Control of Lossy Compression of Three-dimensional Data

Pavel Diblík, Pavel Slavík
Department of Computer Science and Engineering
Czech Technical University
Karlovo nám. 13, 121 35 Prague
Czech Republic
e-mail:diblik@cs.felk.cvut.cz, slavik@cs.felk.cvut.cz

Abstract

This paper describes lossy compression method based on Discrete Cosine Transformation (DCT). The method is designed for compression of three-dimensional volume data and provides the ability to control errors in decompressed data. The three-dimensional extension of the DCT, denoted 3DCT, allows to reach the high compression ratio due to total removing of spatial redundancy. Thanks to the control ability is possible to guarantee the maximum level of error in decompressed data.

1. Introduction

Compression of two-dimensional information has been known for quite a long time. Both lossless and lossy compression methods are being used respectively in applications utilizing compression. The compression method can be extended into three-dimensional space. In this paper, we'll discuss the issues of effective compression of voxel data, putting stress on the medical data compression applications. We carried out a set of experiments with several lossy compression methods. The method that gave the best results was three-dimensional Discrete Cosine Transformation. The very heart of the system implemented are the modules for quantization process and arithmetic coding. High compression ratio has been achieved due to the full three-dimensional correlation. Concerning the main field of application, i.e. medical data compression, where certain level of error must be guaranteed, we have developed a special approach able to get decompressed data with strictly specified difference from original data. Besides, a methodology for measuring the loss of information during the compression process was designed and implemented.

2. Method selection

The main reason for getting into the research is a huge data volume requirement for storing three-dimensional data in a raw form on storage medium. e.g. volume with resolution $256 \times 256 \times 256$ voxels in 256 shadows of gray represents 16 MB data size.

We tried to develop such a method, which will be suitable for the compression of three-dimensional volume data generated by a medicine cathode–tube tomograph and magneto resonation tomograph.

There are two main groups of the compression methods. First one are lossless methods and the second one lossy methods. We started our research by using the compression of three-dimensional volume data by means of lossless compression in order to get some standard comparison with other methods. We developed a method that gives better compression ratio then common programs like Unix Compress or GNUzip programs but we needed much better ratio. Therefore we started experiments with lossy compression methods.

Most lossy image compression algorithms can be splitted into three parts: invertible transformation, lossy quantization and entropy encoding. We focused on the first part hoping to find transformation which is able to remove spatial redundancy of voxels in all three-dimensions. Commonly known compression standard for two-dimensional images - JPEG - is based on the same three-part type scheme as described above. The invertible transformation used in JPEG is two-dimensional discrete cosine transformation – 2DCT, described in [1] and [3]. We decided to use this transformation in solving our problem of the three-dimensional compression but in its the three-dimensional version, denoted 3DCT in this paper.

3. 3DCT – three-dimensional DCT

Input data can be imagined as a cube of voxels, which is taken as an extension of a pixel matrix used in 2DCT. For the transformation purpose we had to divide the cube into disjunct regular cubes of the same size, which all together fill in the input volume. The smallest cube size is $2 \times 2 \times 2$. Upper limit of cube size has not been stated. Larger size of a cube is more useful for spatial redundancy removing, but needs better accuracy of transformed coefficients than the smaller one.

The next step of the algorithm is applying 3DCT on each cube. The cube can be apprehend as two-dimensional array of one-dimensional vectors and three-dimensional cosine transformation can be executed as series of three one-dimensional cosine transformation. Each single transformation must be performed on different cube orientation. Maybe better explanation is through three-dimensional transposition T_3. Let $c_{x,y,z}$ quote the cube element. The T_3 can be understand as rotation of cube around main diagonal. Let $C^{T_3} = T_3(C)$, so $c^{T_3}_{x,y,z} = c_{y,z,x}$, then 3DCT can be written as $3DCT(C) = DCT(T^3(DCT(T^3(DCT(C)))))$. After the transformation we got cubes filled with transformed coefficients.

The next step is quantization. This step also includes changing three-dimensional configuration of coefficients to one-dimensional configuration suitable for the last step – entropy encoding. We used the extension of 2D zigzag traversal used in JPEG. The coefficients are grouped into planes by their position inside cube. One plane contains all coefficients with identical sum of $s = x + y + z$. The traversal is carried out in increasing order of s and 2D zigzagging is used within each plane. For graphic description refer to [6].

