

Research Article

Parallel Fractal Compression Method for Big Video Data

Shuai Liu ^{1,2,3} Weiling Bai ^{1,2} Gaocheng Liu,^{1,2} Wenhui Li ³
and Hari M. Srivastava ^{4,5}

¹College of Computer Science, Inner Mongolia University, Hohhot, China 010012

²Inner Mongolia Key Laboratory of Social Computing and Data Processing, Inner Mongolia University, Hohhot, China 010012

³Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China

⁴Department of Mathematics and Statistics, University of Victoria, Victoria, British Columbia, Canada V8W 3R4

⁵Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 40402, Taiwan

Correspondence should be addressed to Shuai Liu; cs.liu.shuai@gmail.com

Received 5 June 2018; Revised 9 August 2018; Accepted 16 August 2018; Published 3 October 2018

Academic Editor: Paolo Bellavista

Copyright © 2018 Shuai Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the development of technologies such as multimedia technology and information technology, a great deal of video data is generated every day. However, storing and transmitting big video data requires a large quantity of storage space and network bandwidth because of its large scale. Therefore, the compression method of big video data has become a challenging research topic at present. Performance of existing content-based video sequence compression method is difficult to be effectively improved. Therefore, in this paper, we present a fractal-based parallel compression method without content for big video data. First of all, in order to reduce computational complexity, a video sequence is divided into several fragments according to the spatial and temporal similarity. Secondly, domain and range blocks are classified based on the color similarity feature to reduce computational complexity in each video fragment. Meanwhile, a fractal compression method is deployed in a SIMD parallel environment to reduce compression time and improve the compression ratio. Finally, experimental results show that the proposed method not only improves the quality of the recovered image but also improves the compression speed by compared with existing compression algorithms.

1. Introduction

With the rapid development of the Internet and intellectual mobile terminals, multimedia video and image applications are becoming more and more widespread. Video data is ubiquitous and plays a critical role in all aspects of people's lives, including urban security, medical care, education, communications, industrial production, and film and television. Since video applications generate massive amounts of video data every moment, the amount of global video data has exploded quickly [1]. Big video data do not only broaden the horizon of human beings, and enable us to better experience and recognize the world around us, but also have buried a large amount of valuable information waiting for exploration [2].

In order to effectively store and manage big video data, efficient video compression technology has become crucial. The purpose of video compression is to ensure that the video compression ratio is maximized while maintaining a certain image quality [3]. Video compression technology has been widely used today, such as digital cameras, USB cameras, video phones, video on demand, video conferencing systems, and digital surveillance systems. Meanwhile, many video coding methods including fractal coding have been improved, and some coding techniques have been applied to video coding standards [4].

Existing content-based video compression methods have reached bottlenecks. However, rule-based fractal image compression technology is a potential image compression method. Its potentially high compression ratio characteristic

has made it subject to attention of many scholars [5]. Fractal image compression technology converts a digital image into a set of contractive affine transformations (CAT) according to self-similarity of the image, and parameters of CAT of the image are stored as a compression file [6]. Moreover, a corresponding decompression process is very simple to suit the situations where an image is compressed once and decompressed many times, such as video on demand.

However, the traditional fractal video compression method performs frame-by-frame compression on the entire video without considering spatial-temporal similarity, which causes a lot of computational redundancy [7]. Moreover, since each range block is matched with all domain blocks, such a large amount of calculation leads to a high computational complexity. Therefore, the existing fractal compression method is underutilized.

In this paper, we propose a novel fractal video compression method. First, an entire video is divided into several fragments according to the spatial-temporal similarity of video frames. Each fragment contains several similar video frames. Secondly, we classify range and domain blocks according to their color similarity feature in each subfragment. Meanwhile, in order to compress real-time big video data, we propose a video compression framework with a dual-layer parallel structure. In the first layer of the parallel structure, the central server allocates all video fragments to multiple processors averagely for parallel compression. In the second layer, the processor allocates all kinds of range and domain blocks to multiple computing nodes for simultaneous matching search.

The rest of the paper is organized as follows. Section 2 reviews related work on video compression methods and fractal image coding. Section 3 presents our proposed parallel fractal compression method for big video data and analyzes its computational complexity. In Section 4, in order to verify the effectiveness of the proposed method, the traditional fractal compression method and some recent image compression methods are used as comparisons. Experimental results show the effectiveness of our method. Finally, Section 5 concludes our work and describes the direction of future research.

2. Related Works

2.1. Video Sequence Compression Method. Since the 1980s, video coding technology has developed rapidly and has been a hot research field soon. In March 2003, two international organizations for standardizations ISO/IEC and ITU-T jointly developed the video coding standard H.264/AVC [8]. H.264/AVC achieved good results in many aspects such as coding efficiency, image quality, network adaptability, and error resistance. However, its coding algorithm had a high degree of complexity [9]. After years, many new technologies such as motion compensation, transformation, interpolation, and entropy coding had demonstrated their superiority. Therefore, ISO/IEC and ITU-T jointly developed a new video coding standard, H.265/HEVC [4], in November 2013.

The video coding standard defined a syntax and semantic constraint decoder of code stream with open-ended encoders. Therefore, in order to further improve compression efficiency, coding technology is perfected on the premise of conforming to constraint of code stream [10]. The existing research on coding optimization mainly focused on two aspects. One was how the coding efficiency can be further improved; the other was how the coding complexity can be effectively reduced.

In order to improve coding performance, HEVC defined 35 intraprediction directions and advanced interframe interpolation techniques with spatial correlation of images [11]. Zhang et al. proposed a method for the recombination and prediction of the reference frame with background modeling [12]. It effectively allocated resources and improved coding performance. Ugur et al. proposed an adaptive filtering technique. The design of interpolation and deblocking filter improved coding efficiency [13]. Seo et al. proposed a rate control method to maximize coding efficiency [14].

The above methods mainly eliminated redundancy by adding a large number of coding modes, optimization parameters, and traversal searching techniques. Since the compression ratio and time can be improved by mining the visual redundancy in video, Zhang et al. proposed a block-adaptive residual preprocessing method based on the stereoscopic visual JND (just notice difference) model [15]. It effectively reduced unnecessary perceived redundancy without degrading visual quality. Luo et al. proposed an alternative perceptual video coding method to improve the existing H.264/advanced video control (AVC) framework, which achieved significant bit savings while maintaining visual quality [16]. By combining video coding with visual perception, Wang et al. proposed a game-based efficient coding method and a bit allocation method, which effectively improved network adaptability and coding efficiency [17].

The compression algorithm of HEVC had a high complexity because many technologies were introduced in it, such as hierarchical variable-size coding units, multiscale prediction units, transform units, and multireference frame motion estimation. Thus, Guo et al. proposed a method to reduce the computational complexity of video compression standards [18]. It included an intraframe and a chroma search algorithm, which accelerated the prediction process of luminance and chromaticity macroblocks. Potluri et al. introduced a new 8-point DCT approximation that required 14 additions without multiplication [19]. Pan et al. proposed an efficient motion and disparity estimation algorithm to reduce the computational complexity of multiview video coding [20].

Today, content-based video compression technologies made it difficult to get major breakthroughs. Most of the existing video compression methods improve the compression quality by increasing the computational complexity of encoding. The computational complexity of existing video compression methods is generally high due to a large number of parameters and mathematics that need to be calculated. Therefore, this research direction is to build an efficient coding framework. The method we proposed better balances

the relationship between computational complexity and image quality.

2.2. Fractal Image Compression. Fractal image compression was a compression method by CAT [21]. It was first proposed by Barnsley and Hurd in the mid-1980s according to the mathematical theory of fractal geometry. Then, Jacquin divided a coded image into small pieces of equal size and explored the mapping relationship between these small pieces [22]. Fractal image coding has advantages such as high compression ratio and independence from resolution. However, its compression time is very long because each range block needs to be searched with all domain blocks. The key of acceleration in fractal coding is a well-designed search scheme. At present, improvement of a fractal coding algorithm is mainly divided into two directions, which are sub-block classification and neighborhood search.

