EUROGRAPHICS Workshop on Multimedia (2004)
N. Correia, J. Jorge, T. Chambel, Z. Pan (Editors)

Progressive Image Transmission Using Singular Value Decomposition

Chin-Chen Chang and Yi-Long Liu

Department of Computer Science and Information Engineering
National Chung Cheng University

Abstract
There are many progressive image transmission schemes using vector quantization-related schemes, but there are some problems associated with them. First, these schemes need to train a codebook, and this codebook will directly decide the quality of the recovered image. VQ-related PIT methods also need some time to search the codebook to find the indices of corresponding vectors. Thus, this paper looks for a new PIT method without VQ. By applying SVD to PIT, image will be decomposed into three matrices. After decomposition, certain processes are applied to these matrices, and PIT will be achieved by transmitting these three matrices. The use of the proposed method resulted in higher image quality than would result from the use of traditional methods, like the bit-plane method and the improved bit-plane method, in the experiments conducted in this paper. This paper proposes a new way of PIT, and this new method is achieved PIT without VQ.

Categories and Subject Descriptors (according to ACM CCS): I.5.4 [Applications]: Signal processing

1. Introduction

Recently, many image progressive transmission (PIT) researches use vector quantization-related schemes [CheCha97], [HCC03], [JCC97]. However, the schemes using VQ need to train a good codebook from a large number of images first. Then they also need to search this codebook to get the indices of corresponding vectors for each image. When VQ is applied to the PIT scheme, it also needs to transmit the codebook first. The sender and receiver need to have the same codebook to achieve the PIT. Thus, a codebook is good or not directly affects the quality of the PIT.

Image compression using VQ-based SVD, proposed in [CSC21], [YanLu95], compresses the decomposed matrices using VQ. It can achieve PIT by transmitting each index of each block. But it still needs to train the codebook. Furthermore, it needs to train two codebooks for two decomposed matrices and also needs to search vectors from these two codebooks.

This paper looks for a new PIT scheme without using VQ. Traditional PIT schemes without VQ are the bit-plane method (BPM) and the improved bit-plane method (IBPM) [CSC99]. BPM and IBPM need to transmit all phases to have a clear image. In the low phases of these two methods, there are less representative colors. Although the PSNR of the recovered image is very high in some phases, the recovered image still looks bad. But PIT using the SVD method will result in clear images in the low phases. And the amount of data transmitted using the SVD method is less than those using BPM and IBPM, respectively. Thus, the proposed method is an effective PIT method.

Singular value decomposition (SVD) is a linear algebra matrix decomposition method [GVL89] and is widely used in many areas like watermarking [HTH01], data hiding [CSC21], and noise estimation [KNY97]. SVD will decompose a matrix into three matrices, and when these three matrices are multiplied, the original matrix will be obtained.

In this paper, SVD is applied to each block divided from an image and decomposes each divided block into three matrices. After each block is decomposed, three new matrices are generated. And after some processing, these three matrices are used to achieve the PIT.

The rest of this paper is organized as follo+ws. In Section 2, the property of singular value decomposition and the traditional non-VQ PIT methods BPM and IBPM are introduced. The proposed SVD-based PIT scheme is described in Section 3. Section 4 describes the experimental results. The conclusions are given in Section 5.

2. Related Works

Singular value decomposition is a matrix decomposition method. It can be applied to all kinds of matrices even if a matrix is not square. It will decompose a matrix into three matrices. Given a matrix M of size m*n, when SVD is applied to M, SVD will decompose M into three matrices: U, D, and V. The equation is shown below:

$$M = U*D*V^T \quad (V^T \text{ means the transpose of V})$$
$$=[u_1,u_2,\ldots,u_m]*\begin{bmatrix} d_1 & 0 & . & 0 \\ 0 & d_2 & . & . \\ . & . & . & . \\ . & . & . & 0 \\ . & . & . & d_n \\ 0 & . & . & 0 \end{bmatrix}*[v_1,v_2,\ldots,v_n]^T, \quad (1)$$

where U and V are square matrices. The sizes of U, V, and D are m*m, n*n, and m*n, respectively. Here U and V are real unitary matrices ($U*U^T=1$, $V*V^T=1$), and D is a diagonal matrix with nonnegative entries. But in this paper, only a square matrix is considered, so only a matrix where m=n is used. The values $d_1,d_2,\ldots,d_n$ in the diagonal part of D are called the singular values of M and are also the nonnegative square root of the eigenvalues of M^TM.

