LETTER

Adaptive Block-Wise Compressive Image Sensing Based on Visual Perception

Xue ZHANG[†], ***Nonmember***, **Anhong WANG**[†a)], ***Member***, **Bing ZENG**[††b)], **Lei LIU**[†], ***and*** **Zhuo LIU**[†], ***Nonmembers***

SUMMARY Numerous examples in image processing have demonstrated that human visual perception can be exploited to improve processing performance. This paper presents another showcase in which some visual information is employed to guide adaptive block-wise compressive sensing (ABCS) for image data, i.e., a varying CS-sampling rate is applied on different blocks according to the visual contents in each block. To this end, we propose a visual analysis based on the discrete cosine transform (DCT) coefficients of each block reconstructed at the decoder side. The analysis result is sent back to the CS encoder, stage-by-stage via a feedback channel, so that we can decide which blocks should be further CS-sampled and what is the extra sampling rate. In this way, we can perform multiple passes of reconstruction to improve the quality progressively. Simulation results show that our scheme leads to a significant improvement over the existing ones with a fixed sampling rate.

key words: *compressive sensing, human visual perception, adaptive sampling, DCT*

1. Introduction

The theory of compressive sensing (CS) has initiated a tremendous wave of activity in the recent research of processing sparse signals [1]–[3]. Due to the sparseness – an intrinsic property in many signals in practice, CS can sample a signal at a rate much lower than the Nyquist rate while still enabling a nearly exact reconstruction.

The sampling used in CS is usually implemented via a random matrix. When applied to a 2-D image whose size is usually quite big, however, CS would become very impractical because a huge memory is needed to store the random sampling operator and the reconstruction is very expensive computationally. Efficient solutions have been proposed recently to these problems. For instance, several fast algorithms [4]–[6] have been developed to speed up the CS reconstruction. Meanwhile, a block-wise compressive sensing (BCS) scheme [7], [8] is proposed, which needs a very small and fixed (cross all blocks) sampling matrix so as to reduce the memory requirement significantly. This BCS scheme focuses on comparing several reconstruction algorithms, such as gradient projection for sparse reconstruction (GPSR), sparsity adaptive matching pursuits (SAMP), total variation (TV) based, and smoothed projected Landweber (SPL) with different transforms (e.g., DCT, DWT, and contourlet transform). However, we notice that the image acquisition is always implemented at a fixed sampling rate, without considering the diversified contents in various blocks. To overcome this drawback, we suggest in our recent work [9] an adaptive block-wise compressive sensing (ABCS) technique to achieve the progressive reconstruction. In our ABCS method, we first apply BCS at a very low (and fixed) rate to produce a base-layer. Then, each image block will be further CS-sampled, but at varying rates, according to image contents within each block.

More specifically, each block is classified into "*Smooth*", "*Texture*", and "*Other*", by examining the normalized variance of each block against some pre-determined thresholds. Such a classification is performed at the encoder side using the original image and thus some overheads are needed to signal the block type. In this work, we propose to shift the classification to the decoder side to avoid overheads and keep the encoder extremely simple. The analysis result will be sent back to the encoder side, stage-by-stage, via a feedback channel.

2. The Compressive Sensing Principle

Given a real-valued signal $\boldsymbol{x} \in \mathbb{R}^N$ (of length N), it is said to be sparse if there exists a basis matrix $\Psi \in \mathbb{R}^{N\times N}$ (of size $N \times N$) such that $\boldsymbol{x} = \Psi\boldsymbol{\alpha}$, where $\|\boldsymbol{\alpha}\|_0 = K \ll N$ (i.e., $\boldsymbol{\alpha}$ has K non-zero coefficients only). The CS theory tells us that such a K-sparse $\boldsymbol{x}$ can be reconstructed (with certain accuracy) by M measurements:

$$\boldsymbol{y} = \Phi\boldsymbol{x} = \Phi(\Psi\boldsymbol{\alpha}), \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^M$ denotes the measurement-vector of length M, Φ is an $M\times N$ measurement (or sampling) matrix that is incoherent with Ψ, and $M = \mathrm{O}\,(K \log(N/K))$, $K < M \ll N$.