The sub-block classification method classifies image sub-blocks according to a certain characteristic. Thus, intragroup search is used to replace global search during matching. Jacquin divided image blocks into shade blocks, edge blocks, and midrange blocks according to visual geometry [23]. Jacobs et al. proposed a more refined classification method to obtain 72 types of sub-blocks by classifying image sub-blocks based on gray mean and variance [24]. Jiang et al. applied the K -means clustering algorithm into fractal image compression to cluster range and domain blocks [25]. Jaferzadeh et al. used pixel space and 1D-DCT vectors to implement fuzzy clustering, which improved the speed of compression with equal decoding quality [26]. Wu et al. divided domain blocks into simple blocks and complex blocks [27]. In order to shorten the compression time, only complex blocks are coded. However, a simple block is stored by its pixel mean and coordinates of the upper left corner.

Assuming that positions of optimal matching domain blocks are often concentrated near the range block, the neighborhood search method refers to searching its optimal matching block only in the neighborhood of a range block, which narrows the search range from global to local search [28]. Chong et al. proposed an improved formula for prequantized nearest neighborhood search, which was based on orthogonal projection and fractal transform parameters [29]. In addition, they derived an optimal adaptive scheme for approximating search parameters to improve the performance of the algorithm. Truong et al. proposed a new search strategy based on spatial correlation of images [30]. Lin et al. implemented a neighborhood search by utilizing a phenomenon that similarly edge-shaped blocks are concentrated in certain regions [31]. Wang et al. calculated and sorted the standard deviation of domain blocks. Each range block was limited to search for domain blocks with similar standard deviation [32].

Searching for the optimal matching block in the neighborhood of a range block speeds up the compression process, but ultimately results in a larger pixel difference between the decoded image and the original image. This strategy trades for shorter compression times at the expense of image quality. The proposed method uses the idea of classification to narrow the matching range of range blocks while ensuring

image quality. Although there are many classification methods, the method we proposed has a lower calculation amount and its speed is satisfactory.

3. The Proposed Fractal Compression Methods

3.1. Traditional Fractal Compression Method for Video Sequence. The traditional fractal compression method for video sequence performs frame-by-frame compression on the entire video. In the traditional method, an image f is divided into n_R nonoverlapping blocks R_1, R_2, \dots, R_{n_R} (range block) with the same size.

Assuming that the size of f is $M \times M$ and the size of R_i is $N \times N$, a $2N \times 2N$ interception window is used to traverse along horizontal and vertical directions of f by a given step δ . Each movement of the interception block constitutes a domain block. All n_D blocks constitute a search space $S_D = \{D_1, D_2, \dots, D_{n_D}\}$, where D_j is the j -th domain block. It is obvious that $n_R = (M/N)^2$ and $n_D = (M - 2N/\delta + 1)^2$.

For each domain block D_j , the mean of four neighboring pixels generates an $N \times N$ pixel block D'_j .

Then, eight isometric transformations are performed on $S'_D = \{D'_1, D'_2, \dots, D'_{n_D}\}$ to generate codebook Ω of matching operation.

For an arbitrary range block R_i , the optimal matching block D'_j is determined by 1, where $t_k \in \{t_1, \dots, t_8\}$ are eight isometric transformations. s_i and o_i are parameters of affine transformation.

$$d(R_i, W_i(D'_j)) = \min \left\| R_i - (s_i \cdot (t_k(D'_j)) + o_i) \right\|^2. \quad (1)$$

Equations 2 and 3 are used to calculate s_i and o_i , where \bar{r}_i and \bar{d}_j represent the mean intensity of block R_i and D'_j , $\langle \rangle$ is the inner product in the Euclidean space, and $\| \cdot \|$ stands for 2-Norm. I is an identity matrix with the same size as R_i and D'_j .

$$s_i = \frac{\langle R_i - \bar{r}_i \cdot I, D'_j - \bar{d}_j \cdot I \rangle}{\| D'_j - \bar{d}_j \cdot I \|^2}, \quad (2)$$

$$o_i = \bar{r}_i - s_i \cdot \bar{d}_j. \quad (3)$$

For each range block R_i , the transform cluster $W_i(x_j, y_j, t_i, s_i, o_i)$ obtained by 1 is its fractal code, where (x_j, y_j) represents the location of D'_j .

The computational complexity of the traditional fractal compression algorithm is given by Lemma 1.

Lemma 1. *The computational complexity of traditional fractal image compression algorithm is $O(M^4/N^2)$.*

Proof. In the traditional fractal image compression method, searching for the optimal matching D_j for each R_i requires

a global search of codebook Ω . The total number of range blocks n_R and domain blocks n_D is shown in 4.

$$\begin{aligned} n_R &= \left(\frac{M}{N}\right)^2, \\ n_D &= \left(\frac{M-2N}{\delta} + 1\right)^2. \end{aligned} \quad (4)$$

Each range block R_i must be searched for the optimal self-similar D_j by comparing n_D times with all domain blocks. The number of comparisons T_{com} to complete compression is given by 5.

$$T_{\text{com}} = \left(\frac{M}{N}\right)^2 \times 8 \times \left(\frac{M-2N}{\delta} + 1\right)^2 = O\left(\frac{M^4}{N^2}\right). \quad (5)$$

Based on the discussion above, Lemma 1 is proved. Proof is finished.

3.2. The Proposed Fractal Video Sequence Compression Method. There are two drawbacks of traditional fractal video compression. First, spatial-temporal similarity is not considered in the frame-by-frame compression method. Second, using the global search method to find the optimal matching block results in a long compression time. Therefore, we propose a novel video sequence compression method. Our method is divided into two steps. First, the video sequence is divided into video fragments according to spatial-temporal similarity. Second, domain and range blocks are classified based on color similarity feature within each fragment. A compression flow chart is shown in Figure 1. We explain steps to fragment the video in Section 3.2.1 and describe the steps to classify domain and range blocks in Section 3.2.2.

3.2.1. Video Content Classification Based on Spatial-Temporal Similarity. Video data is an unstructured data that has both temporal and spatial properties. Before it is compressed, video data may be processed and structured according to spatial-temporal similarity. Therefore, we fragment the video sequence based on image content. Figure 2 shows three video fragments that belong to the same video sequence "Jogging" [33]. Each row represents a fragment. There are large differences in video frames between different fragments. In a fragment, the difference between sequential frames is small.

According to the similarity of video content, we use the color histogram method to fragment a video. The HSV color model is selected in this paper, as shown in Figure 3. The HSV color model is composed of three components, which are H (hue), S (saturation), and V (value). The human eye has different sensitivities to the three components. Therefore, the weights of the three components are modified to save storage space and reduce computational complexity.

H, S, and V are divided into a , b , and c intervals, respectively. According to this quantization level, each color

component is synthesized as a G -dimensional feature vector by 6.

$$G = (a-1) \cdot b \cdot c + (b-1) \cdot c + (c-1) + 1. \quad (6)$$

Similarity of histograms will be accurately calculated by 7, where $P = (p_1, p_2, \dots, p_G)$ and $Q = (q_1, q_2, \dots, q_G)$ represent color feature vectors of frames P and Q .

$$d(P, Q) = \sqrt{1 - \frac{\sum_{i=1}^G \sqrt{p_i \cdot q_i}}{\sqrt{\sum_{i=1}^G p_i} \cdot \sqrt{\sum_{i=1}^G q_i}}}. \quad (7)$$

A large change occurs between the two frames P and Q if $d(P, Q)$ is larger than a certain threshold T . Therefore, P and Q are divided into different video fragments by

$$\begin{aligned} P, Q \in V_i, \quad d(P, Q) \leq T, \\ P \in V_i, Q \in V_j, \quad \text{otherwise.} \end{aligned} \quad (8)$$

Finally, the entire video V is divided into n video fragments $V = \bigcup_{i=1}^n V_i$, where each video fragment V_i contains u_i frame images $f_{i,j}$: $V_i = \bigcup_{j=1}^{u_i} f_{i,j}$.