$$(M^T*M)*x_i=d_i*x_i \text{ for } i=1,2,\ldots,r , \quad (2)$$

where r is the rank of M, r≤n, and x_i is the corresponding eigenvector of d_i. If the rank of the matrix M is r, the number of nonzero singular values will be r, and the singular values d_i's in the diagonal of D will satisfy

$$d_1 \geq d_2 \geq \ldots \geq d_r \geq d_{r+1} = d_{r+2} = \ldots = d_n = 0 \quad (3)$$

U is composed of $u_1,u_2,\ldots,u_n$ and each u_i=[$u_{i1}, u_{i2},\ldots, u_{in}$]^T, 1 ≤ i ≤ n, {$u_1,u_2,\ldots,u_r$} is the orthonormal basis of the column space of M, where r ≤ n. V is the same as U, where v_i is a collection of orthonormal basis of the column space of M^T and 1 ≤ i ≤ r. Then { $u_{r+1},u_{r+2},\ldots,u_n$ } and { $v_{r+1},v_{r+2},\ldots,v_n$ } are the orthonormal basis of the kernel space of M^T and the kernel space of M, respectively. Because $d_{r+1}=d_{r+2}=\ldots=d_n=0$, Equation (1) can be rewritten as follows:

$$M=U*D*V^T=[u_1,u_2,\ldots,u_n]*\begin{bmatrix} d_1 & 0 & . & . & 0 \\ 0 & d_2 & & & . \\ . & & & . & . \\ . & & & . & 0 \\ 0 & . & . & 0 & d_n \end{bmatrix}*[v_1,v_2,\ldots,v_n]^T=\sum_{i=1}^{r} d_i*u_i*v_i^T \quad (4)$$

Using Equation (4), the image progressive transmission method can be achieved. Given a matrix of size n*n, the matrix will be decomposed into three matrices, each of size n*n, by applying SVD on it. And these three matrices can be separated into r parts. After these r parts are combined, the original matrix will be recovered.

Because u_i and v_i are orthonormal, they satisfy the property $\|u_i\|$=1 and $\|v_i\|$=1. By this property, we can prove that -1 ≤ u_{ik}, v_{ik}≤ 1 , for i=1,2,...,n and k=1,2,…,n .

$\|u_i\| = u_i^{T}* u_i$ =[$u_{i1}, u_{i2},\ldots, u_{in}$] * [$u_{i1}, u_{i2},\ldots, u_{in}$]^T=1 for i=1,2,..n

= $u_{i1}^2+ u_{i2}^2+\ldots+ u_{in}^2$, $u_{ik}^2 \geq 0$ and $\|u_i\|$=1 for k=1,2,..,n

=> $u_{ik}^2 \leq 1$. That is, $|u_{ik}| \leq 1$.

for i=1,2,..n and k=1,2,..,n

So we have $-1 \leq u_{ik} \leq 1$.

for i=1,2,..n and k=1,2,..,n (5)

The number of progressive phases that a matrix with n*n entries needs is r, where r≤n. If a matrix has a full rank n, it will need n phases to recover the original image. But in fact, a matrix with size 8*8 will always have a high PSNR in the second and the third phases. Thus, it is not necessary to transmit a full phase. Only a half or a less number of phases is needed so that a high PSNR can be obtained.

3. The Proposed Method

In this section, we propose a new method using a non-VQ PIT to obtain a higher quality image in fewer phases with a lower amount of transmission. The method is called SVD progressive image transmission. This method can be separated into four parts: the decomposition part, the quantization part, the sending part, and the combination part.

Figure 1 shows the flowchart of the SVD PIT.