The quantization process represents the part of compression that produces loss of information. It is very important to do quantization very carefully. We can quantize coefficients by scaling to acceptable range and rounding to integer values. Another idea is to use logarithmical encoding of coefficients to achieve better accuracy of quantization.

The last step is an entropy encoding. There are lots of methods available and it was hard to select the best one. Huffman coding used in JPEG is a special case of more general arithmetic coding. We did not deal with this part as on previous one, therefore we are using standard algorithms (like Unix Compress) or our implementation of adaptive arithmetic coding, which is based on algorithm described in [5].

4. Loss Measuring

In applying the lossy compression method, decompressed data ware changed, if compared with the original data. We wanted to know, how big the change is and how big the level of loss was. There ware available two groups of measuring methods: objective and subjective ones. The subjective one is based on subjective judgement of the user. For our purpose we use an objective methods. Using the quantitative level of loss we expected good matching with the subjective opinion, i.e. visual opinion. It was important to formulate the level of loss as an analytical expression, because in this form it can be used as optimum criterion when setting parameters having influence on the compression ratio and the level of loss. We use such terms as distortion, error and information loss of reconstructed volume data interchangeably, because the loss of information produces mentioned distortion in reconstructed data (errors in comparison to the original data).

The difference between original and decompressed data, in our case three-dimensional volume data, can be described by the difference between two discrete three-parametric functions. The value of the function is determined by voxel's value on given position. The difference between two functions can be represented by the root mean square error ($RMSE$), but it does not give a chance to distinguish mall errors from few big errors. That is why we decided to use not one criterion only, but the group of three voxel images, so the commonly used peak signal to noise ratio is criteria. Our measurement was applied for the volume date having 8 bits per voxel resolution. Therefore the commonly used peak signal to noise ratio was computed as

$$PSNR = 20\log(\frac{255}{RMSE}),$$

where $RMSE$ is defined as

$$RMSE = \sqrt{\frac{1}{N_1 N_2 N_3}\sum_{r=1}^{N_1}\sum_{s=1}^{N_2}\sum_{t=1}^{N_3}[f(r,s,t) - \hat{f}(r,s,t)]^2}.$$

The others functions used for loss measuring are:
the lossy dispersion of absolute error

$$\sigma^2 = \frac{1}{N_1 N_2 N_3}\sum_{r=1}^{N_1}\sum_{s=1}^{N_2}\sum_{t=1}^{N_3}(|f(r,s,t) - \hat{f}(r,s,t)| - \alpha)^2,$$

where α (absolute error mean value) is defined as

$$\alpha = \frac{1}{N_1 N_2 N_3}\sum_{r=1}^{N_1}\sum_{s=1}^{N_2}\sum_{t=1}^{N_3}|f(r,s,t) - \hat{f}(r,s,t)|,$$

and the maximum of absolute error

$$\delta = \frac{100\%}{255} max(|f(r,s,t) - \hat{f}(r,s,t)|),$$

where N_1, N_2 and N_3 are the width, height and depth of the volume data in voxels, f represents the original data and $\hat{f}$ represents the reconstructed, i.e. decompressed data.

To avoid misleading the readers it is important to explain exactly how the compression ratios were computed. We defined it as

$$\frac{the\ number\ of\ bits\ in\ the\ original\ image}{the\ number\ of\ bits\ in\ the\ compressed\ image}.$$

As it is almost impossible to select only one objective criterion, we use all four criteria describing a compression performance: the compression ratio and three distortions describing functions mentioned above, where PSNR should be taken as the main function of the distortion measuring.

5. Compression control

The main advantage of compression methods based on the described approach is the possibility of controlling both the lossy level and compression ratio. There are two main types of controlling methods. First kind provides the indirect control by reducing the accuracy of transformed coefficients, i.e. by changing the transformed data. The influence to the decompressed data cannot be directly expressed. The control of the size of the cube relates to the type of control. Let us refer to the Tables 1 and 2 to understand the influence of a cube size on compression ratio and level of loss. More indirect control can be done by quantization. Beside using the different mapping (liner, logarithm), the omission of small coefficients can be used. The control is done by setting the maximal value of the coefficient which can be omit. We use term "coefficient threshold" instead "maximum value of omitted coefficients". A special case of two described indirect control methods is shown in Figure 2. The input parameters of compression are the cube order i.e. the length of the cube's edge and the coefficient threshold. The focused output parameters are PSNR and the compression ratio.