3.2.2. Video Fragment Compression Method Based on Color Similarity. In order to accelerate the fractal compression speed and improve the decoding quality of image, we combine sequential video frames into a whole image matrix to classify domain blocks in each video fragment V_i .

Figures 4(a) and 4(b) are two frames in the same video sequence. Block 1 is R_i of Figure 4(a). In the traditional frame-by-frame video compression algorithm, block 2 is the optimal-matching domain block D_j of R_i in Figure 4(a). However, block 3 in Figure 4(b) is D_j of R_i by the proposed method. The mean squared error (MSE) is calculated by 9, where n^2 is the number of pixels of the image. $X_{i,j}$ and $Y_{i,j}$, respectively, represent the gray value of images X and Y at position (i, j) . Block 3's MSE is 193, and block 2's MSE is 351. Therefore, block 3 is better than block 2.

$$\text{MSE} = \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n (X_{i,j} - Y_{i,j})^2. \quad (9)$$

Matching error $E(R, D)^2$ of two image blocks R and D is calculated by 10, where the definition of \bar{r} , \bar{d} , and I is the same as that in 2 and 3.

$$E(R, D)^2 = \|R - \bar{r} \cdot I\|^2 - \frac{|\langle R - \bar{r} \cdot I, D - \bar{d} \cdot I \rangle|^2}{\|D - \bar{d} \cdot I\|^2}. \quad (10)$$

Theorem 1. For two matrices D and R with the same size, $E(R, D)^2 \geq \|D - \bar{d} \cdot I\|^2 - b_i^2$ ($a_i \cdot \|D - \bar{d} \cdot I\| + b_i \cdot \|R - \bar{r} \cdot I\|$), where a_i and b_i are any unit in matrix $a = R - \bar{r} \cdot I$ and $b = D - \bar{d} \cdot I$.

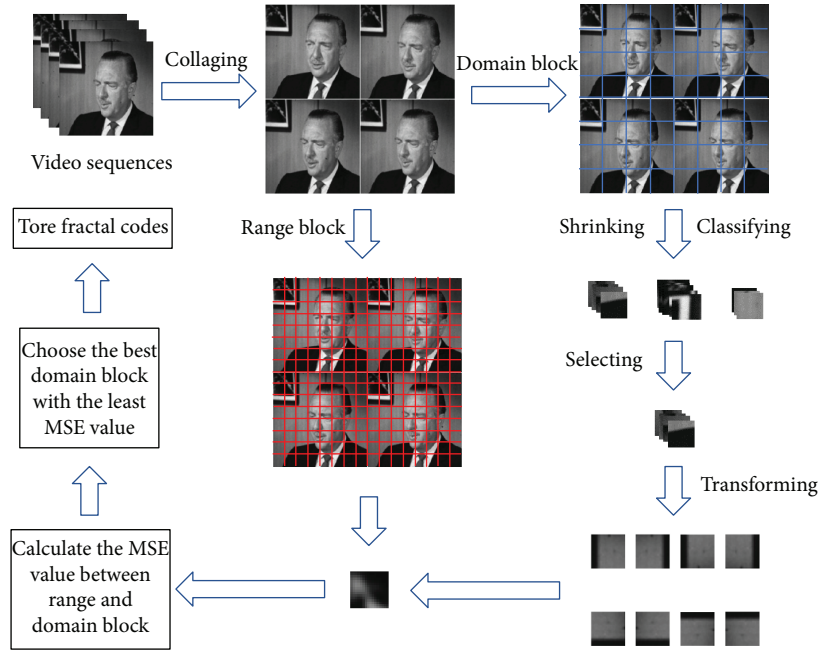


FIGURE 1: The proposed fractal compression process.



FIGURE 2: Video fragments of “Jogging.”

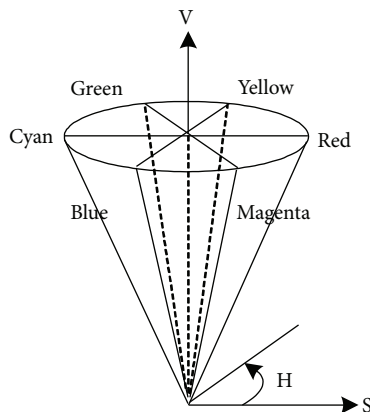


FIGURE 3: HSV color model.

Proof. Let $R - \bar{r} \cdot I = a, D - \bar{d} \cdot I = b$, 10 is described as

$$E(R, D)^2 = \frac{\|a\|^2 \cdot \|b\|^2 - |\langle a, b \rangle|^2}{\|b\|^2}. \quad (11)$$

Build 12 and 13, where n is the total number of pixels of R and D .

$$X' = \left(-\frac{b_i}{\|b\|^2} \cdot b_1, -\frac{b_i}{\|b\|^2} \cdot b_2, -\frac{b_i}{\|b\|^2} \cdot b_3, \dots, 1 - \frac{b_i}{\|b\|^2} \cdot b_n \right), \quad (12)$$

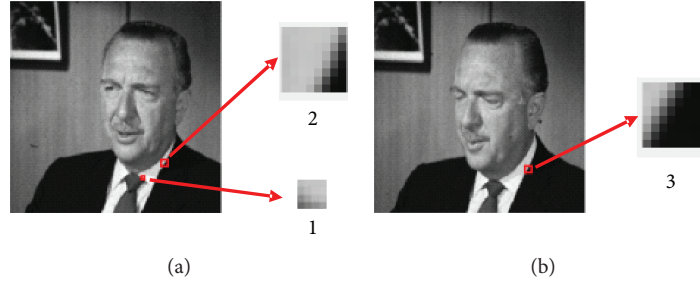


FIGURE 4: Combinatorial compression.

$$\lambda = \frac{\langle b, b \rangle}{a_i \cdot \langle b, b \rangle - b_i \cdot \langle a, b \rangle}. \quad (13)$$

Thus, $X = \lambda X'$ satisfies $\langle b, X \rangle = 0$ and $\langle a, X \rangle = 1$; we have

$$E(R, D)^2 = \frac{\|a\|^2 \cdot \|b\|^2 - |\langle a, b \rangle|^2}{\|b\|^2} \geq \|X\|^2. \quad (14)$$

Therefore, by 12 and 13, 11 is described as follows:

$$\begin{aligned} E(R, D)^2 &\geq \langle \lambda X', \lambda X' \rangle = \frac{1}{(a_i \cdot \langle b, b \rangle - b_i \cdot \langle a, b \rangle)^2} \\ &\quad \cdot \left(\sum_{j=1}^{n-1} b_n^2 b_i^2 + (\langle b, b \rangle - b_i b_n)^2 \right) \\ &= \frac{\|b\|^2 - b_i^2}{(a_i \cdot \|b\|/\|a\| - b_i \cdot \langle a, b \rangle / \|b\| \cdot \|a\|)^2 \cdot \|a\|^2} \\ &\geq \frac{\|b\|^2 - b_i^2}{((a_i \cdot \|b\|/\|a\|) + b_i)^2 \cdot \|a\|^2} \\ &= \frac{\|D - \bar{d} \cdot I\|^2 - b_i^2}{(a_i \cdot \|D - \bar{d} \cdot I\| + b_i \cdot \|R - \bar{r} \cdot I\|)^2}. \end{aligned} \quad (15)$$

Based on the discussion above, Theorem 1 is proved. Proof is finished.