Clearly, it is quite difficult to run CS directly on an image of size $N\times N$, simply because the image's size is too big. In the BCS method, an original image is first partitioned into non-overlapped blocks of size $W \times W$. Then, each block $\boldsymbol{x}_i$ (where i stands for the block's index) is sampled with the same CS operator to produce m measurements:

$$\boldsymbol{y}_i = \Phi_W \cdot \boldsymbol{x}_i, \tag{2}$$

where Φ_W is an $m \times W^2$ measurement matrix, $m = M \times$

Manuscript received July 30, 2012.
Manuscript revised October 13, 2012.

[†]The authors are with Institute of Digital Media and Communication, Taiyuan University of Science and Technology, Taiyuan, 030024, China.
[††]The author is with Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, 518057, Hong Kong.
a) E-mail: wah_ty@yahoo.com.cn
b) E-mail: eezeng@ust.hk

DOI: 10.1587/transinf.E96.D.383

W^2/N^2, and $m \ll W^2$. The equivalent sampling operator Φ appeared in Eq. (1) for the whole image is a block-wise diagonal matrix composed by Φ_W. It is clear that the BCS method requires much less storage.

3. Compressive Image Sensing Based on Visual Perception

3.1 ABCS Driven by Visual Perception

The proposed compressive image sensing scheme based on visual analysis is shown in Fig. 1: An original image is divided into $W \times W$ blocks; each block is first CS-sampled at a very low (and fixed) rate and then reconstructed to get the 1^{st}-stage image (the base-layer). Here, the reconstruction is accomplished by the smoothed projected Landweber (SPL) algorithm with DCT as the transform [8]. Afterward, each block will be decided whether being CS-sampled again according to the visual analysis result at the decoder side.

After the visual analysis, blocks are classified into PLAIN, EDGE, and TEXTURE. Then, we take advantage of a feedback channel to send the classification result back to the encoder. According to the block type, we allocate different rates to further CS-sample various blocks. Comparing with our earlier work [9] (that classifies blocks into "*Smooth*", "*Texture*", and "*Other*"), we would like to highlight the following: The classification in [9] is done on each original image at the encoder side. In a practical CS scenario, this is not realistic because each original image stands only in the real world and thus is not available before being sensed. Therefore, any classification (if needed) can only be done at the receiver (or decoder) side upon the sensed image – exactly as what is suggested in this work.

After the classification, the bit-allocation employed in [9] follows the ordering: "*Texture*" > "*Other*" > "*Smooth*". Nevertheless, it is known that human visual perception is not as purely linear as this. For instance, experience tells us that: (1) when an image is compressed at a very low rate, most blocks become very smooth and human eyes now are more sensitive to the so-called blocking effect across neighboring smooth blocks rather than texture or other blocks; and (2) the blocking effect will be diminishing gradually when a higher rate is applied, whereas texture and other blocks might still experience a serious distortion and thus human eyes now will pay more attention to them. As a result, the bit allocation cannot be as linear as described above; instead, a more sophisticated compromising is needed, see Sect. 3.3 for the details.

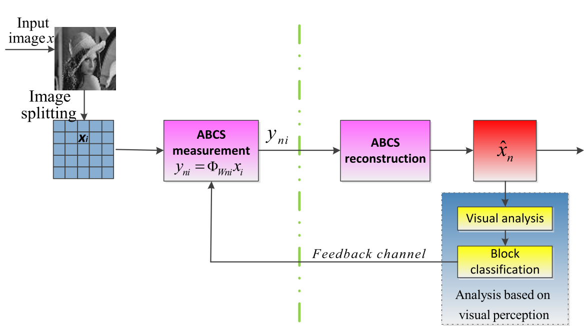

Fig. 1 Compressive image sensing based on visual perception.