The first step of our new method is to decompose an image using SVD and obtain three matrices U, D, and V. In the second step, according to Property (5), the matrices U and V will be quantized, and D will also be quantized but in a different way. Then the matrices Uq, Dq, and Vq will be obtained. After quantization, the total memory size needed to store these three matrices can be a little smaller than the original image. The third step is to send the information step by step. The fourth step is to combine the information received in every phase.

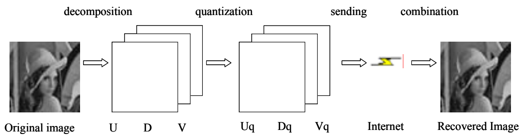


Figure 1: *The flowchart of SVD PIT*

3.1 Decomposition Step

An image I of size w*h is first divided into several blocks B_i, where $0 \le i \le (w*h)/(k*k)$, with each block of size k*k. SVD is applied to each B_i. Then each B_i is decomposed into two matrices Ub_i and Vb_i and one vector Db_i. Ub_i and Vb_i are matrices of size k*k. Db_i is a one-dimension vector of size k. All Ub_i's will form the matrix U, and all Vb_i's will form the matrix V. U and V will be the same size as image I. Because, when SVD is applied, the matrix which contains singular values is a diagonal matrix, only a one-dimensional vector is needed to store the singular values. Then the matrix D is of size [(w*h)/(k*k)]*k, where (w*h)/(k*k) is the quantity of B_i.

Determining the value of k is important in this step. The value k will affect the transmitted bit per pixel (bpp) in each transmission phase. If the value k is increased, the bpp of the transmission phase will decrease. The decrease of the bpp will be illustrated in Section 4.2.

Figure 2 illustrates the decomposition of each B_i. The decomposed matrices Ubi and Vb_i will be stored in U and V, Db_i is stored in the matrices D. After decomposition, the memory size needed to store the image will be larger, so quantization is needed to process the decomposed matrices U, D, and V.

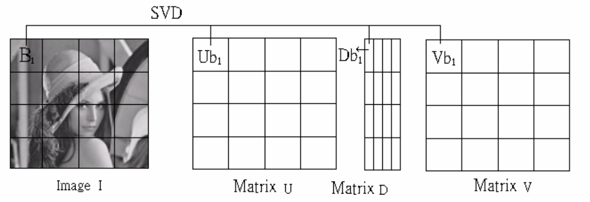


Figure 2: *The decomposition of B_i*

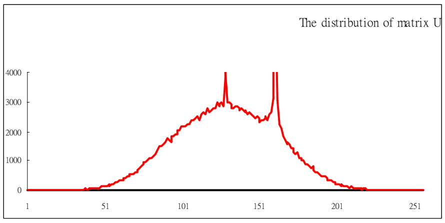
3.2 Quantization Step

Because the memory size of the blocks after SVD will be larger, this step is used to reduce the amount of decomposed matrices. According to Equation (5), all elements in U and V will be between -1 and 1. Although the elements in U and V are all floating point values, fewer bits that less than the original floating point values need can be used to represent them.

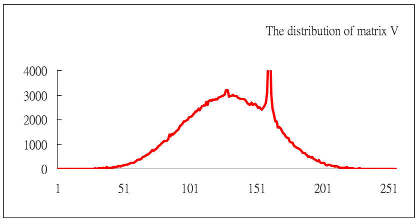
3.2.1 Direct 8-bit Quantization

In this paper, we first directly use 8-bit quantization on U and V. By using 8 bits, the quantization levels will be 256. Thus, the interval (-1,1) is divided into 256 levels, and all values in U and V are quantized into these 256 levels. This is the simplest method of quantization, but from observation of the distribution of all values in U and V, they are likely to be a normal distribution. Assume a distribution of the U and V matrices of the image Lena of size 512*512, with block size 16*16 (k=16). The quantization levels from 1 to 256 are on the x-axis. The length of each interval is 1/256, and the y-axis gives the times of elements appear in each level. Almost all values are centered around -0.5 to 0.5, that is ($-2/\sqrt{k}$, $2/\sqrt{k}$), and a large number of values are centered from $-1/\sqrt{k}$ to $1/\sqrt{k}$. As shown in Figure 3, there are tip points around $1/\sqrt{k}$. This happens only on the positive side. From observation on our experiment, the values in the first column of each block in Ub_i and Vb_i are all positive and almost are centered around $1/\sqrt{k}$. So the numbers of values around $1/\sqrt{k}$ are very large. That is why the tip points occurred around $1/\sqrt{k}$. According to the above property, the quantization can be revised. In this paper, we quantize the values which are less than $-2/\sqrt{k}$ and larger than $2/\sqrt{k}$ using only six levels, and the main 250 levels are centered in the interval $(-2/\sqrt{k}, 2/\sqrt{k})$.