To be able to guarantee a certain level of data error we cannot use indirect methods effectively because the result of using indirect methods depends on the quality of input data. Otherwise the direct control methods give the independent results on the input data. The following direct control method offers flexibility to select maximal level of error of decompressed data. The control is realised through additional data chunks, which correct the decompressed data to the required form (Figure 1). The data distribution enables to use necessary amount of data only for the required maximal error. The methods can be combined together to get optimal results, i.e. the maximal compression ratio at given level of loss.

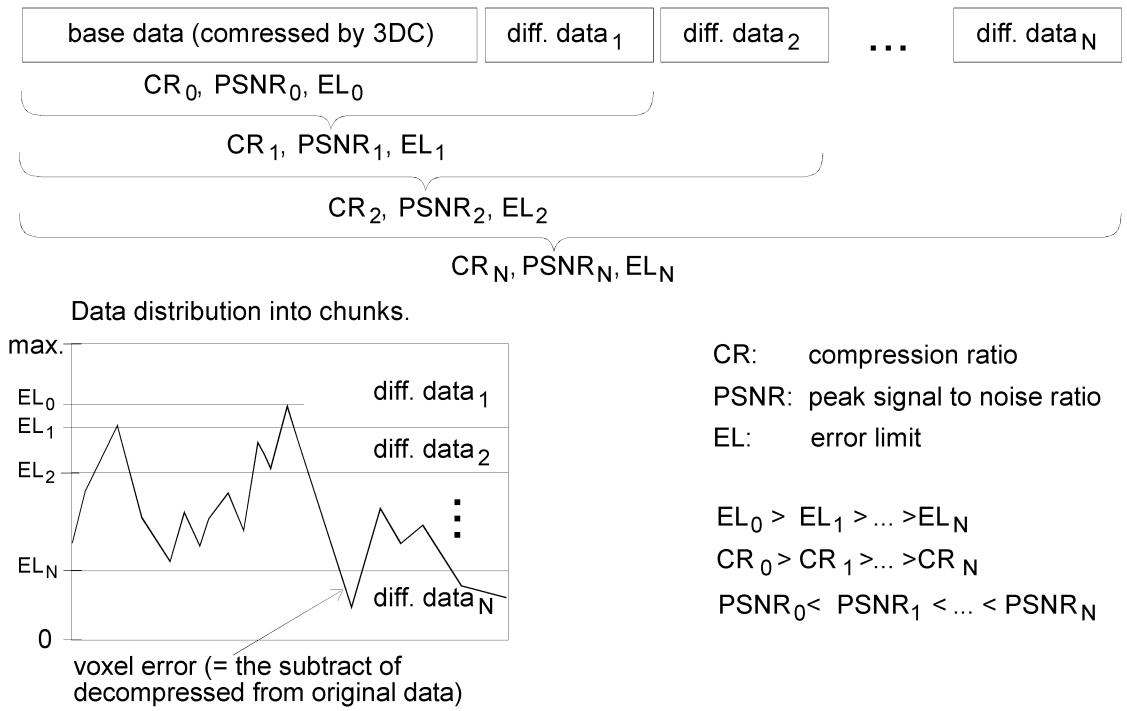

Figure 1: Data structure used by the direct control method.

6. Experimental Results

The first implementation is not an effective one. We have implemented 3DCT (direct and inverse ones) in a standard way, but after finding the best size of transformation matrix (the order of which should be power of two) we would like to use the fast 3DCT. The quantization process is now a fixed one. We can use quantization table with optimal coefficients for given input data. As mentioned above, the final step of compression is entropy encoding of quantized coefficients. Using arithmetic coding we achieved results described in the following tables. The numbers correspond to the compression of three-dimensional volume data of size $492 \times 240 \times 155$ voxels, 8 bits per voxel. To check suitability of arithmetic coding we compare it with common Unix compression program Compress.

	transformation matrix order							
	2	4	6	8	10	12	14	16
arith. coding	3.87	14.53	32.66	56.05	84.86	112.22	149.42	223.18
Compress	6.02	21.78	47.85	78.94	114.81	146.41	185.97	218.90

Table 1: Compression ratios in relation to transformation matrix order.

Program Compress represents compression method using a dictionary opposite to arithmetic coding, which belongs to group of statistical compression methods (described in [4]). The dictionary based method got better results, so we will replace arithmetic coding in future implementations. If we compare compression ratios achieved by lossless methods (Compress 4.12, GNUzip 3.98) and ratios in Table 1, the lossy method gives substantially better compression ratio. But good compression ratio has been achieved on the account of the loss of information. Next table shows influence of transform matrix order on the level of loss.

	transformation matrix order							
	2	4	6	8	10	12	14	16
$PSNR$ (dB)	49.30	44.20	41.65	39.89	38.58	37.41	36.51	34.63
σ^2	0.59	1.12	1.47	1.76	2.00	2.26	2.46	2.95
δ (%)	1.96	5.88	11.76	15.29	26.67	27.45	32.55	39.17

Table 2: Level of loss in depend on transform matrix order.