Let $GA = \|D - \bar{d} \cdot I\|^2 - b_i^2 / (a_i \cdot \|D - \bar{d} \cdot I\| + b_i \cdot \|R - \bar{r} \cdot I\|)^2$; $E(R, D)^2$ between D_{j1}' and D_{j2}' with R_i is approximated by calculating GA. Meanwhile, a_i and b_i are replaced by mean gray values of gray matrices a and b . Assuming that $GA(R_i, D_{j1}')$, $GA(R_i, D_{j2}')$ is smaller than a certain value σ . Then, $GA(D_{j1}', D_{j2}')$ is not much different. Therefore, we classify domain blocks by an automatic classification method according to Theorem 1.

The mean gray value $D_A = (h_{A1}, h_{A2}, \dots, h_{AN \times N})$ of all remaining domain blocks is calculated by 16 as the initial cluster center of the i -th category of domain blocks, where n is the number of domain blocks and $h_{n(k,l)}$ represents the gray value of the n -th domain block at position (k, l) .

$$h_{A(k,l)} = \frac{1}{n} (h_{1(k,l)} + h_{2(k,l)} + \dots + h_{n(k,l)}). \quad (16)$$

The distance between range block R_i and center $\{c_1, c_2, \dots, c_m\}$ is calculated by 17, where $h_{R_i(k,l)}$ and $h_{c_j(k,l)}$ represent the gray value of R_i and the j -th center c_j at position (k, l) , respectively.

$$\text{dis}(R_i, c_j) = \frac{\left[\sum_{k=1}^N \sum_{l=1}^N (h_{R_i(k,l)} - h_{c_j(k,l)})^2 \right]}{N \times N}. \quad (17)$$

A matching search is performed between each category of range blocks and the same category of domain blocks, such as C_{R_i} is matched with C_{D_i} . A matching process between different categories of range blocks is performed independently.

The computational complexity of the proposed algorithm is given by Theorem 2.

Theorem 2. *The computational complexity of the proposed algorithm is $O(M^4/N^2m^2)$.*

Proof. Assuming that all domain blocks are uniformly classified into m categories. Correspondingly, range blocks are classified into m categories. The number of each category of domain blocks C_D and range blocks C_R is given by

$$\begin{aligned} C_D &= \frac{((M - 2N/\delta) + 1)^2}{m}, \\ C_R &= \frac{(M/N)^2}{m}. \end{aligned} \quad (18)$$

Each category of range blocks only needs to be matched with its corresponding domain category. The times of comparisons T_{com} within each category is shown by 19.

$$T_{\text{com}} = \left(\frac{M}{N}\right)^2 \times 8 \times \left(\frac{M - 2N}{\delta} + 1\right)^2 \times \left(\frac{1}{m^2}\right) = O\left(\frac{M^4}{N^2m^2}\right). \quad (19)$$

Based on the discussion above, Theorem 2 is proved. Proof is finished.

Classification Algorithm of Domain and Range Blocks (one frame e.g.,).

Input: n_D domain blocks and n_R range blocks.

Output: domain block categories $C_{D1}, C_{D2} \dots C_{Dm}$ and range block categories $C_{R1}, C_{R2} \dots C_{Rm}$.

Set: the number of categories m and a set of threshold sequences $\{\sigma_1, \sigma_2, \dots, \sigma_t\}$

Repeat

Mean gray value $D_A = (h_{A1}, h_{A2}, \dots, h_{AN \times N})$ of n remaining domain blocks is calculated by Eq.16;

σ_p is the median of threshold sequence $\{\sigma_1, \sigma_2, \dots, \sigma_t\}$;

$\omega = 0$;

Repeat

Remove a domain block D'_j from $\{D'_1, D'_2, \dots, D'_n\}$;

GA between D_A and D'_j is calculated;

If $GA \leq \sigma_p$ then

$\omega = \omega + 1$;

$D'_j \in C_{Di}$;

End

Until Each domain block of $\{D'_1, D'_2, \dots, D'_n\}$ is compared;

If $\omega \gg n_D/m$ then

$\sigma_p = \sigma_{p+1}$;

$\omega = 0$;

Perform previous "Repeat" step, until $|\omega - n_D/m| \leq 10^{\log_{10}(n_D)-1}$;

End

If $\omega \ll n_D/m$ then

$\sigma_p = \sigma_{p-1}$;

$\omega = 0$;

Perform previous "Repeat" step, until $|\omega - n_D/m| \leq 10^{\log_{10}(n_D)-1}$;

End

The i -th category of domain blocks C_{Di} is determined;

Until n_D domain blocks are classified and $C_{D1}, C_{D2} \dots C_{Dm}$ are got;

Center $c_1, c_2 \dots c_m$ of s categories of domain blocks are obtained by Eq.16;

Repeat

Remove a range block R_i from $\{R_1, R_2 \dots R_{n_R}\}$;

Distance $dis(R_i, c)$ between R_i and center $\{c_1, c_2 \dots c_m\}$ are calculated by Eq.17;

Select the smallest distance $dis(R_i, c_j)$;

$R_i \in C_{Ri}$;

Until n_R range blocks are classified and $C_{R1}, C_{R2} \dots C_{Rm}$ are got;

ALGORITHM 1

Since the matching range for each range block is narrowed down to one of m categories of domain blocks, instead of traditional global search, theoretically the compression speed of the proposed algorithm is m times as fast as the traditional algorithm. In order to improve the compression speed obviously, normally $m \ll ((M - 2N/\delta) + 1)^2$.

Inference 1. The compression speed of the proposed combined algorithm is m'/f times as fast as the traditional algorithm.

Proof. Assuming that the time required to compress a single frame by the traditional algorithm is t . All domain blocks are divided into m' categories when f frames are combined to compress integrally. Correspondingly, all range blocks are divided into m' categories.

The mean time required to compress the f frame is $t \cdot f / m'$. Therefore, the compression time of the proposed

combined algorithm is $t \cdot f / m' \cdot f$. However, the time required to compress the f frame by the traditional algorithm is $t \cdot f$.

Based on the discussion above, Inference 1 is proved. Proof is finished.

Inference 1 indicates that the compression speed of the proposed combined algorithm is determined by two factors. One is the number of combined images and the other is the class number of domain blocks. How to balance these two factors is related to compression time and image quality.

3.3. Parallel Framework for Big Video Data Compression. Parallel computing lay the foundation of methodology for the solution of complex problems. Considering the compression of each video fragment without affecting each other and that the matching search between range blocks of different categories is independent of each other, the proposed algorithm is deployed on a parallel architecture of SIMD to

Proposed Fractal Video Sequence Compression Algorithm

Input: video sequence

Output: fractal code of video sequence

Repeat

Take out sequential m frames of images in the video sequence $\{f_1, f_2, \dots, f_m\}$ and combine them into a large image matrix F for holistic compression. Divide F into non-overlapping $N \times N$ range blocks;

A $2N \times 2N$ window with a step size of δ is used to intercept domain block along F ;

Domain blocks are contracted by mean of neighboring four pixels;

According to Algorithm 1, domain blocks are classified and the corresponding range block categories are obtained;

Repeat

Remove a range block R_i from the p -th range block category and set an initial value Error;

Repeat

Remove a domain block D_j from the p -th domain block category, perform eight isometric transformations, and calculate $s, \alpha, E^2(R, D)$ according to Eqs.2-3 and Eq.10;

If $E^2(R, D) < Error$ then

Replace Error with $E^2(R, D)$;

End

Until All domain blocks in the p -th domain block category are completely searched;

Store fractal code $\{x_j, y_j, t_p, s, \alpha_i\}$ of range block R_i ;

Until All range block categories have been matched with their corresponding domain block category;

Store fractal code of the entire image F ;

Until The entire image sequence is compressed and fractal codes of it is obtained.

ALGORITHM 2

improve its efficiency. Thus, a double-parallel video compression framework is built as shown in Figure 5. In the first layer of the parallel framework, all video fragments are allocated to multiple processors. In the second layer, all kinds of range and domain blocks are allocated to multiple computing nodes.

