixel-based compression algorithm. What's more, another limitation was, Pixel-based compression algorithm would not be coupled to the acceleration techniques of rendering In GPU because of its SIMD architecture. In the light of this, CCHVQ will be introduced in detail in next section.

III. CONTENT-BASED CLASSICAL HIERARCHICAL VQ

CVR, generally speaking, can be divided into three classes of methods based on the location of decompression and rendering, respectively software CVR, hardware CVR and hybrid CVR[4]. For software CVR, Ning and Hesselink [2] argued that the decompression should allow fast, direct, random access to voxels. For hardware CVR, Fout and Ma[4] recognized two more desired traits: one is compact, separable decompression; the other is uniform decompression. These demands only focus on the rendering speed, for the decompression is coupled to rendering in CVR domain. However, one more important idea should keep in mind is that volume rendering is a method of extracting meaningful information from volumetric data[11]. In addition to the decompression restrict ,the characteristics of the volumeric data should be coupled to the compression and rendering. This is even more important when we take the classified VQ for a solution of compression as classification can identify the characteristics of the volumetric data. So, we recognize three more desired traits that specifically target classified VQ.

1. Classified method should be directly relating to the common traits of the volume data. For the object of the compression is to save more meaningful information about the volume date using less bits.
2. Different classes can held their own compression scheme .With the efficiently of today's GPU, bits allocation can be more flexibility, different classes can held their own compression scheme so that leave more codewords to express the detail of the reconstruction image.
3. Classification scheme should be coupled to the acceleration techniques of rendering: The final goal of CVR are faster rendering speed and better image fidelity. To get that goal, Classification scheme should be considered with the acceleration techniques of rendering, for example empty space leaping.

These above constraints associated with these objectives put more emphasis on the relations among the volumetric data characteristics 、compression algorithm and rendering, not only the relations between compression algorithm and rendering. That's to say, before compression, we may extract the meaningful information using certain scheme. By this way, we can get higher compression rate for that the blocks with useless information will be discarded and better reconstruction image quality for that total codewords will leave to save the detail of the image.Under this background, this paper presents an content-based volume data compression algorithm. The procedure of the algorithm is as follows: During the data pre-process stage, statistic the characteristics of the data itself using histogram technique; From the histogram info , classify the blocks which divided from total volume data into two groups, the blocks with meaningless information as one group(also called empty blocks), and those with meaningful information as the other(also called object blocks);Only object blocks are decomposed into a three hierarchical representation manner and vector quantized in order to leave more details of the reconstructed image [3,5]. The overall algorithm is illustrated in Fig.II.

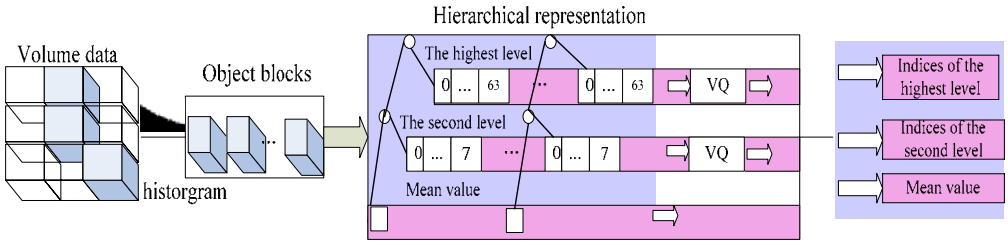


Figure II. Improved classical hierarchical VQ

A. Trait of volume data

Volume rendering [11] is the process of projecting a volumetric three-dimensional dataset onto a two-dimensional image plane in order to gain some meaningful information about the dataset. A volumetric data set is typically a set V of samples (x, y, z, v),also called voxels, representing the value v of the some property of the data, at a 3D location (x, y, z).If the value is simply a 0 or an integer i within a set I ,with a value of 0 indicating background and the value of i indicating the presence of an object o_i ,then the data is referred to as binary data[11]. So, before using classification scheme, to investigate the characteristic of the volume data is needed. Historgram Info, a simple but useful way , to be used by us to get the distribution of the volume data. Taking the large size of the volume data into account, we firstly resample the volume data(uniformly resample from *x*,*y*,*z* direction) in order to reduce the algorithm complexity. If the resample data is is high dynamic range, for conventional, the volume datasets should be normalized to 0~255 according by (4) firstly. Here V_s' is the volume data value after normalization. V_s presents the resample volume value. $V_{\min}$ holds the minimum value of the resample data, while $V_{\max}$ holds the maximum value.

