An image reconstruction algorithm based on classified block

Yuanwei Zhang*
Tibet Museum of Natural Science, Lhasa, Tibet, 850000, China
*Corresponding author’s e-mail: kangjian1@ha.chinamobile.com

Abstract. This paper constructs a compression image reconstruction algorithm (ABCS-MMSE). ABCS-MMSE overcomes the defects of compression sensing sampling. It not only constructs the block measurement matrix at encoding end but also generates the block recovery matrix at decoding end. Experimental results show that the average reconstruction time of ABCS-MMSE is much lower than that of BCS-SPL-DDWT and that of BCS-SPL-DCT. At the same time, for the images with smooth information, the reconstruction quality of ABCS-MMSE is not much worse than that of BCS-SPL-DDWT and that of BCS-SPL-DCT.

1. Introduction
First give the signal $x \in \mathbb{R}^N$, the perception matrix $\Phi$ (such as its size is $M \times N$, and $M \ll N$) and the measurement vector $y = \Phi x$. The definition of compression rate is $R = M/N$. According to CST [1,2], the signals which have sparse representation in some linear transformation domains can be accurately recovered from measurements of these signals with very high probability. In this case, $x = \Psi c$, where $\Psi$ is a sparse inverse transformation, $c$ is a sparse coefficient vector and has only $S$ ($S \ll N$) nonzero thresholds; the recovered signal can be synthesized by $\tilde{x} = \Psi \tilde{c}$, and there is the following expression:

$$\tilde{c} = \arg \min_{c} \| \alpha \|_{1} \text{ subject to } y = \Phi \Psi \tilde{c}$$

(1)

whereby $\| \alpha \|_{1}$ is a $\ell_1$ norm, and calculates the number of non-zero thresholds about $\alpha$ .

An image that has $D = L \times L$ pixel values can be transformed into a one-dimensional vector $x \in \mathbb{R}^{L^2}$ using column stacking or row stacking. In the meantime, the dimension of $\Phi$ is $M \times L^2$ and the dimension of $\Psi$ is $L^2 \times L^2$.

Adaptive sampling mechanism based on the reweighted block Compressive Sensing (BCS) schemes allocates the CS-measurements to image blocks according to the statistical information of each block[5,6]. A Decomposition-based CS-recovery framework (DCR) utilizing residual reconstruction and state-of-the-art filters iteratively refines residual measurement. Decomposition based Texture preserving Reconstruction (DETER) is on the basis of the priors of DCR which are a weighted total variation and nonlocal structures in the gradient domain. DETER has solved the preserving small-scale textures in reconstructing image at low subrate[9]. An altered adaptive BCS algorithm with flexible partitioning and error analysis has used the weighted mean information entropy that is adopted as the basis for partitioning BCS. The local saliency and variance is introduced to step-less adaptive sampling that can distinguish and sample between smooth blocks and textural blocks[10]. A JPEG lifting algorithm based on adaptive BCS, that adopts mean information entropy and multifeature
saliency indexes, works out the fusion between the adaptive BCS and the JPEG compression and optimizes the compression rate of one same image[11]. A method that can improve the global image reconstruction performance was proposed in[3]; this method used a BCS sampling strategy which was also used in[3,4][7,8], and estimated the entire image reconstruction performance after embedding direct transformations; these direct transformations are Contourlet Transform(CT)-Smooth Projected Landweber(SPL)[3] and Dual-tree Discrete Wavelet Transform (DDWT)-SPL[3].

The rest of this paper is organized as follows. Section 2 gives a brief traditional block compressed sensing sampling and a detailed adaptive block measurement and reconstruction strategy. Section 3 gives detailed experiment simulation results and analyses; In section 4, conclusions are drawn.

2. Basic Principles

2.1. Block-based Traditional Compression Sampling

In block compressed sensing sampling (BCSS)[3,4][7,8], an image is decomposed into q blocks which are non-overlapping, the size of each image block is $B \times B$, and every block can be individually compressed and sensed. The measurement vector of j-th block $y_j$ is as follows:

$$y_j = \Phi_B x_j$$ (2)

Where $x_j \in \mathbb{R}^{B \times B} (j = 1, \cdots, q)$ is a set of column vector blocks that are stacked and $\Phi_B (\Phi_B^T \Phi_B = I, I$ is a unit matrix,) is an orthonormal independent identical distribution Gaussian matrix in equation (3). Thus, the $\Phi$ of a whole image is a block diagonal matrix, using the following expression:

$$\Phi = \begin{pmatrix} \Phi_B & 0 & \cdots & 0 \\ 0 & \Phi_B & \vdots \\ \vdots & \ddots & 0 \\ 0 & \cdots & 0 & \Phi_B \end{pmatrix}$$ (3)

Where the row number of $\Phi$ is $q_B \times q$, and the column number of $\Phi$ is $(B^2) \times q$. The size of $\Phi$ can also be expressed as $n \times D$, where $n$ is a compressed sensing measuring value that is assumed to be obtained and $D$ has the same meaning as discussed above.