Speed-up ratio and parallel efficiency were used to measure the performance of the parallel algorithm. Their definitions in this paper are shown by Lemma 2.

Assume that A contains f frames and it is divided into n fragments. All range blocks and domain blocks are classified into s categories in each fragment. Compression is deployed in a parallel environment of p processors with c compute nodes per processor.

Lemma 2. Speed-up ratio $S_p(A)$ of the parallel algorithm is $p \cdot c \cdot s \cdot n / f$, and its parallel efficiency is $S_p(A) \cdot f / p \cdot c \cdot s \cdot n$.

Proof. The speed-up ratio is defined as in 20, where $t_s(A)$ is the time required to compress video sequence A by the traditional fractal compression method and $t_p(A)$ is the time required by the parallel method.

$$S_p(A) = \frac{t_s(A)}{t_p(A)}. \quad (20)$$

Let $t_s(A) = T \cdot f$, the time required of each fragment is $T \cdot f/n$.

Because each processor gets n/p fragments, the time required of each processor is $n/p \cdot T \cdot f/n = T \cdot f/p$.

After range blocks and domain blocks are classified and allocated to compute nodes, the time required of each processor is reduced to $T \cdot f/p \cdot 1/s \cdot c = T \cdot f/p \cdot s \cdot c$.

Because each fragment contains f/n frames, the compression time is $T \cdot f/p \cdot s \cdot c \cdot f/n = T \cdot f^2/p \cdot s \cdot c \cdot n$.

Thus, speed-up $S_p(A) = T \cdot f / (T \cdot f^2/p \cdot s \cdot c \cdot n) = p \cdot s \cdot c \cdot n / f$.

Parallel efficiency is defined as in 21, where $S'_p(A)$ is an absolute parallel speed-up without $(s \cdot n)/f$ resulting from the classification of domain and range blocks, and $p(A)$ is the number of compute nodes.

$$E_p(A) = \frac{S'_p(A)}{p(A)}. \quad (21)$$

Since $S'_p(A) = S_p(A) \cdot f/s \cdot n$ and $p(A) = p \cdot c$, parallel efficiency $E_p(A) = S_p(A) \cdot f/p \cdot c \cdot s \cdot n$.

Based on the discussion above, Lemma 2 is proved. Proof is finished.

The amount of computation for compressing arbitrary video data based on a serial algorithm is considerably large. Compared with the traditional serial algorithm, the more the number of processors, the higher the unit calculation efficiency and the higher the parallel computing efficiency. However, it is necessary to consider the actual situation of computing resources and the number of video segments and the class number of domain blocks.

Theorem 3. The computational complexity of parallel algorithm is $O(f/p \cdot M^4/N^2 \cdot c^2/s^2)$.

Proof. According to Theorem 2, the computational complexity of a single frame is $O(M^4/N^2 m^2)$, where m represents the category number of domain and range blocks.

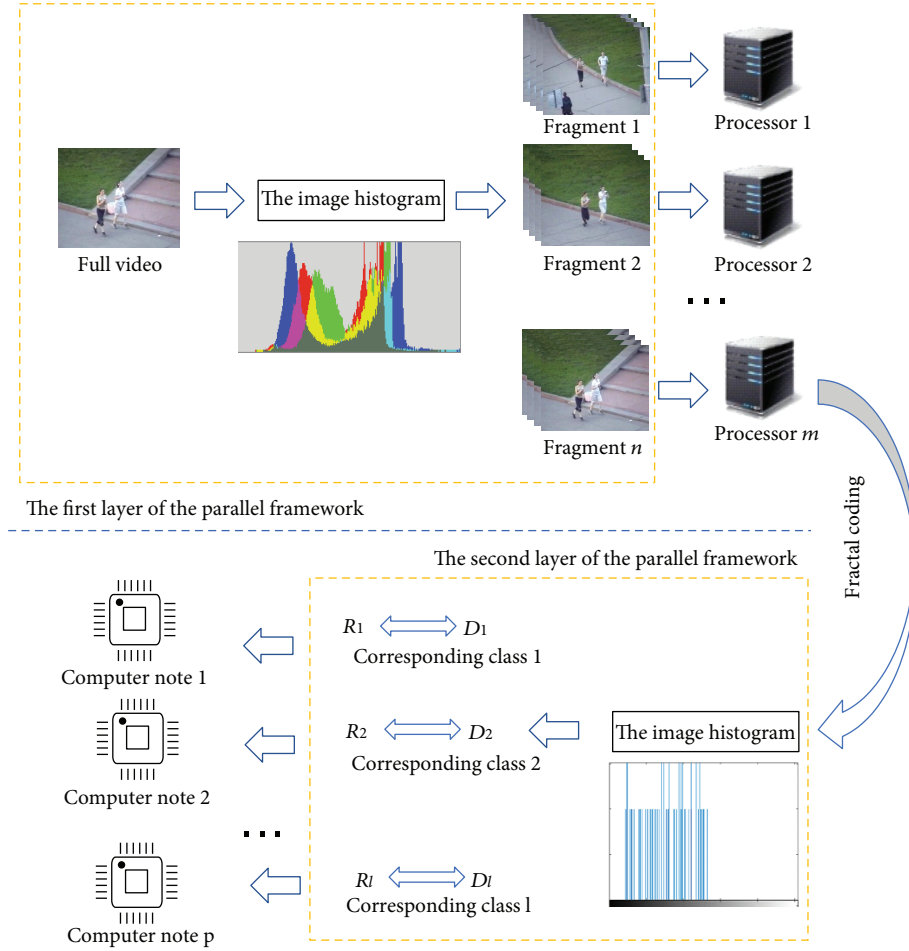


FIGURE 5: Parallel framework.

Therefore, the computational complexity of the parallel algorithm is $O(f_p \cdot M^4/N^2 \cdot 1/m_c^2)$, where f_p is the number of video frames assigned to each processor and m_c is the class number of range and domain blocks assigned to each compute node. In this parallel environment, $f_p = f/p$ and $m_c = s/c$.

Based on the discussion above, Theorem 3 is proved. Proof is finished.

Compared with the computational complexity $O(f/p \cdot M^4/N^2 \cdot c^2/s^2)$ of the parallel method we proposed, the computational complexity of the traditional serial method is $(p \cdot s^2/c^2)$ times as much as it. A reasonable allocation of p , s , and c values can improve parallel efficiency while reducing computational complexity.

4. Experiments and Analysis

In this paper, experiments are performed on the computer with an Intel Core i5-4590 CPU and a 12 GB memory, and the operating environment is Matlab 2016a. Three standard grayscale image sequences are used, which were Walter Cronkite moving head, chemical plant flyover (close view), and chemical plant flyover (far view) [34]. They are renamed

to Seq1, Seq2, and Seq3, respectively. The image size is $256 \times 256 \times 8$ bit. The proposed algorithm is compared with the traditional fractal video sequence compression algorithm from three aspects, which are comparison of single-frame image compression, comparison of sequential frame compression, and comparison of the traditional serial method with the proposed parallel method. In these experiments, the range block has size 4×4 , the domain block has size 8×8 , and both horizontal step and vertical step are 1. Finally, the range block with size 8×8 and the domain block with size 16×16 are added to the experiment when compared with the AVQ_{IS} algorithm [35].

In this paper, a parallel environment consisting of one manager and four processors with four compute nodes each processor is constructed when the traditional serial algorithm is compared with the parallel algorithm. First, the central server fragments the entire video sequence according to its spatial-temporal similarity and sequentially distributes it to processors numbered 1–4 for compression. Then, all range and domain blocks are automatically classified within the video fragment and assigned to four compute nodes for independent work.

We evaluate the quality of the decoded image by calculating 22. The peak signal-to-noise ratio (PSNR) is the logarithm of the mean square error (MSE) between the

Parallel Video Sequence Compression Algorithm Based on Fractal

Input: video sequence

Output: fractal code of video sequence

Fragment: n video fragments $V = \sum_{i=1}^n V_i$ are allocated to multiple processors for compression, which are divided according to the spatial-temporal similarity of video sequence.