$$V_s' = \frac{V_s - V_{\min}}{V_{\max}} \times 255 \qquad (4)$$

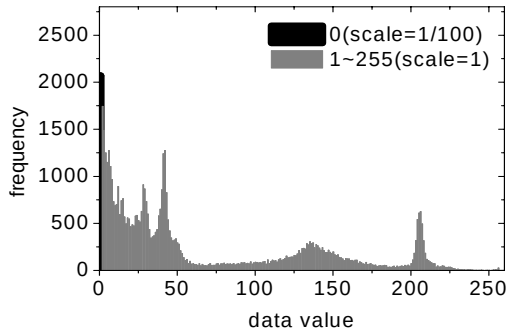
For compassion convenience, the datasets used for experiments are the same as that used in FCHVQ, which list in TABLE I Noting that bigger datasets can be divided into smaller $256\times256\times256$ volume[4] firstly, and then composite them to the whole datasets by certain rules.

For all datasets, the resample data size is $64\times64\times64$. Figure III shows the histogram info of the volume data. Here, the x-axis of histogram is the data value, while the y-axis is frequency of each value(0~255) in the normalization resample volume data set.

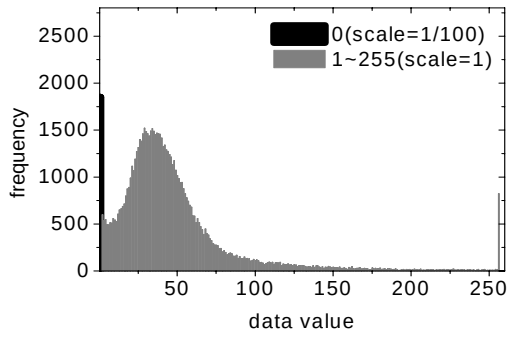
TABLE I. DATASETS USED FOR EXPERIMENTS

Volume data	Bits	Resolution	Size
Bonsai	8	256×256×256	16MB
foot	8	256×256×256	16MB
aneurism	8	256×256×256	16MB

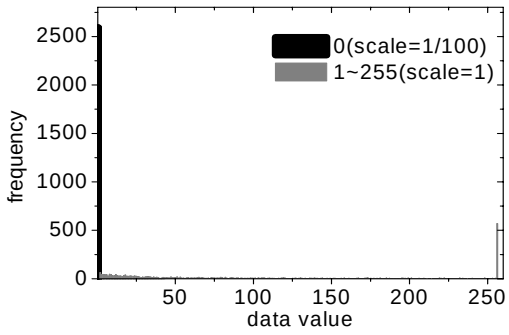
Notice that the frequency of value 0 scale to 0.01 in figure III. So,from the histogram info of the resample volume data, we can see the distribution of the volume data. Here, at least two traits of the volume data can be concluded .One is that different volume data have different data distribution, while the other is that different volume data have the same phenomenon which is different volume data holds a lots of value of 0 that indicating background.



a) bonsai histogram



b)foot histogram



c) aneurism histogram

Figure III. Histogram info of the volume data

Those traits of the volume data will provide the prior information for compression. The volume data is then partitioned into volume bricks($4 \times 4 \times 4$) [3,4,5]which will be classified into two groups and different groups take different compression scheme.

B. Content-based Classification

An important conclusion of the volume data characteristic is that the useful object information just occupy a little proportion of the volume data while the background proportion takes more proportion. It is not unique, [8] presented that the volumetric data set can be divided into two classes: one class with the empty blocks and the other class with the object blocks. And empty blocks made no contribution to the final image , so empty space leaping can be used for acceleration of rendering but do not influence the image fidelity. We also take this classification method before compression.