Quantization is also applied to matrix D. The quantization of matrix D is direct. The decimal of each element in matrix D is truncated directly. That is because the singular values in D are the multiplication coefficients of each matrix $u_i{*}v_i^T$, which is shown in Equation (4). Because the values in u_i and v_i are between -1 and 1, the values in $u_i{*}v_i^T$ are quite small. If the value of a corresponding singular value is not large enough, the effect will be very small. So we truncate its decimal directly. The decimal of each singular value will not affect the quality of the transmitted image too much. But after this processing, the needed memory size is still larger than the size needed for the original image.



(a) The distribution of matrix U



(b) The distribution of matrix V

Figure 3: *The distribution of the 512*512 Lena image with block size 16*16*

3.2.2 Truncation

Because SVD has a property like the discrete cosine transformation's energy centralization, singular values from d_1 to d_n in Db_i can be seen as the DC to AC in DCT. The larger singular value is more important, so we can truncate some unimportant singular values and their corresponding u_i and v_i. In this way, memory size will be saved, and the quality of this image will not be distorted too seriously. This means that, in the original method shown in equation (4), a matrix is composed of $u_1{*}d_1{*}v_1^T + \ldots + u_r{*}d_r{*}v_r^T$, where $r \le n$. Now, however, we use only $u_1{*}d_1{*}v_1^T + \ldots + u_l{*}d_l{*}v_l^T$, where $1 \le l \le r$, to represent the original matrix and truncate u_{l+1}, d_{l+1}, v_{l+1}, …, u_r, d_r, v_r. In this way, memory size will be saved.

When an image is decomposed, a threshold value T should be calculated first. The value T represents the average number of singular values which are larger than the given threshold H in a block in matrices U and V. Then we will keep only the first T triples (u_i, d_i, v_i)'s , where $1 \le i \le T$, and use them to recover the original matrix.

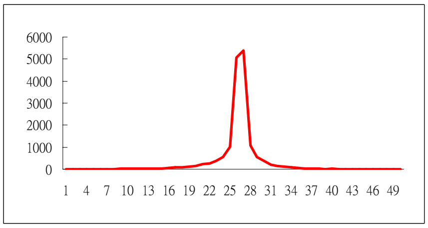
The threshold value H is always given by k because a large number of values in U and V are centered around $-1/\sqrt{k}$ to $1/\sqrt{k}$. The values of each $u_i{*}v_i^T$ will always be less than 1/k. Thus, if d_i is less than k, the entry of this matrix $d_i{*}\, u_i{*}v_i^T$ will always be less than 1. The influence of this matrix and the matrices after it are not so important. Accordingly, if the value T is 6.7, the 8^{th} to k^{th} columns of each block in U and V will be pruned away. In the same way for D, the first 7 singular values will be kept and the 8^{th} to k^{th} singular values will be pruned away in each block. Thus, the needed memory for the image will be reduced. By this method, the needed memory size is reduced to a little less than the original image.

3.2.3 Bit Reduction Quantization

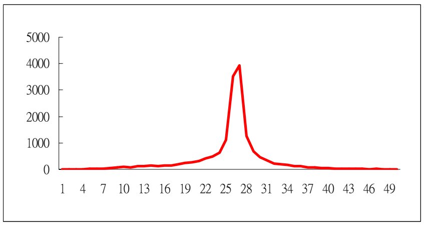
As shown in Figure 3, there are some properties that can be used to reduce the amount of transmission. In the first phase, all first columns of each block in matrices U and V are transmitted during the transmission phase. From our experiment, we observed that all the values in the first column of each block in the U and V matrices are all positive, where those values are between 0 and 1. The distribution of all first columns in each block in matrices U and V are shown in Figure 4. Because the amount of values which is larger than 0.5 is quite few, the graph only shows the range from 0 to 0.5. A lot of values are around $1/\sqrt{k}$, where k is equal to 16. Thus, lots of values are around 0.25.