2.2. Adaptive Block Compressed Sensing

2.2.1. Block Classification Criterion. According to block classification criterion, an image can be divided into smooth blocks, texture blocks and transition blocks. These three types of blocks respectively have different characteristics, so they can be assigned the corresponding sampling rates according to the corresponding block characteristics. Here, we classify these blocks by calculating the standard variance of each block[3-8].
\[ d_j = \left( \sum_{i=1}^{B \times B} (p_{ji} - \bar{p}_j)^2 \right) / (B \times B), \text{where } \bar{p}_j = \left( \sum_{i=1}^{B \times B} p_{ji} \right) / (B \times B) \]  

(4)

\[ \hat{d}_j = (d_j - d_{\min}) / (d_{\max} - d_{\min}) \]  

(5)

\[ d_{\min} = \min \{d_1, \cdots, d_j, \cdots, d_q\}, \quad d_{\max} = \max \{d_1, \cdots, d_j, \cdots, d_q\} \]  

(6)

where \( p_{ji} \) is the i-th pixel value of the j-th block; \( \bar{p}_j \) is the average pixel value of the j-th block; \( d_j \) is the pixel value variance of the j-th block; \( d_{\min} \) and \( d_{\max} \) is respectively the minimum value and maximum value in q block pixel value variances.

if \( \hat{d}_j \leq T_1, x_j \in \text{smooth block}; \) 
if \( \hat{d}_j > T_2, x_j \in \text{textural block}; \) 
if \( T_1 < \hat{d}_j \leq T_2, x_j \in \text{transitive block}. \)  

(7)

where \( T_1 \) and \( T_2 \) is respectively 0.1 and 0.3.

Figure 1 shows the subjective visual renderings of Barbara after implementing the above block classification criterion, where a black block represents a smooth block, a grey block is on behalf of a transition block and a white block is on behalf of a texture block.

2.2.2. Adaptive Block Measurement and Reconstruction Based on Minimum Mean Square Error.

According to the above block classification criteria, this paper allocates the corresponding measurement rate for a particular type of block in order to build an adaptive measurement reconstruction method. A higher measurement rate should be assigned to the texture blocks; a lower measurement rate should be assigned to the smooth blocks; a transitive measurement rate between the higher measurement rate and lower measurement rate should be assigned to the transitive blocks. Therefore, adaptive block compression sensing sampling method can be formulated as follows:

\[ y_k = \Phi^k_B \ast x_k, \quad y_i = \Phi^i_B \ast x_i, \quad y_m = \Phi^m_B \ast x_m \]  

(8)

Here, \( x_k \) is a column vector made up of a stack of smooth block columns; \( \Phi^k_B \) is a smooth block measurement matrix formed by the front KN row which is extracted from \( \Phi \); a smooth block measurement value \( y_k \) \( (k = 1, \cdots, KN) \) is applied; \( KN \) denotes the total number of smooth blocks. This paper uses a linear estimation method based on Minimum Mean Square Error (MMSE) to estimate the measurement values[7,8].
Here, $y_k$ is a one-dimensional vector formed by the measurement values; $\Phi_B^{-k}$ represents the recovery matrix of the smooth blocks; $\tilde{x}_k$ represents the reconstruction estimator for the smooth blocks; $R_{xx}^{-k}$ represents the autocorrelation function of the input smooth blocks. The concrete expression of $R_{xx}^{-k}$ is given as following:

$$R_{xx}^{-k} = R_{xx}^{-k} = E(xx^T) = \begin{pmatrix} E(x_1x_1) & E(x_1x_2) & \cdots & E(x_1x_{B^2}) \\ E(x_2x_1) & E(x_2x_2) & \cdots & E(x_2x_{B^2}) \\ \vdots & \vdots & \ddots & \vdots \\ E(x_{B^2}x_1) & E(x_{B^2}x_2) & \cdots & E(x_{B^2}x_{B^2}) \end{pmatrix}$$ (12)

$$R_{xx}^{-k}(k,l) = E(x_kx_l) = \lambda^\eta, \eta = \text{dis} \text{tan}(x_k, x_l) = \sqrt{(h_k - h_l)^2 + (p_k - p_l)^2}$$ (13)