Two advantages can be concluded when take this classification scheme: Firstly, the classification scheme for compression that the most of capacity for keeping empty blocks should be saved. We just set a flag to the empty blocks . Secondly, the compressed volume data rendering can take the advantage of the empty space leaping acceleration technique. We then give the brief specification of our classification scheme, and the differences of the classification scheme between FCHVQ and CCHVQ.

For classification, we firstly get the mean value of each block $Mean(B_i)$. Here n is the block size and $V(x, y, z)$ is the value of volume data in location(*x*,*y*,*z*).

$$Mean(B_i) = \frac{1}{n \times n \times n} \sum_{x=x_i}^{x_i+n} \sum_{y=y_i}^{y_i+n} \sum_{z=z_i}^{z_i+n} V(x, y, z) \qquad (5)$$

Our Classification scheme is very easy: if the $Mean(B_i)$ equals 0, then the block belongs to empty block(noted as B_{empty}), otherwise the block belongs to object block(noted as B_{object}).

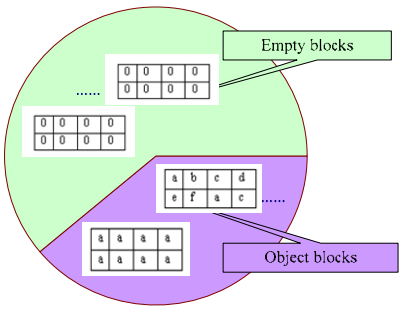


Figure IV.the classification strategy of CCHVQ

From Fig.I and Fig. IV, the differences of the classification strategy of the two algorithms illustrate clearly. Noting the white area in Fig.I is bigger than that in Fig.IV, which can be proved by Lemma 1. Suppose N_{empty} is the number of empty blocks while N_{object} is the number of object blocks. N_{empty} plus N_{object} is the total number of blocks of volume dataset.

Lemma 1. When Using the same block size of the same volume data in CCHVQ and FCHVQ , $N_{g=0} > N_{empty}$.

Proof.

Note that $N_{g=0}$ is the number of blocks that hold the zero average gradient value [5]. The block that has the zero average gradient value, in other words ,the data in one block have the same value, just as illustrated in Fig.I. Those blocks can be classified into two classes: one class is that the same value is 0(also called empty block), and the other is the same value a(1~255). That's to say, empty blocks are part of blocks those hold zero average gradient value. So, $N_{g=0} > N_{empty}$.

Proved.

We list the detail classification results of the classification strategy among different volume data in TABLE II, including $N_{g=0}$, N_{empty} , $N_{g=0}$, $N_{g!=0}$ and the proportion of N_{empty} of total blocks, which can be computed by $N_{empty}/(N_{empty} + N_{object})$. From table II, we can see N_{empty} takes the most proportion of the total blocks, in particular for aneurism volume dataset, there are 97.03% blocks are empty blocks. One more fact is that for the testing volume datasets, the difference between $N_{g=0}$ and N_{empty} is very small. The compression method of $N_{g=0}$ is just to save the mean value in FCHVQ for reconstruction, while in CCHVQ, higher compression rate can be obtained because we just focus on the object blocks while for empty blocks, nothing is saved.

TABLE II. COMPARISON OF THE CLASSIFICATION STRATEGY

volume data	CCHVQ			FCHVQ	
	N_{empty}	N_{object}	$\frac{N_{empty}}{N_{empty}+N_{object}}$	$N_{g=0}$	$N_{g!=0}$
bonsai	185154	76990	70.63%	187831	74313
aneurism	254361	17783	97.03%	255800	6344
foot	179883	82261	68.62%	180291	81853

C.Object blocks encoding

As mentioned above, the compression scheme of our proposed algorithm is to take different ways to different classes. For object blocks , an efficient hierarchical VQ is applied in order to get higher compression rate on the premise of the good quality of reconstruction image . Specifically, object blocks are decomposed into a three hierarchical representation manner like the way used in [3,5] and then the vectors of different hierarchical use vector quantization to compress.