The approach of control level of loss by means of transformation matrix order is straightforward. All tests were made with fixed quantization rules. We use one byte per quantized coefficient. In the case of arithmetic coding there is a possibility to use non integer value of bits per quantized coefficient. This should be alternative to the control level of loss and compression ratio. The diagrams in Figure 2 represents relation between the cube size (specified by cube order), coefficient threshold, PSNR and compression ratio. The meaning of diagrams is clear: greater cube size and/or higher coefficient threshold decrease PSNR but increase the compression ratio.

The behaviour of compression in relation to the error limit and coefficient threshold as input parameters of compression shows the Figure 3. The diagrams in this Figure are measured for fixed cube order equals to 4. The left diagram in Figure 3 shows how to set the coefficient threshold and error limit in order to get the highest PSNR or compression ratio. More complex diagram in Figure 4 answers to question:" How to set all input compression parameters, i.e. cube order, coefficient threshold and error limit, to get the lowest loss of information by given compressed ratio?" For example you need the compression ratio at least 100, then you should use cube order 8, error limit 16 and the coefficient threshold 1. You can use the greater cube order or higher coefficient threshold, then you get better compression ratio, but always you get worse PSNR.

The efficiency of 3DCT can be shown on comparison with conventional two-dimensional discrete cosine transform (2DCT). The compression ratios and lossy level of these methods for the three-dimensional volume data are also presented in table 3. 2DCT was used by JPEG algorithm on sliced input data. As expected, the compression ratio achieved by 3DCT is better then 2DCT on the same level of loss because 3DCT uses all three dimensions to remove spatial redundancy.

		method of compression	
		2DCT	3DCT
distortion	$PSNR$ (dB)	45.34	46.66
	σ^2	1.78	1.27
	δ (%)	10.98	12.94
Compression ratio		4.97	22.32

Table 3: Comparison of 2DCT (JPEG) and 3DCT

The JPEG compression was made with standard parameters (quality=32, Huffman coding) and 3DCT method used cube order 4. The table 3 presents much better compression ratio on 3DCT method side by same distortion on both sides.

The Figures 5 - 8 show influence of information loss on medical three-dimensional volume data obtained from cathode-tube tomograph. The volume data represent density of plexure in certain space position. For creation first eight Figures, three-dimensional volume data of human hand plexure density distribution were used.

The displayed surfaces on Figures were generated by segmentation based on threshold. Voxel value is proportional to the density of plexure. In order to display bone plexure, threshold over 100 has been used.

7. Conclusion and Future Research

The average compression ratio achieved by lossless compression on tested volume data was approximately 4 (by standard UNIX prog. Compress). The ratio with lossy compression can be 20 or more without visible distortion in image. Weak points of 3DCT method are time and memory consumption now, but it can be decreased by better implementation and optimization. Moreover, we see a way in parallelising of transformation calculation. The main advantage of 3DCT is the capability to control compression ratio and relevant level of loss. We would like to try to move control of compression behaviour from transformation part to quantization part in order to optimize transformation part from the point of speed. The question is which algorithm best fits for coding quantized coefficients. There is an idea of making combination of LZSS as dictionary method and final coding by arithmetic coding. The direct control method gives the choice to specify of data error exactly. The further work will be concentrated on investigation of dependencies between compression parameters and data output. Also the influence of the quality and type of input data on the quality of output data will be tested.

References

[1] K. R. Rao, P. Yip: *Discrete Cosine Transform; algoritmus, advantages and applications*, Academic Press, New York 1990

[2] M. Rollins, F. Carden: *Possible Harmonic-wavelet Hybrids in Image Compression*,Proceedings Data Compression Conference 92, IEEE Computer Society Press, pp.191-199.

[3] L. Jaroslavskij, I. Bajla: *Metody a systemy cislicoveho zpracovania obrazov*, Alfa, Bratislava 1989, ISBN 80–05–00046–4

[4] B. Melichar: *Textove informacni systemy*, CTU, Praha 1992

[5] I. H. Witten, R. M. Neal, J. G. Cleary: *Arithmetic coding for data compression, Communication of the ACM 6/1987*, pp.520–540

[6] B. Yeo, B. Liu: *Volume Rendering of DCT-Based Compressed 3D Scalar Data*, IEEE Transactions on Visualization and Computer Graphics, vol.1,March 1995, pp.29-43.