More differences between the current ABCS scheme and the one presented in [9] are as follow:

- The current scheme shifts the classification to the decoder side. By doing so, we can keep the encoder to be extremely simple to support applications such as distributed video coding (DVC). This has also saved all overheads that are required to be sent through the down-link channel, as the block-type information is transmitted back through an up-link channel.
- The classification in the current scheme is based on human visual perception which emphasizes EDGE (an important visual pattern); whereas the previous one only involves block-wise statistics of the 1^{st} and 2^{nd} orders.
- The current classification needs to be run at each stage whenever a newly reconstructed image is obtained at the CS decoder. Thus, the block type may change from one to another. For instance, most blocks will be classified as PLAIN at the 1^{st} stage; but some of these PLAIN blocks will be reclassified as EDGE or TEXTURE when an extra sampling rate is applied.

3.2 Block Classification

A visual model based on the block-wise DCT coefficients has been proposed in [10]. Here, an image is divided into 8×8 blocks and each block is DCT-transformed. Each block of DCT coefficients is divided into four indicative areas as shown in Fig. 2, where the absolute sums of the DCT values in these four areas are denoted as DC, L (low frequency), E (edge), and H (high frequency), respectively. Then, each block is assigned to the PLAIN, EDGE, or TEXTURE class according to relations between the values of L, E, and H, and some pre-determined thresholds, see [10] for the details. We follow this model to perform the block classification in this work.

3.3 Dynamic Bit-Allocation

Since the classification employed here is based on the reconstructed image at the decoder side, most blocks will be classified as PLAIN at the 1^{st} stage, including many blocks

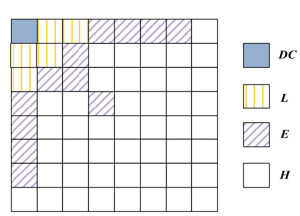

Fig. 2 Block classification employed in our work: four indicative areas (with different marks) are fixed for all blocks.

that would be otherwise classified as EDGE or TEXTURE if the original image is used for classification (as we did in [9]). Thus, it is natural that a higher rate should be assigned to these PLAIN blocks at the 2^{nd} stage. As the sampling rate increases, some of these PLAIN blocks will be re-classified as EDGE or TEXTURE and thus they continue to deserve a higher rate at the following stages. At the same time, the visual quality of those blocks that remain to be PLAIN will be improved significantly at the 2^{nd} stage so that any additional random samples won't be able to yield a noticeable change visually. Based on these discussions, the bit-allocation should have dynamic characteristics as follows:

At the 2^{nd} stage: TEXTURE ≤ EDGE < PLAIN - nearly all bits are allocated to PLAIN blocks;

At the 3^{rd} stage: TEXTURE < EDGE - zero bits are allocated to PLAIN blocks;

At the 4^{th} stage: TEXTURE > EDGE.

Notice that, in this work, we do not try to find the optimal rate-allocation at various stages. Instead, our focus is to demonstrate that such a dynamic allocation is necessary, while leaving the optimal allocation problem to our future work.

4. Simulation Results

We choose the BCS method proposed in [8] as the comparison benchmark. However, we modify the block size to be 8×8 and choose the DCT as the transform for the reconstruction. Notice that (1) it usually leads to a better performance if a larger block size is used (e.g., 32×32 is used in [8] and [9]) and (2) other transforms such as CT (which has been used in [8] and [9]) and DWT (which is also included in [8]) are able to yield a better performance, too. For all experimental results presented below, we tried to maintain all selections as simple as possible so as to focus on solely demonstrating the effectiveness of adopting a dynamic bit-allocation strategy.

Four random samples are taken for each block at the 1^{st} stage; and extra random samples will be added whenever a new stage is needed. At the end of each stage, we count the total number of random samples, and then average it over all blocks to calculate an equivalent rate $R_{\rm BCS}$ in order to run the corresponding BCS (with the fixed rate $R_{\rm BCS}$ to all blocks).