As for any VQ, the process generally can be divided into three aspects: code book design, codeword search, indexed allocation [10]. Among these steps, code book design plays an important role in the performance of the algorithm. LBG-algorithm developed one of the earliest vector quantization algorithms suitable for practical

applications[12]. Because of the sensitivity to the initial codebook in LBG, so far many optimized algorithms [13,14] have been proposed to improve the code book design.

For compassion convenience, the improved VQ used in this work(see FigureV)also takes the splitting scheme based on a principal component analysis[14,15,16]. Specifically, a splitting scheme based on a principal component analysis (PCA) is used by us to find an initial codebook which is then refined by LBG algorithm, finally, we restrict the search to the k-neighborhood of the initial cell in the quantization stage. The difference between VQ based on PCA split used in HVQ and FCHVQ should be pointed out is that only the small part of object blocks are trained in VQ. Thus, we do not only save a large amount of computation, but also can get better reconstruction image quantity because of leaving the whole code words to the object blocks.

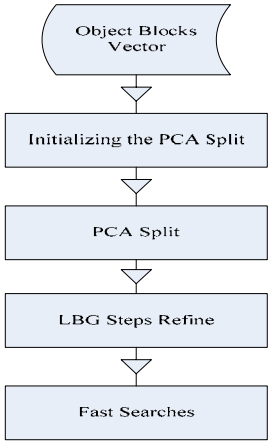


Figure V. VQ base on PCA split

D. Compression rate and Comparison

After compression of object blocks ,the original volume dataset can be reconstructed by the codebooks, indexes, mean value of the object blocks and the flags of each block which are used for identifying the different classes. So, considering a volume of $N \times M \times K$ data points, each point holds B bytes, the block size of is $n \times n \times n$ and the down-sample block size of hierarchical representation is $(n/2) \times (n/2) \times n/2$. The compression rate of CCHVQ can be computed by (6).

$$CCHVQ\_Rate_{comp} = \frac{N \times M \times K \times B}{N \times M \times K/(2n \times 2n \times 2n) + 3 \times N_{object} + C_{codebook}} \quad (6)$$

Here, $N \times M \times K/(n \times n \times n \times 8)$ is the capacity of the flags. And $C_{codebook}$ presents the capacity of the codebook. As for each object block, we should store the mean value, the indexes in the highest level and the second level, so $3 \times N_{object}$ are needed.

Lemma 2. When the number of empty blocks is more than 4.17 percent of the total number of blocks, in other words,

$$N_{empty}/(N_{object} + N_{empty}) > 1/24,$$
$$CCHVQ\_Rate_{comp} > HVQ\_Rate_{comp}.$$

Proof.

If $CCHVQ\_Rate_{comp} > HVQ\_Rate_{comp}$, then

$N \times M \times K/(2n \times 2n \times 2n) + 3 \times N_{object}$ should be less than $3 \times N \times M \times K/(n \times n \times n)$.

Then $N_{object} < 23 \times (N \times M \times K)/24 \times (n \times n \times n)$,

add N_{empty} on both sides of the equation, get:

$N_{object} + N_{empty} < N_{empty} + 23 \times N \times M \times K/24 \times (n \times n \times n)$,

Because $N_{object} + N_{empty} = N \times M \times K/n \times n \times n$,

So $N_{empty} > N \times M \times K/(24 \times n \times n \times n)$,

divided by $N_{object} + N_{empty}$ on both sides of the equation, get:

$N_{empty}/(N_{object} + N_{empty}) > 1/24$.

So, if $N_{empty}/(N_{object} + N_{empty}) > 1/24$,

then $CCHVQ\_Rate_{comp} > HVQ\_Rate_{comp}$.

Proved.

Considering empty blocks in the volumetric data always occupies a certain percentage, many experiments show that empty blocks is much more than 4.17 percent of the total number of blocks. See N_{empty} in the above table II , there are more than 60 percentage of the blocks are empty blocks, that's the key point why our algorithm can be more memory efficient than that of HVQ.

Lemma 3. If $N_{g=0} \approx N_{empty}$,then

$$CCHVQ\,Rate_{comp} > FCHVQ\,Rate_{comp}.$$

Proof.