Here, the autocorrelation functions of the smooth blocks, the transitive blocks and the textural blocks are all identical to $R_{xx}^{-k}$. The position coordinates of $k$-th pixel element $x_k$ are $(h_k, p_k)$; the position coordinates of $l$-th pixel element $x_l$ are $(h_l, p_l)$; $\lambda$ is the correlation coefficient between the pixel elements, and there is $0.9 < \lambda < 1$; $\eta$ is the Euclidean distance between $x_k$ and $x_l$.

3. **Experiment Environments and Analyses of Results**

This paper’s computing platform has Intel Core i3-3110M running with 64-bit Windows10, 4GB RAM and Matlab 2012a.

From table 1, it can be seen that, for Goldhill, PSNR of ABCS-MMSE is higher than that of BCS-SPL-DDWT and BCS-SPL-DCT from 0.1 to 0.5. For Mandrill, PSNR of ABCS-MMSE is higher than that of BCS-SPL-DDWT and BCS-SPL-DCT from 0.2 to 0.5. For Barbara, PSNR of ABCS-MMSE is lower than that of BCS-SPL-DDWT and BCS-SPL-DCT from 0.1 to 0.5. For Peppers, PSNR of ABCS-MMSE is 1.4729dB lower than that of BCS-SPL-DDWT and is 1.2563dB higher than that of BCS-SPL-DCT on average.
Table 1. PSNR(dB) and Reconstruction time(s) comparison of the three reconstruction algorithms

| Image   | M/N | ABCS-MMSE | BCS-SPL-DDWT | value 1 | BCS-SPL-DCT | value 2 |
|---------|-----|-----------|--------------|---------|-------------|---------|
| Barbara | 0.1 | 22.5524/0.4649 | 22.5922/67.3290 | -0.0398 | 22.7609/27.1066 | -0.2085 |
|         | 0.2 | 24.0188/0.4504 | 24.1247/37.8756 | -0.0159 | 24.3750/21.9456 | -0.3562 |
|         | 0.3 | 25.3935/0.6193 | 25.6865/15.6474 | -0.2930 | 25.9223/17.0831 | -0.5288 |
|         | 0.4 | 26.8808/0.8181 | 27.2116/14.4057 | -0.3308 | 27.4994/22.5704 | -0.6186 |
|         | 0.5 | 28.6470/1.0434 | 28.8205/11.5050 | -0.1735 | 29.1438/15.3826 | -0.4968 |
| Average |     | 25.4985/0.6792 | 25.6871/29.3525 | -0.1886 | 25.9403/20.8177 | -0.4418 |

| Peppers | 0.1 | 28.0025/0.4433 | 28.9578/45.3891 | -0.9552 | 26.2246/31.5859 | +1.7779 |
|         | 0.2 | 30.3692/0.5132 | 31.9931/23.1698 | -1.6239 | 29.2477/25.2413 | +1.1215 |
|         | 0.3 | 32.0505/0.7723 | 33.7187/15.9940 | -1.6682 | 30.7209/29.0948 | +1.3296 |
|         | 0.4 | 33.5019/1.0769 | 35.1228/12.2000 | -1.6209 | 32.1411/29.6675 | +1.3608 |
|         | 0.5 | 34.8862/1.5334 | 36.3826/9.5368 | -1.4964 | 34.1945/30.3361 | +0.6917 |
| Average |     | 31.7621/0.8678 | 33.2350/21.2579 | -1.4729 | 30.5057/29.1851 | +1.2563 |

| Mandrill | 0.1 | 20.3636/0.3132 | 20.6726/44.6029 | -0.3090 | 20.2210/20.9059 | +0.1426 |
|          | 0.2 | 22.0232/0.3952 | 21.8536/18.7782 | +0.1696 | 20.9865/31.9823 | +1.0367 |
|          | 0.3 | 23.4975/0.5460 | 22.8960/12.5586 | +0.6015 | 21.9298/28.9856 | +1.5677 |
|          | 0.4 | 24.9604/0.7247 | 23.9609/10.5340 | +0.9995 | 23.0687/25.4606 | +1.8917 |
|          | 0.5 | 26.5878/0.9638 | 25.0922/9.3603 | +1.4956 | 24.4873/12.3121 | +2.1005 |
| Average  |     | 23.4865/0.5886 | 22.8951/19.1668 | +0.5914 | 22.1386/23.9293 | +1.3478 |