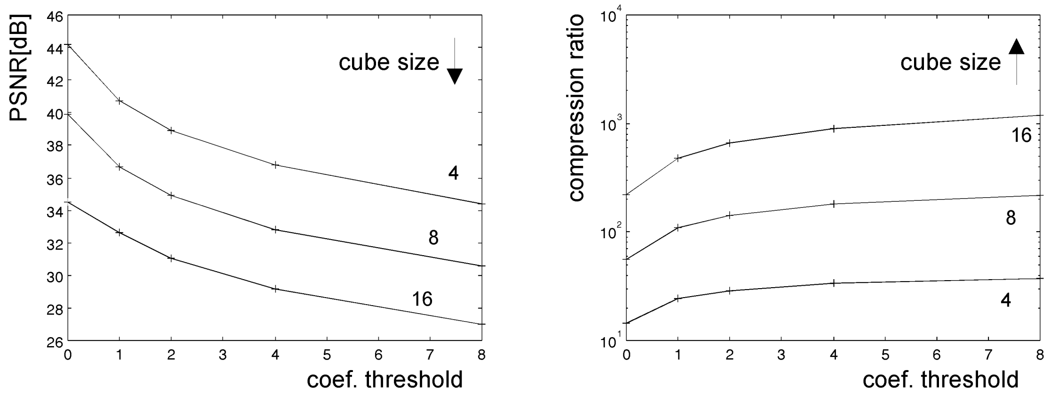

Figure 2: PSNR and compression ratio in dependency of cube order and coefficient threshold.

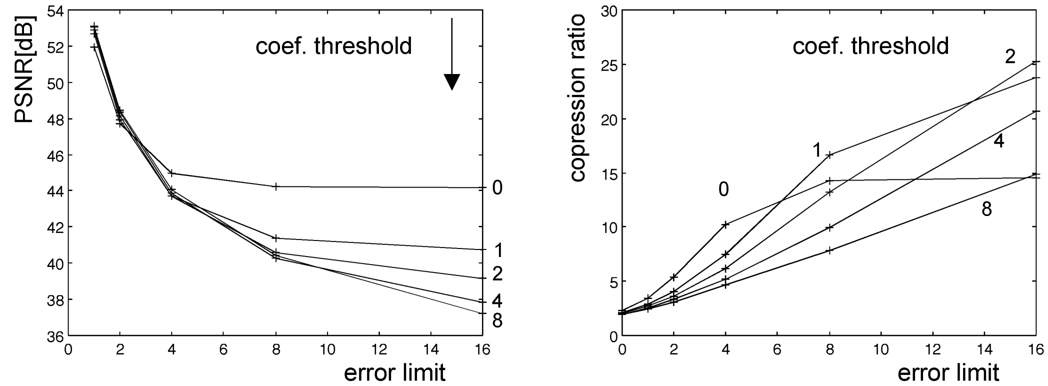

Figure 3: PSNR and compression ratio in dependency of coefficient threshold and error limit.

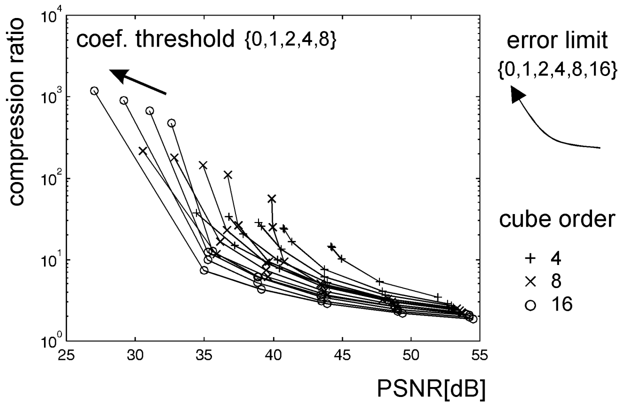

Figure 4: The relation between PSNR and compression ratio in dependency of cube order, coefficient threshold and error limit.