According to this property, the first phase can be reduced further. The range of all values in the first phase is smaller than those of all values in the other phases, so fewer bits can be used to quantize the values in the first phase. According to our experiment, instead of 8 bits being directly used, five bits are enough for the first phase. When five bits are used, there are thirty-two levels that can be used to quantize the values. Lots of values are centered around $1/\sqrt{k}$, so that twenty-six levels are used to quantize the values around $1/\sqrt{k}$, and the remaining six levels are used to quantize the values that are far from $1/\sqrt{k}$. Also, the mean values for each level are calculated. These mean values need to be transmitted, too, and are used to reduce distortion. The amount of the first phase needed for transmission is reduced.

The importance of a block from the second phase to the T^{th} phase decreases. The larger singular value is more important, so bit reductions are used in the second phase to the T^{th} phase. The second phase is the most important phase, aside from the first phase, so 7 bits are used in this phase. The third phase uses 6 bits, and the fourth phase to the T^{th} phase (the last phase) uses 5 bits to quantize their corresponding blocks. In this way, the amount needed for transmission is reduced for the whole image.



(a) The distribution of all first columns of each block in matrix U



(b) The distribution of all first columns of each block in matrix V

Figure 4: *The distribution of the first phase of 512*512 Lena with block size 16*16*

3.3 Sending Step

After the quantization step, an image is decomposed into three quantized matrices Uq, Vq, and Dq. When sending the first phase, a sender sends all the first columns in each block in Uq and Vq and all the first items of each vector d (all the first singular values). In the second phase, the sender will send all the second column of each block in Uq and Vq and all the second items of each vector d, and so on. When singular values are sent, because all singular values in the same phase are close to each other, and the singular values in the higher phase are very small, fewer bits can be used for the singular value in the higher phase. All the first singular values are very large. When they are sent, the largest singular value in the first phase needs to be found because the number of bits a singular value requires for the first phase needs to be decided. Similarly in each phase, the largest singular values in each phase need to be found in order that the number of bits needed to represent them can be determined.

3.4 Combination Step

In each phase, each block of a receiver will receive two columns u_i and v_i and one value d_i to represent one block. By multiplying $d_i{*}u_i{*}v_i^T$ for each block, we will produce a matrix with size k*k, where k*k is the block size. In each phase, there are three items, which are d_i, u_i, v_i in each block, and we use $d_i{*}u_i{*}v_i^T$ to represent the original block in the recovered image.

The first phase is the base phase. All the calculated blocks are called base blocks, and all the base blocks form the recovered image. The second phase will add the blocks calculated in this phase to the base blocks, and the added block will become the new base blocks. All the new base blocks also form the recovered image for this phase. Similarly, in the third phase, the blocks will be calculated and added to the base blocks, and so on. After the third phase, the value $\sum_{i=1}^{3} d_i{*}u_i{*}v_i^T$ is used to represent the block. In each phase and each block, two columns and one value will be used to calculate a matrix (block), and this matrix added to the base block will generate a new base block. After all the phases are processed, the recovered image will be generated.

4. Experiments

In this section, some experimental result shows. The experiment includes different size of images, which are images Lena, Baboon, Pepper and Airplane of sizes 256*256 and 512*512. The images are processed with a different value k, and all the results are shown below.

4.1 Progressive Transmission Result

Figure 5 shows the recovered images for each phase by using the bits reduction quantization method. The image is Lena with 512*512 pixels. The block size is 16*16. H=16 and T is rounded up to 6. The first phase uses the received mean values to recover the original quantization level. In our proposed method, from the second phase to the last phase, the mean values for each phase can be calculated, but calculation is not required. In this experiment, we only transmit the mean values for all quantization levels in the first phase, and from the second phase we use a fixed quantization level to recover the blocks.