If $CCHVQ\_Rate_{comp} > FCHVQ\_Rate_{comp}$,

then $3 \times N_{object}$ should less than $N \times M \times K/n \times n \times n + 2 \times N_{g!=0}$,

Because $(N_{g=0} + N_{g!=0}) = N \times M \times K/n \times n \times n$

and $(N_{empty} + N_{object}) = N \times M \times K/n \times n \times n$

So , $3N_{empty} - 2N_{g=0} > 0$

Here, if $N_{g=0} \approx N_{empty}$,so $2(N_{g=0} - N_{empty})$ is small,

So, $N_{empty} - 2(N_{g=0} - N_{empty}) > 0$,

Then $CCHVQ\_Rate_{comp} > FCHVQ\_Rate_{comp}$.

Proved.

From Lemma 2 and Lemma 3, we demonstrate the memory efficient of CCHVQ. We will show the detail compression performance(include concrete compression rate, the reconstructed image quality, the encode and decode complexity) of CCHVQ , HVQ and FCHVQ in next section, results and comparison.

E. Decoding

The decoding algorithm is similar like FCHVQ which to use the saved compressed information to reconstruct the image block by block. Firstly, get the flag of the reconstructed block . If the flag is zero, we just skip that blocks. If not, we should first get the mean value of the block, and then get the difference between the mean value of the block and data in the down-sample block according to the index of second level, finally, get the difference between each data in the block and that in down-sample block according to the index of highest level. The sum of the value got in the previous steps is the reconstructed value in the object block.

Different from decoding of FCHVQ, for empty blocks , we just skip that blocks ,while in FCHVQ for those blocks whose average gradient values are zero, we need replace their whole block data with their mean values. Evidently, our method is faster than FCHVQ. What's more, when decompress in GPU, for empty blocks, we just discard that blocks for that these blocks make no contributes to the final reconstructed image. So, acceleration techniques for GPU-based volume rendering [9],for example, empty space leaping can be well used.

IV. Results and Comparison

The performance of VQ is usually measured by the compression rate, the reconstructed image quality and the encode and decode complexity. In this work ,the compression rate is measured by equation(1),while the reconstructed image quality is evaluated by the peak signal to noise ratio (PSNR)[10]. Considering a volume of $N \times M \times K$ data points,the value of the raw data is x_{ijk} , L is the maximum of the volume data， usually is 255.And the value of the reconstructed data is y_{ijk} .Here $0 \le i \le N-1, 0 \le j \le M-1, 0 \le k \le K-1$, then MSE and the PSNR can be defined as follows.

$$MSE = \frac{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\sum_{k=0}^{K-1}(x_{ijk} - y_{ijk})^2}{M \times N \times N}$$

$$PSNR = 10 \times \log_{10}\frac{L^2}{MSE} \quad \text{db}$$

For comparison, the encode and decode complexity is simply using the running time. All the experiments results have been obtained by programming with C++/VS2005 on 2.0GHz Inter(R) core2 Duo CPU with 2GB main memory under Window XP.

A. Encoding results and comparison

In order to provide a context for the evaluation of our work, we compare our approach (CCHVQ) with analogous implementations of FCHVQ and HVQ.

The comparison of the compression rate and reconstructed image quality of CCHVQ ,FCHVQ and HVQ among different volume data illustrates in Fig.VII and Fig.VIII. Here , the number of codeword in the codebook is 256.

From Fig.VII, CCHVQ algorithm can get much higher compression rate than that obtained from HVQ and FCHVQ. Especially for aneurism volume data , the compression rate of CCHVQ is almost 9 times more than that of HVQ and almost 3 times more than that of FCHVQ. And for bonsai and foot ,the compression of CCHVQ is nearly 2 times more than that of HVQ. Our memory efficient is not at the cost of the reconstructed image quality, on the contrary, the PSNR obtained from CCHVQ(see Fig. VIII) is about 0.1~0.2 higher than that of FCHVQ and 0.2~0.4 higher than that of HVQ.