| Goldhill | 0.1 | 27.2169/0.4804 | 26.9076/40.9999 | +0.3093 | 26.3556/26.9429 | +0.8613 |
|          | 0.2 | 29.3579/0.5102 | 28.8981/24.4013 | +0.4598 | 28.1425/22.5457 | +1.2154 |
|          | 0.3 | 30.9880/0.6763 | 30.3680/15.1769 | +0.6200 | 29.6116/24.2125 | +1.3764 |
|          | 0.4 | 32.5066/1.0857 | 31.7038/11.7866 | +0.8030 | 30.4676/29.0459 | +2.0392 |
|          | 0.5 | 34.0748/1.4543 | 33.0549/9.4849 | +1.0199 | 32.2936/18.4863 | +1.7812 |
| Average  |     | 30.8289/0.8374 | 30.1865/20.3699 | +0.6424 | 29.3742/24.2477 | +1.4547 |

In this table, the data are obtained under 32*32 image block division. PSNR of BCS-SPL-DDWT subtracted from that of ABCS-MMSE is the value 1. PSNR of BCS-SPL-DCT subtracted from that of ABCS-MMSE is the value 2.

Figure 2. At compression rate 0.1, the reconstructed Barbara, Peppers, Mandrill and Goldhill

As can be seen from table 1, the reconstruction running time of ABCS-MMSE is far lower than that of BCS-SPL-DDWT and BCS-SPL-DCT. From figure 2, it can be seen that ABCS-MMSE not only eliminate shadow and block artifacts but also acquire not bad subjective visual quality.

4. Conclusions
ABCS-MMSE which uses linear projection to reconstruct an original image has gone beyond the traditional BCS. The experimental results show that ABCS-MMSE significantly reduces the reconstruction running time, and the image reconstruction quality of ABCS-MMSE is still better than that of BCS-SPL-DDWT and BCS-SPL-DCT, which is for the images with more smooth blocks. Even if ABCS-MMSE processes the image with more textural blocks, the image reconstruction quality of ABCS-MMSE is not worse than that of BCS-SPL-DDWT and BCS-SPL-DCT. ABCS-MMSE is the first approach to be considered in processing many huge size images, transmission, and decoding.


Acknowledgements

This work was supported by Natural Science Foundation of Tibet(XZ2018ZRG-140).

References

[1] Donoho,D.L.(2006)Compressed sensing.IEEE Transactions on Information Theory,52:1289-1306
[2] Candès,E.J.,Wakin,M.B.(2008) An introduction to compressive sampling.IEEE Signal Processing Magazine, 25: 21-30.
[3] Mun,S., Fowler,J.E. (2009) Block compressed sensing of images using directional transforms. In:16th IEEE International Conference on Image Processing. Cario. pp. 3021-3024.
[4] Gan,L. (2007) Block compressed sensing of natural images. In:15th International Conference on Digital Signal Processing. Cardiff. pp. 403-406.
[5] Zhu,S.Y., Zeng,B., Gabbouj,M. (2014)Adaptive reweighted compressed sensing for image compression. In:IEEE International Symposium on Circuits and Systems. Melbourne. pp. 1-4.
[6] Zhu,S.Y., Zeng,B., Gabbouj,M. (2015) Adaptive sampling for compressed sensing based image compression. Journal of Visual Communication and Image Representation, 30: 94-105.
[7] Wang,A.H., Liu,L., Zeng,B., Bai,H.H. (2011) Progressive image coding based on an adaptive block compressed sensing.IEICE Electronics Express, 8: 575-581.
[8] She,Q.S., Luo,Z.Z., Zhu,Y.P., Zou,H.B., Chen,Y. (2009) Spatially adaptive image reconstruction via compressive sensing. In:IEEE 7th Asian Control Conference.Hong Kong. pp. 1570-1575.
[9] Canh,T.N., Dinh,K.Q., Jeon,B. (2016) Compressive sensing reconstruction via decomposition. Signal Processing:Image Communication, 49: 63-78.
[10] Zhu,Y.J., Liu,W.B., Shen,Q. (2019) Adaptive algorithm on block-compressive sensing and noisy data estimation.Electronics, 8: 753-778.
[11] Zhu,Y., Liu,W.B., Shen,Q., Wu,Y., Bao,H. (2020) JPEG lifting algorithm based on adaptive block compressed sensing. Mathematical Problems in Engineering, 2020: 1-17.