Figure 6 shows the maximum d value in each phase. The table shows that the needed bits for the d values can be reduced in each phase. In the first phase of Figure 6 (a), the d values need twelve bits for each value, and in the second phase the d values need ten bits for each value. The needed bits for the d values can be decided by their maximum value in each phase.

Figure 7 shows the needed T for each image with a different k. Value T indicates the number of phases an image with different k is needed for transmission to have good effect. In our experiment, the 512*512 Lena image is the principal example. Its T with k=16 is 5.272. In Figure 5, T is rounded up to 6. So six phases are used to store and transmit this image.

4.2 A Comparison of BPM and IBPM

Figure 8 shows a comparison of BPM and IBPM using the method of bits reduction quantization. In the first three phases, the proposed method has a higher PSNR than BPM and IBPM. After the fourth phase, the PSNR will become lower than them. Although BPM and IBPM have a higher PSNR than the proposed method, they only have sixteen kinds of colors to represent an image in the fourth phase. The recovered image will look bad. But in our proposed method, it is clear and with full colors in the third phase. When the bpp (bits per pixel) is compared with BPM and IBPM, the proposed method has a total 3.062 for a 256*256 image and a 3.039 for a 512*512 image in the first four phases, only a little higher than BPM and IBPM in the first three phases. The total amount of transmission in the bits reduction method in the first four phases is almost the same as the total amount of transmission in BPM and IBPM in the first three phases. So if the PSNR in the fourth phase of the bits reduction method is compared to the PSNR in the third phase of BPM and IBPM, the PSNR of the bits reduction method is higher than those of the BPM and the IBPM.

The bpp of the proposed method is fixed from phase four to the last phase. The direct 8-bits method will have a fixed bpp in all phases. The bits reduction method will have a different bpp in each phase. But both methods have the same property in that if k is increased, the bpp of each phase will decrease. Increasing the block size will decrease the bpp for each phase, but more phases will be needed to transmit all of an image. Figure 9 shows the bpp for a different block size k and the same k. If k is decreased, the bpp will increase, but fewer phases will be needed to transmit all of an image.

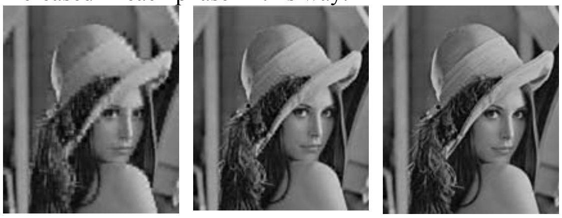
Comparing Figure 9 and Figure 7, if we want to decrease the bpp in each phase, we can increase the block size k. For example, if k=32, the image Lena needs eight phases in order to be a clear image, but in each phase the bpp is decreased to around 0.37. Increasing the block size will cause some block effects in the sharp area of the image. If the whole image is a smooth image, a bigger value k is suitable and will not cause a serious block effect. If the block size k needs to be increased to a larger number, the direct 8-bits method will have a better performance and will not cause a serious block effect.

4.3 Comparison with the Direct 8-bits method

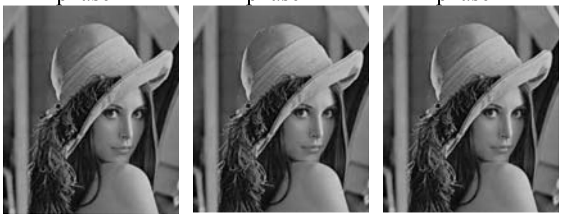
The PSNR of the proposed method is close to the original SVD method shown in Figure 10 (a). The original SVD method does not use quantization but only decompose an original block into three real number matrices. It wastes too much space in storing the matrices. The proposed method needs a lower storage memory and fewer bpp for transmission in each phase, but its PSNR will be close to that of the original SVD method.