Classify: Combine u_i frames $f_{i,j} : V_i = \sum_{j=1}^{u_i} f_{i,j}$ contained in video fragment V_i into a large image matrix F for overall compression;

Divide F into non-overlapping $N \times N$ range blocks;

A $2N \times 2N$ window with a step size of δ is used to intercept domain block along F ;

Domain blocks are contracted by mean of neighboring four pixels;

According to Algorithm 1, domain blocks are classified and the corresponding range block categories are obtained;

Assign all categories of range block and corresponding domain blocks to multiple compute nodes for matching search;

Repeat

Remove a range block R_i from the c -th range block category and set an initial value Error;

Repeat

Remove a domain block D_j from the c -th domain block category, perform eight isometric transformations, and calculate $s, o, E^2(R, D)$ according to Eqs.2-3 and Eq.10;

If $E^2(R, D) < Error$ then

Replace error with $E^2(R, D)$;

End

Until All domain blocks in the c -th domain block category are completely searched;

Store fractal code $\{x_j, y_j, t_j, s_j, o_j\}$ of range block R_i ;

Until All range block categories have been matched with their corresponding domain block category;

Merge: Combine the results of multiple compute nodes to obtain fractal code of video fragment V_i ;

Merge: Combine the results of multiple processors to obtain fractal code of whole video V ;

ALGORITHM 3

TABLE 1: Threshold sequence.

0.000005	0.0000075	0.00001	0.000025	0.00005	0.000075	0.0001	0.00025	0.0005
----------	-----------	---------	----------	---------	----------	--------	---------	--------

original image and the decoded image relative to $(2^n - 1)^2$, where $2^n - 1$ represents the upper limit of the gray level and n is the storage bit of each pixel. The higher the PSNR value, the lesser the distortion.

$$\text{PSNR} = 10 \times \log_{10} \left[\frac{(2^n - 1)^2}{\text{MSE}} \right]. \quad (22)$$

Meanwhile, we use the compression ratio (CR) to measure compression performance by comparing with the AVQ_{IS} algorithm. The compression ratio is defined as the ratio of the original data to the compressed data in 23, where H represents the number of range blocks and $\{8, 8, 3, 5, 7\}$ represents the quantization level of fractal parameters $\{x_j, y_j, t_j, s_j, o_j\}$.

$$\text{CR} = \frac{256 \times 256 \times 8}{H \times (8 + 8 + 3 + 5 + 7)}. \quad (23)$$

4.1. Comparison of Single-Frame Image Compression. Four sequential frames of Seq1, Seq2, and Seq3 are selected to perform frame-by-frame compression. The mean compression time and PSNR of 4 frames are calculated. The threshold sequence is shown in Table 1. With the automatic

classification method, domain blocks were divided into 10 categories. Correspondingly, range blocks were divided into 10 categories. Restored images obtained by the traditional method and proposed method are shown in Table 2. The experimental data is shown in Table 3.

From Tables 2 and 3, it can be seen that the quality of reconstructed images obtained by the proposed algorithm and traditional algorithm is not obviously different from being uncompressed. It indicates that the proposed algorithm is feasible. Moreover, compared with the traditional fractal compression algorithm, although PSNR of the proposed algorithm is slightly decreased, the compression speed is 8~9 times as fast. According to Theorem 2 and its analysis, since range and domain blocks are divided into ten categories, the compression speed should theoretically increase by a factor of ten. Because classification of domain and range blocks is nonuniform, the speed-up ratio of the proposed method is lower than the theoretical value of 10.

4.2. Comparison of Sequential Frame Compression. First, 4 sequential frames of Seq1, Seq2, and Seq3 are compressed by the traditional fractal method. Second, they are compressed by the proposed method frame by frame. Finally, 4 frames are combined to compress together by the proposed method. The threshold sequence is shown in

TABLE 2: Comparison of decoding effects between traditional algorithm and proposed algorithm.


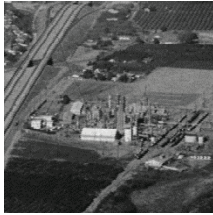







Sequence no.	1	2	3
Uncompressed			
The traditional fractal compression algorithm			
The proposed algorithm			

TABLE 3: Comparison of performance between traditional algorithm and proposed algorithm.

Algorithm	Seq1		Seq2		Seq3	
	T (s)	PSNR (dB)	T (s)	PSNR (dB)	T (s)	PSNR (dB)
Traditional	7939	44.71	7595	34.86	7470	33.59
Proposed	823	43.43	837	33.51	874	32.30

Table 1. After combination, domain and range blocks of Seq1, Seq2, and Seq3 are divided into 20, 15, and 15 categories, respectively. The comparison of performance of the three methods is shown in Table 4, where PSNR is the mean of 4 frames.

From Tables 4 and 5, it can be seen that the image quality of the decoding image obtained by the three algorithms is comparable. Suppose that compressing one frame takes time t . According to Inference 1, when 4 frames are compressed together, the time required becomes $4t$. Since the classification of domain and range blocks can speed up the compression process, the compression time is reduced to $4t/m$. In this experiment, domain and range blocks of Seq1, Seq2, and Seq3 are divided into 20, 15, and 15 categories, respectively. Therefore, the theoretical compression speed is further reduced to $4t/20 \sim 4t/15$, that is, the theoretical value should be 3~5 times. When 4 frames are combined for compression, its compression speed is approximately 2~4 times as fast as the traditional algorithm, which is lower than the theoretical value of 3~5 times. It is because classification of domain and range blocks is nonuniform.

Compared with single-frame compression, the speed-up ratio of the combination algorithm is reduced. However, its image quality is closer to the original image.

4.3. Comparison of Traditional Serial and Proposed Parallel Compression. Video sequence Seq1 contains 16 frames. In the double-layer parallel framework, the distribution of its compression task is shown in Figure 6. Seq1 is divided into 4 video fragments. Processors numbered P1, P2, P3, and P4 get 4, 4, 4, and 4 frames, respectively. Then, all domain blocks are divided into 20 categories, and 4 compute nodes C1, C2, C3, and C4 obtain 5, 5, 5, and 5 categories of range blocks, respectively. Video sequence Seq2 includes 32 frames and Seq3 includes 11 frames. The distribution of Seq2 and Seq3 is shown in Figures 7 and 8. The threshold sequence is shown in Table 6.

Serial computing was used in the traditional fractal video compression method. The performance data of the traditional serial method and the parallel proposed method is shown in Table 7. T is the whole time required of video sequence compression. PSNR is the mean of all images included in each video sequence.

TABLE 4: Comparison of decoding effects of three algorithms.










Sequence no.	1	2	3
The traditional fractal compression algorithm			
The proposed single-frame algorithm			
The combination algorithm			

TABLE 5: Comparison of performance of different strategies for the traditional algorithm and proposed algorithm.