For the same volume data, take aneurism for example, we use different codebook sizes, CCHVQ also can obtain efficient memory and better reconstructed image quality than other algorithms. For example ,when the codebook size is 128, the compression rate and the PSNR obtained from CCHVQ, FCHVQ and HVQ are respectively 268.80,53.22,22.69. While the PSNR obtained from CCHVQ,HVQ and FCHVQ are respectively 35.46db, 35.36db and 35.34db.

B. Decoding time and encoding time comparison

At the same time ,our decode algorithm runs faster than that of FCHVQ and HVQ. Noting the we skip the empty blocks, so no time waste on those bolocks. See Figure VI for the detail comparison of the decode times among different volume datasets and different algorithm.

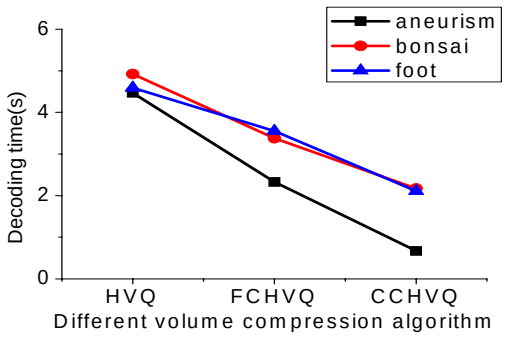
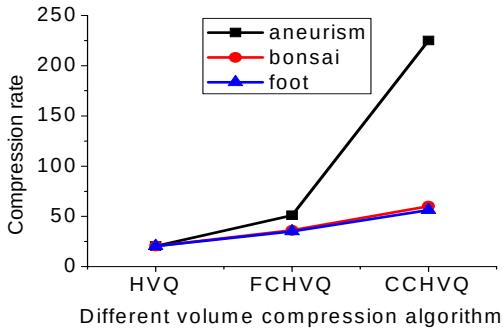


Figure VI .the decode times among different algorithm

However, our proposed algorithm is a little more time-consuming than that of HVQ and FCHVQ. But exclude the getting histogram info process, the encode time is a little lower than FCHVQ, but also higher than HVQ. But In the CVR domain, compression can be slow since it is performed only once offline[4].

V. Conclusions and Future Works

We have devised a content-based classified hierarchical VQ to obtain memory efficiency, better reconstructed image quality and faster decoding. Since volume rendering is the process of gaining some meaningful information about a dataset, our content-based classification scheme before compression can classify which blocks is the meaningless information (i.e., empty blocks), hence no space and no time will be wasted on these blocks. While for the object blocks, a three hierarchical representation manner and a PCA-based split



FigureVII. Compression rate comparison among different algorithm

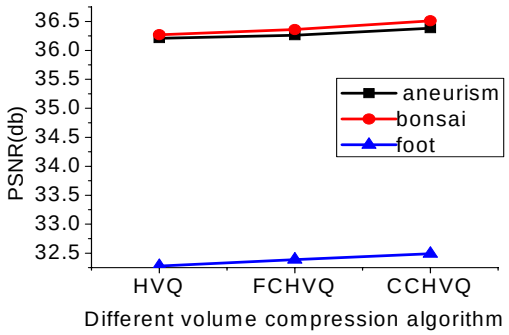


Figure VIII. PSNR comparison among different algorithm

VQ have been employed in order to get better reconstructed image quality.

We then compare our approach (CCHVQ) with analogous implementations of FCHVQ and HVQ. The experimental results also demonstrate the improvement of compression rate, reconstructed image quality and decoding speed. Compression rate, reconstructed image quality and decoding speed are the the principal causes in CVR domain.

Although we have not done decoding and rendering on GPU, our classification scheme should be coupled to the acceleration techniques of rendering, such as empty space leaping. In the future, we are planning to investigate the relations between classified VQ and the transfer function and choose more advanced rendering algorithms to make volume rendering more meaningful and faster.

ACKNOWLEDGMENT

This work was supported by the Project of National Natural Science Foundation of Zhejiang province under grant No: Y1080901, Natural Science Foundation of Hunan province under grant No: 07JJ6140, 07JJ6109, 05FJ3018.