5. Conclusions

Most progressive image transmission methods use VQ-related methods. This paper proposed a new way to progressively transmit an image without VQ processing. Unlike the traditional BPM and IBPM methods, the proposed method will have a higher PSNR in the lower phases and require a lower amount of transmission. This method is time-consuming in the decomposition step and the combination step. Decomposition using SVD has been researched for a long time. There are some papers about speedup of SVD [HsiDel96], so the decomposition step is not the problem. Similar to the combination step, matrix (vector) multiplication has been researched for a long time. In [BAI88] a speedup method was proposed. Thus, there are models for SVD decomposition and combination. The problem of taking too much time in these two steps has thus been resolved. From the experimental results, we see that the proposed method is an efficient and effective method in progressive image transmission and can obtain a clear recovered image in fewer phases. As the experiment shows, a big block size k can reduce the bpp of an image, and a small block size will have a better performance in the low phases. Thus, we can use different block sizes of an image in our future work. A large block size k is used in smooth areas while a small k is used in sharp areas. Therefore, the bpp can be reduced even further, and the PSNR will also be increased in each phase in this way.



(a) The first phase (b) The second phase (c) The third phase



(d) The fourth phase (e) The fifth phase (f) The sixth phase

Figure 5: *The recovered image of 512*512 Lena with block size 16*16 in each phase*

References

[BAI88] BAILEY, D. H.: Extra-High Speed Matrix Multiplication on the Cray-2. *SIAM Journal on Scientific and Statistical Computing*, Vol. 9, No. 3, (1988), 603-607.

[CSC99] CHANG, C. C., SHINE, F. C., CHEN, T. S.: A New Scheme of Progressive Image Transmission Base on Bit-Plane Method. *Proceedings of the Fifth Asia-Pacific Conference on Communications and Fourth Optoelectronics and Communications Conference (APCC/OECC 99')*, Vol. 2, Beijing, China, (1999), 892-895.

[CheCha97] CHEN, T. S. AND CHANG, C. C.: Progressive Image Transmission Using Side Match Method. *Information Systems and Technologies for Network Society*, World Scientific Publishing Co. Pte. Ltd., Singapore, (1997), 191-198.

[CSC21] CHUNG, K. L., SHEN, C. H., CHANG, L. C.: A novel SVD- and VQ-based image hiding scheme. *Pattern Recognition Letters*, Vol. 22, No. 9, (2001), 1051-1058.

[GVL89] GOLUB, G. H. AND VAN LOAN, C. F..: Matrix Computations. *The Johns Hopkins University Press*, Baltimore, MD (Chapter 8), (1989).

[HsiDel96] HSIAO, S. F. and DELOSME, J. M.: Parallel singular value decomposition of complex matrices using multidimensional CORDIC algorithms. *IEEE Transactions on Signal Processing*, Vol. 44, No. 3, (1996), 685 -697.

[HTH01] HSIEH, M. S., TSENG, D. C. AND HUANG, Y. H.: Hiding Digital Watermarks Using Multiresolution Wavelet Transform. *IEEE Transactions on Industrial Electronics*, Vol. 48, No. 5, (2001), 875-882.

[HCC03] HUNG, K. L., CHANG, C. C. AND CHEN, T. S.: A Side-Match Reconstruction Method Using Tree Structure VQ for Transmitting Image Progressively. *To appear in IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, (2003).

[JCC97] JIANG, J. H., CHANG, C. C. AND CHEN, T. S.: Selective Progressive Image Transmission Using Diagonal Sampling Technique. *Proceedings of International Symposium on Digital Media Information Base*, Nara, Japan, (1997), 59-67.

[KNY97] KONSTANTINIDES, K., NATARAJAN, B. AND YOVANOF, G. S.: Noise Estimation and Filtering Using Blocked-Based Singular Value Decomposition. *IEEE Transactions on Image Processing*, Vol. 10, No. 3, (1997), 479-483.

[YanLu95] YANG, J. F. AND LU, C. L.: Combined Techniques of Singular Value Decomposition and Vector Quantization for Image Coding. *IEEE Transactions on Image Processing*, Vol. 4, No. 8, (1995), 1141-1146.