Algorithm	Seq1		Seq2		Seq3	
	T (s)	PSNR (dB)	T (s)	PSNR (dB)	T (s)	PSNR (dB)
Traditional	31,756	44.71	30,380	34.86	29,880	33.59
Single-frame	3292	43.43	3348	33.51	3496	32.30
Combination	8085	44.04	12,231	34.01	12,377	32.92

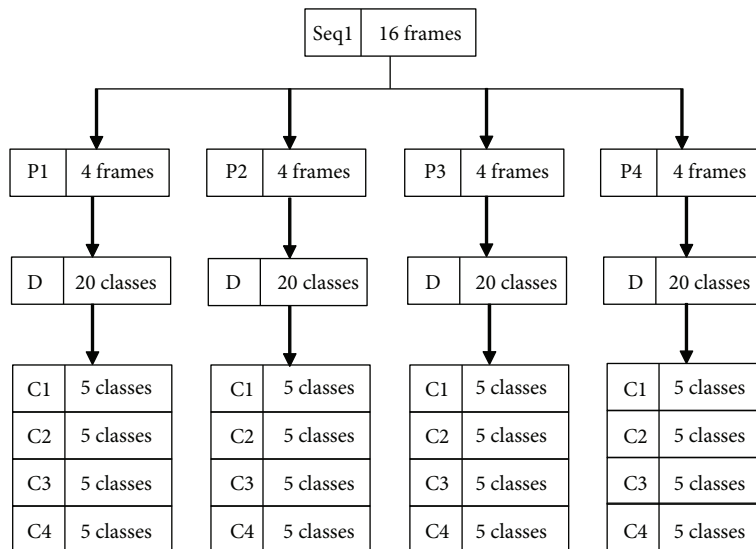


FIGURE 6: The distribution of Seq1.

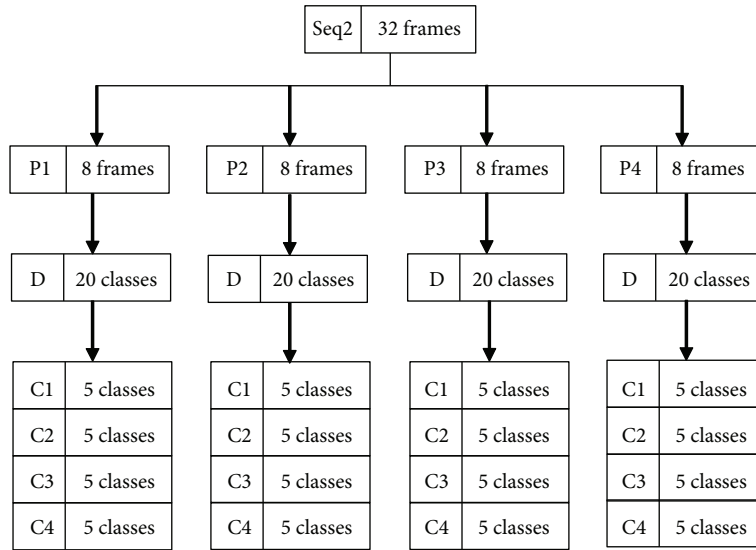


FIGURE 7: The distribution of Seq2.

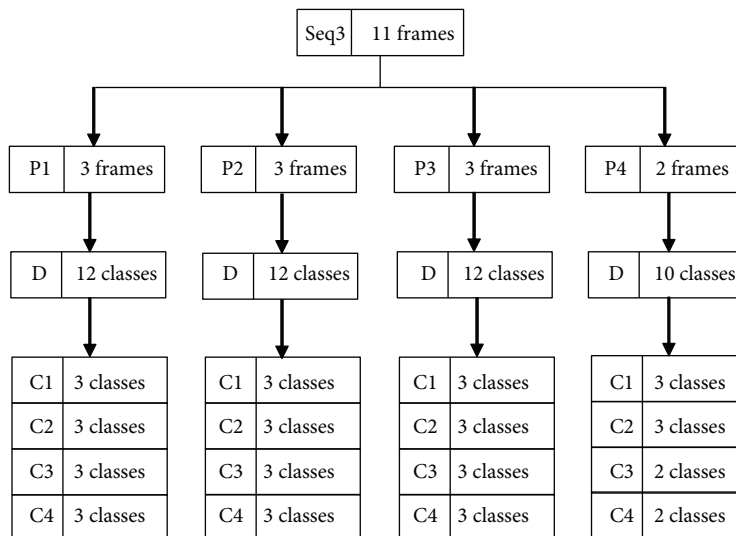


FIGURE 8: The distribution of Seq3.

TABLE 6: Threshold sequence.

0.000001	0.000025	0.000050	0.000075	0.00010	0.000025	0.000050	0.000075	0.0001
----------	----------	----------	----------	---------	----------	----------	----------	--------

TABLE 7: Comparison of performance of the traditional algorithm and the proposed algorithm.

Seq	Algorithm	T (s)	PSNR (dB)	$S_p(A)$	$E_p(A)$
1	Serial traditional	127,034	44.74	62.42	78.03
	Parallel proposed	2035	44.04		
2	Serial traditional	243,055	34.84	26.55	66.38
	Parallel proposed	9154	34.49		
3	Serial traditional	82,175	33.58	40.56	58.09
	Parallel proposed	2026	32.50		

TABLE 8: Comparison of decoding effects of proposed algorithms and the AVQ_{IS} algorithm.










Sequence no.	1	2	3
The proposed algorithm (4 × 4)			
The proposed algorithm (8 × 8)			
The AVQ _{IS} algorithm			

TABLE 9: Comparison of performance of the proposed algorithms and the AVQ_{IS} algorithm.

Algorithm	Seq1		Seq2		Seq3	
	CR	PSNR (dB)	CR	PSNR (dB)	CR	PSNR (dB)
Proposed (4 × 4)	4.13	44.04	4.13	34.49	4.13	32.50
Proposed (8 × 8)	16.52	34.99	16.52	27.56	16.52	26.36
AVQ _{IS}	9.79	31.80	2.44	33.62	2.13	33.75

Table 7 shows that under the premise of ensuring that PSNR is comparable, the mean compression speed of the parallel algorithm achieves more than 40 times that of the traditional serial algorithm, and parallel efficiency averages 67%. In this experiment, the numbers of processors, compute nodes, and fragments are all four, except for the number of frames and the class number of domain blocks. Video sequences Seq1, Seq2, and Seq3 contain 16, 32, and 11 frames, respectively. The class numbers of domain and range blocks are 20, 20, and 12, respectively. According to Lemma 2, the theoretical acceleration ratios for the three sequences are 80.00, 40.00, and 69.82, respectively. Therefore, the theoretical average speed-up ratio is 63. The actual speed-up is less than the theoretical mean speed-up of 63 times because of communication cost. In addition, since classification of range and domain blocks is nonuniform, it causes a difference between actual speed-up and theoretical speed-up.

4.4. Comparison of the AVQ_{IS} Algorithm and Proposed Algorithm. The AVQ_{IS} algorithm was proposed by Pizzolante

et al., which is an extension of the AVQ algorithm. The algorithm utilized the correlation between sequential frames of image sequence to perform lossy compression on image sequence.

Table 8 shows recovery images obtained by decompressing the compressed video sequence using the proposed algorithm and the AVQ_{IS} algorithm. Table 9 shows CR and PSNR obtained by compressing Seq1, Seq2, and Seq3 using the proposed algorithm and the AVQ_{IS} algorithm.

From Tables 7 and 8, it can be seen that compared with the AVQ_{IS} algorithm, the compression ratio of Seq1 is decreased, but PSNR is obviously improved with the range block size of 4 × 4. The compression ratio of Seq2 and Seq3 is higher than that of the AVQ_{IS} algorithm. When the size of the range block is 8 × 8, the compression ratio of all sequences is higher than that of the AVQ_{IS} algorithm.

5. Conclusions

In order to efficiently compress big video data and reduce the computational complexity of the traditional fractal

video compression method, a double-layer parallel video compression framework based on fractals was proposed. In the first layer of the parallel structure, a video sequence was divided into many fragments according to its spatial-temporal similarity. Then, video fragments were allocated to multiple processors for simultaneous compression. In the second layer, a novel fractal video compression method was used to compress video fragments. All domain and range blocks were classified within each fragment. Processors distributed all domain and range blocks to multiple computing nodes for parallel processing.

Experimental results showed that compared with the traditional fractal video compression method, the proposed parallel method significantly improved the compression speed when the image quality was similar. In addition, the compression ratio was higher when compared with the AVQ_{IS} algorithm. It verified the effectiveness of the proposed method.