REFERENCES

[1] F. Roland, B. Michael, S. Marc. “Sequential Data Compression of Very Large Data in Volume Rendering”. *Conference on Vision, Modeling, and Visualization*,Saarbrücken,Germany,2007

[2] P. Ning and L. Hesselink, “Fast Volume Rendering of Compressed Data,” *IEEE Conference on Visualization*, San Jose, CA, 1993

[3] J. Schneider and R. Westermann, “Compression Domain Volume Rendering,” IEEE Conference on Visualization, Seattle, WA, 2003

[4] N. Fout and K. L. Ma, “Transform Coding for Hardware-Accelerated Volume Rendering,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1600-1607, 2007

[5] L. P. Zhao, D. G. Xiao, and K. L. Li, “An Efficient Algorithm for Large-Scale Volume Data Compression and Its Application in Seismic Data Processing,” in Chinese, *Journal of Computer-Aided Design and Computer Graphics*, vol. 21, no. 11, 2009

[6] J.Kruger, J. Schneider, R. Westermann. “Compression and rendering of iso-surfaces and point sampled geometry”.*The Visual Computer*,22(8):517-530,2006

[7] J.Kruger, J. Schneider, R. Westermann “DUODECIM - a structure for point scan compression and rendering.” *IEEE VGTC Symposium Proceedings*, Leeds, U. K., 2005

[8] X. Tong, Z. S. Tang. “3D texture hardware assisted volume rendering with space leaping”.in Chinese, *Journal of Computers*, vol. 21, no.9,1998

[9] J. Krüger, R. Westermann. “Acceleration techniques for GPU-based volume rendering” *IEEE Conference on Visualization* . Seattle,Washington,2003

[10] S.H. Sun, Z.M. Lu. Technology and application of vector quantization [M]. *Beijing: Science Press*, 2002(in Chinese)

[11] A. Kaufman and K. Mueller, “Overview of Volume Rendering,” chapter for *The Visualization Handbook*, C. Johnson and C. Hansen, Eds., Burlington, MA: Academic Press, 2005

[12] Y. Linde, A. Buzo, R M. Gray. “An algorithm for vector quantizer design”.*IEEE Transactions on Communications*, vol. 28, no. 1,1980

[13] H.L. Liao, Z .Ji, Q.H. Wu. “A Novel Genetic Particle-Pair Optimizer for Vector Quantization in Image Coding.” *IEEE World Congress on Computational Intelligence*, Hong Kong,2008

[14] H.W.Sun, M. Gu, J.G. Sun, Improved codebook design algorithm based on principal component analysis [J]. *Journal of Computer-Aided Design &Computer Graphics*, 2005, 17(10): 2245-2250 (in Chinese)

[15] S Scholkopf, S Mika, C.J.C.Burges, et al.Input space versus feature space in kerner-based methods[J].*IEEE Transations on Neural Networks*,1999,10(5):1000-1017

[16] B. Moghaddam.Principal manifolds and probabilistic subspaces for visual recognition[J].*IEEE Transactions on Pattern Analysis and Machine Intelligence*,2002,24(6):780-788

Li-ping Zhao was born in 1984, HuNan, China. She obtained her master in Hunan University. Her main research interests include data compression, volume data visualization and multimedia.

Guangxue Yue was born in 1963, Guizhou, China. He obtained his master in Hunan University. Professor, the College of Mathematics & Information Engineering, jiaxing University, China. His main research interests include Biological Information, Distributed Computing & Network, and Hybird & Embedded Systems.

De-Gui Xiao was born in 1972,HuNan,China. He obtained his doctor's degree in Huazhong University of Science and Technology.Associate professor, the school of computer science and communication, HuNan University.His main research interests include visualization in scientific computing, intelligent visual monitoring surveillance, video processing.

Zhou Xu was born in 1983, Jiangshu, China. She obtained his master in Hunan University. Her main research interests include Biological information, DNA Computing and Parallel Computing.

Xiang Yu was born in 1983. He obtained master's degree in computer science and engineering from State University of New York at Buffalo, USA. His main research interests are design and evaluation of future networks and distributed systems.