	512*512 (k=16)					
Image name	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6
Lena	3028	757	443	233	199	128
Baboons	3122	559	434	286	267	189
Pepper	3420	718	499	206	148	114
Airplane	3565	670	509	261	144	87

(a) The maximum d value in each phase with block size 16*16

	512*512 (k=32)					
Image name	Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6
Lena	5896	1205	821	549	486	357
Baboons	6051	761	600	501	480	379
Pepper	6464	1486	889	452	391	317
Airplane	7124	1535	824	490	379	270

(b) The maximum d value in each phase with block size 32*32

Figure 6: *The maximum d value in each phase*

H=k	256*256		512*512	
Image name	16*16	32*32	16*16	32*32
Lena	6.543	11.922	5.272	8.031
Baboons	11.402	20.703	11.191	19.879
Pepper	5.844	10.719	6.793	10.211
Airplane	5.945	10.766	4.668	7.590

Figure 7: *The needed T for each image with a different k*

256*256	Phase 1			Phase 2		
Image name	SVD	BPM	IBPM	SVD	BPM	IBPM
Lena	22.737	16.822	18.001	26.432	22.676	23.328
Baboons	22.380	17.613	19.256	24.304	23.334	23.806
Pepper	23.216	16.253	21.201	27.794	22.445	23.206
Airplane	22.720	19.809	21.245	26.466	22.83	26.003
Avg.	22.764	17.624	19.926	26.249	22.821	24.086
BPP	0.703	1	1	0.914	1	1

(a) The comparison of Phases 1 and 2 of a 256*256 image

256*256	Phase 3			Phase 4		
Image name	SVD	BPM	IBPM	SVD	BPM	IBPM
Lena	28.808	28.622	29.077	30.371	34.766	34.917
baboons	25.867	28.647	29.043	27.128	34.686	34.926
pepper	30.597	28.809	29.127	31.989	34.848	34.872
airplane	29.136	29.385	29.532	30.787	34.813	35.1
Avg.	28.602	28.866	29.195	30.069	34.778	34.954
BPP	0.785	1	1	0.660	1	1

(b) The comparison of Phases 3 and 4 of a 256*256 image

512*512	Phase 1			Phase 2		
Image name	SVD	BPM	IBPM	SVD	BPM	IBPM
lena	25.895	16.274	18.098	29.803	22.791	23.131
baboons	21.085	16.63	20.565	23.201	22.348	23.148
pepper	26.116	17.609	19.167	30.497	23.202	23.773
airplane	25.033	19.951	21.463	29.381	22.85	26.067
Avg.	24.532	17.616	19.823	28.220	22.798	24.030
BPP	0.680	1	1	0.914	1	1

(c) The comparison of Phases 1 and 2 of a 512*512 image

512*512	Phase 3			Phase 4		
Image name	SVD	BPM	IBPM	SVD	BPM	IBPM
lena	32.513	28.847	29.304	34.088	34.924	35.116
baboons	24.982	28.819	29.151	26.445	34.836	34.889
pepper	32.836	28.698	29.053	33.938	34.723	34.958
airplane	32.147	29.365	29.525	33.678	34.826	35.105
Avg.	30.619	28.932	29.258	32.037	34.827	35.017
BPP	0.785	1	1	0.660	1	1

(d) The comparison of Phases 3 and 4 of a 512*512 image

Figure 8: *PSNR compare with BPM and IBPM in each phase*

R-bits	Phase 1	Phase 2	Phase 3	Phase 4
8*8	1.424	1.891	1.625	1.359
32*32	0.333	0.448	0.385	0.322

Figure 9: *BPP with a different k*

No quantization (16*16)				
Image name	Phase 1	Phase 2	Phase 3	Phase 4
Lena	26.106	30.369	33.974	36.777
Baboons	21.141	23.304	25.333	27.226
Pepper	26.399	31.364	34.860	37.085
Airplane	25.283	30.139	34.121	37.442

(a) The PSNR from the original SVD method

Direct 8-bits (16*16)				
Image name	Phase 1	Phase 2	Phase 3	Phase 4
Lena	26.000	30.081	33.327	35.610
Baboons	21.093	23.219	25.191	27.002
Pepper	26.243	30.873	33.817	35.454
Airplane	25.057	29.468	32.590	34.628

(b) The PSNR from the direct 8-bits method

Figure 10: *The PSNR from the direct 8-bits method and the original SVD method*