Future research directions will include two aspects. One is that we will further construct new features to enhance the calculation of video segmentation and classification of the domain block. The other is that based on the concepts of cloud computing and fog computing, we will strive for proposing a more effective parallel fractal compression method for real-time big video data processing.

Data Availability

Image sequences used to support the findings of this study are publicly available at <http://sipi.usc.edu/database/database.php?volume=sequences>. Three sequences consist of 16, 32, and 11 256 × 256 images, respectively.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The research is supported by the National Natural Science Foundation of China (Grant No. 61502254), Program for Young Talents of Science and Technology in universities of the Inner Mongolia Autonomous Region (Grant No. NJYT-18-B10), and open funds of the Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education (Grant No. 93K172018K07).

References

- [1] K. A. Huck and J. Labarta, "Detailed load balance analysis of large scale parallel applications," in *2010 39th International Conference on Parallel Processing*, pp. 535–544, San Diego, CA, USA, September 2010.
- [2] D. Li, J. J. Cao, and Y. Yao, "Big data in smart cities," *Science China Information Sciences*, vol. 58, no. 10, pp. 1–12, 2015.
- [3] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC)," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669–1684, 2012.
- [4] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.
- [5] S. Dhawan, "A review of image compression and comparison of its algorithms," *International Journal of electronics & Communication technology*, vol. 2, no. 1, pp. 22–26, 2011.
- [6] D. Chen and D. Singh, "Fractal video compression in OpenCL: an evaluation of CPUs, GPUs, and FPGAs as acceleration platforms," in *2013 18th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 297–304, Yokohama, Japan, January 2013.
- [7] Z. Chen, W. Lin, and K. N. Ngan, "Perceptual video coding: challenges and approaches," in *2010 IEEE International Conference on Multimedia and Expo*, pp. 784–789, Suntec City, Singapore, July 2010.
- [8] Z. Pan, J. Lei, Y. Zhang, X. Sun, and S. Kwong, "Fast motion estimation based on content property for low-complexity H.265/HEVC encoder," *IEEE Transactions on Broadcasting*, vol. 62, no. 3, pp. 675–684, 2016.
- [9] V. Sze, M. Budagavi, and G. J. Sullivan, *High Efficiency Video Coding (HEVC): Algorithms and Architectures*, Integrated circuits and systems, Springer, 2014.
- [10] Y.-J. Ahn, T.-J. Hwang, D.-G. Sim, and W.-J. Han, "Implementation of fast HEVC encoder based on SIMD and data-level parallelism," *EURASIP Journal on Image and Video Processing*, vol. 2014, no. 1, 2014.
- [11] K. Ugur, A. Alshin, E. Alshina et al., "Motion compensated prediction and interpolation filter design in H.265/HEVC," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 946–956, 2013.
- [12] X. Zhang, Y. Tian, T. Huang, S. Dong, and W. Gao, "Optimizing the hierarchical prediction and coding in HEVC for surveillance and conference videos with background modeling," *IEEE Transactions on Image Processing*, vol. 23, no. 10, pp. 4511–4526, 2014.
- [13] K. Ugur, K. Andersson, A. Fuldseth et al., "High performance, low complexity video coding and the emerging HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 12, pp. 1688–1697, 2010.
- [14] C. W. Seo, J. H. Moon, and J. K. Han, "Rate control for consistent objective quality in high efficiency video coding," *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2442–2454, 2013.
- [15] L. Zhang, Q. Peng, Q. H. Wang, and X. Wu, "Stereoscopic perceptual video coding based on just-noticeable-distortion profile," *IEEE Transactions on Broadcasting*, vol. 57, no. 2, pp. 572–581, 2011.
- [16] Z. Luo, L. Song, S. Zheng, and N. Ling, "H.264/advanced video control perceptual optimization coding based on JND-directed coefficient suppression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 6, pp. 935–948, 2013.
- [17] X. Wang, S. Kwong, L. Xu, and Y. Zhang, "Generalized Nash bargaining solution to rate control optimization for spatial scalable video coding," *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 4010–4021, 2014.
- [18] J. I. Guo, J. S. Wang, J. W. Chen, C. H. Chang, Y. H. O. Yang, and C. C. Lin, "Method for Reducing Computational Complexity of Video Compression Standard," US Patent US8111756, (2012).

- [19] U. S. Potluri, A. Madanayake, R. J. Cintra, F. M. Bayer, S. Kulasekera, and A. Edirisuriya, "Improved 8-point approximate DCT for image and video compression requiring only 14 additions," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 6, pp. 1727–1740, 2014.
- [20] Z. Pan, Y. Zhang, and S. Kwong, "Efficient motion and disparity estimation optimization for low complexity multiview video coding," *IEEE Transactions on Broadcasting*, vol. 61, no. 2, pp. 166–176, 2015.
- [21] M. F. Barnsley and L. P. Hurd, *Fractal Image Compression*, AK Peters, 1993.
- [22] A. E. Jacquin, "Fractal image coding: a review," *Proceedings of the IEEE*, vol. 81, no. 10, pp. 1451–1465, 1993.
- [23] A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," *IEEE Transactions on Image Processing*, vol. 1, no. 1, pp. 18–30, 1992.
- [24] E. W. Jacobs, Y. Fisher, and R. D. Boss, "Image compression: a study of the iterated transform method," *Signal Processing*, vol. 29, no. 3, pp. 251–263, 1992.
- [25] Z. Jiang and M. Y. Jiang, "A fast fractal image compression algorithm based on K-mean clustering optimization," *Journal of Electrical & Electronic Education*, vol. 28, no. 2, pp. 44–46, 2006.
- [26] K. Jaferzadeh, K. Kiani, and S. Mozaffari, "Acceleration of fractal image compression using fuzzy clustering and discrete-cosine-transform-based metric," *IET Image Processing*, vol. 6, no. 7, pp. 1024–1030, 2012.
- [27] Y. G. Wu, M. Z. Huang, and Y. L. Wen, "Fractal image compression with variance and mean," in *2003 International Conference on Multimedia and Expo. ICME '03. Proceedings (Cat. No.03TH8698)*, Baltimore, MD, USA, July 2003.
- [28] D. M. Monro and F. Dudbridge, "Fractal approximation of image blocks," in *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 485–488, San Francisco, CA, USA, March 1992.
- [29] C. S. Tong and C. M. Wung, "Adaptive approximate nearest neighbor search for fractal image compression," *IEEE Transactions on Image Processing*, vol. 11, no. 6, pp. 605–615, 2002.
- [30] T. K. Truong, C. M. Kung, J. H. Jeng, and M. L. Hsieh, "Fast fractal image compression using spatial correlation," *Chaos, Solitons & Fractals*, vol. 22, no. 5, pp. 1071–1076, 2004.
- [31] Y. L. Lin and M. S. Wu, "An edge property-based neighborhood region search strategy for fractal image compression," *Computers & Mathematics with Applications*, vol. 62, no. 1, pp. 310–318, 2011.
- [32] X. Wang, D. Zhang, and X. Guo, "Novel hybrid fractal image encoding algorithm using standard deviation and DCT coefficients," *Nonlinear Dynamics*, vol. 73, no. 1-2, pp. 347–355, 2013.
- [33] Y. Wu, J. Lim, and M. H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.
- [34] University of Southern California (USC), "Signal and Image Processing Institute, Volume 4: sequences of The USC-SIPI Image Database," <http://sipi.usc.edu/database/database.php?volume=sequences>.
- [35] R. Pizzolante, B. Carpentieri, and S. De Agostino, "Adaptive vector quantization for lossy compression of image sequences," *Algorithms*, vol. 10, no. 2, p. 51, 2017.

Copyright © 2018 Shuai Liu et al. This is an open access article distributed under the Creative Commons Attribution License (the “License”), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. Notwithstanding the ProQuest Terms and Conditions, you may use this content in accordance with the terms of the License. <http://creativecommons.org/licenses/by/4.0/>