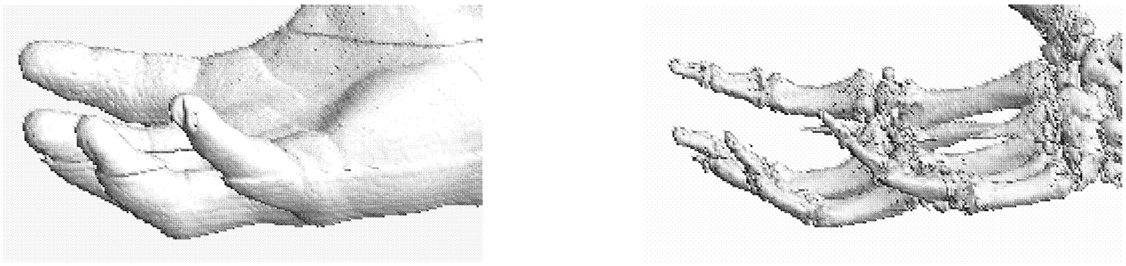

Figure 5: Human hand and bone plexure. Original data - no information loss, compression ratio=4

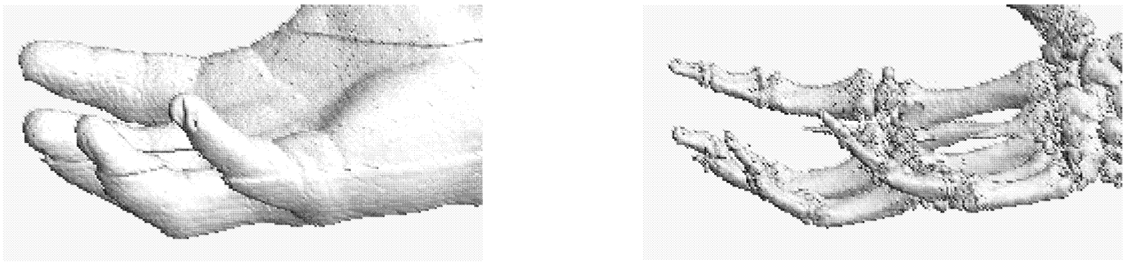

Figure 6: Human hand and bone plexure, cube order=4, $PSNR = 44.20$ dB, $\sigma^2 = 1.12$, $\delta = 5.88\%$, compression ratio=22

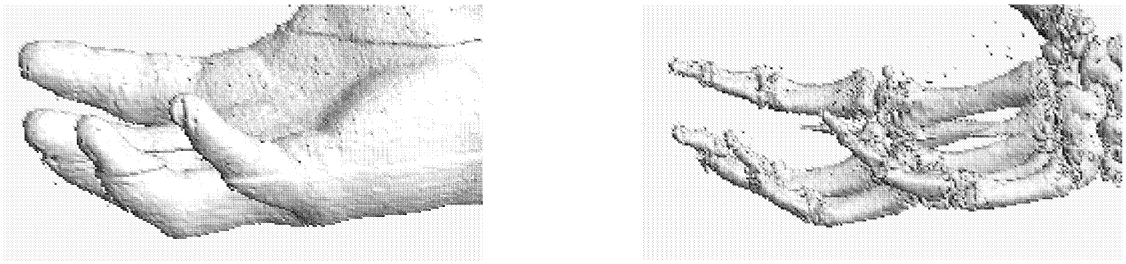

Figure 7: Human hand and bone plexure, cube order=8, $PSNR = 39.89$ dB, σ^2=1.76, δ=29.41%, compression ratio=78

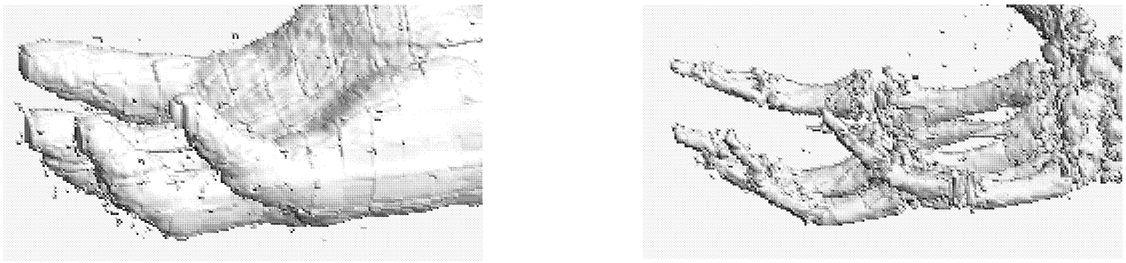

Figure 8: Human hand and bone plexure, cube order=16, $PSNR = 34.65$ dB, σ^2=15.60, δ=39.17%, compression ratio=200