Table 1 lists how many random samples are taken at various stages for different block types where a few test images have been included. Notice that the percentages in parentheses are the portions of various blocks belonging to different classes. The corresponding $\Delta R_{\rm BCS}$ is calculated according to these percentages at each stage and then accumulated to get $R_{\rm BCS}$.

Fig. 3 presents the block classification maps for *Lena*. It can be observed that, with the increase of sampling rate, the map gets closer to the one obtained from the original image.

Table 2 presents the PSNR comparisons of the ABCS and BCS methods. To differentiate various blocks, we present in Table 2 the PSNR values for three types of blocks. As the sampling rate increases, although we end up with a PSNR loss in all PLAIN blocks, the visual effect does not vary much when compared with the results achieved by the corresponding BCS method. At the same time, a much bigger PSNR improvement over the BCS method has been achieved by our ABCS method for all EDGE and TEXTURE blocks. Over each entire image, we have experienced a PSNR loss only at the 2^{nd} stage, but achieved a significant gain at the 3^{rd} and 4^{th} stages.

To have some visual comparisons between the BCS and ABCS methods, we show in Fig. 4 some enlarged portions for *Lena*, from which one can perceive a very noticeable improvement by using our ABCS method. Notice that, for a more fair comparison, we need to count the overhead bits

Table 1 Numbers of random samples taken for different blocks at various stages (and their corresponding percentages).

	Stage	PLAIN	EDGE	TEXTURE	$\Delta R_{\rm BCS}$	$R_{\rm BCS}$
Lena (256×256)	1^{st}	4 (76%)	4 (19%)	4 (5%)	4.0	4
	2^{nd}	8 (70%)	0 (21%)	0 (9%)	6.0	10
	3^{rd}	0 (68%)	22 (17%)	15(15%)	6.0	16
	4^{th}	0	10	29	6.0	22
Cameraman (256×256)	1^{st}	4 (71%)	4 (17%)	4 (12%)	4.0	4
	2^{nd}	8 (74%)	2 (14%)	0 (12%)	6.0	10
	3^{rd}	0 (73%)	26 (11%)	19 (16%)	6.0	16
	4^{th}	0	18	25	6.0	22
Barbara (512×512)	1^{st}	4 (81%)	4 (13%)	4 (6%)	4.0	4
	2^{nd}	7 (69%)	3 (16%)	0 (15%)	6.0	10
	3^{rd}	0 (67%)	20 (12%)	18 (21%)	6.0	16
	4^{th}	0	3	27	6.0	22

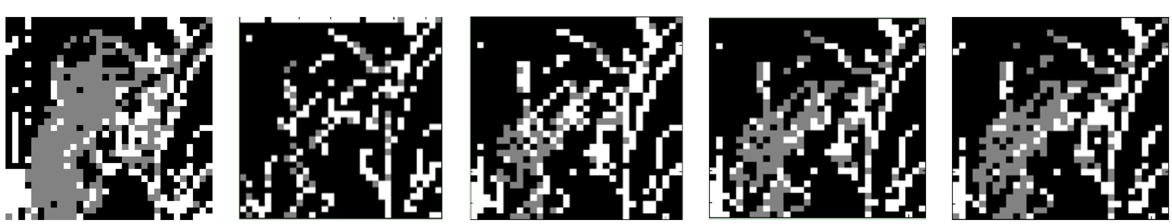
(a) (b) (c) (d) (e)

Fig. 3 Block classification maps for *Lena* (from left to right): using the original image, and ABCS reconstructed images at $R_{\rm BCS}$ = 4, 10, 16, and 22, respectively ("black" for PLAIN, "grey" for TEXTURE, and "white" for EDGE).

Table 2 PSNR at various stages for different test images.

Averaged no. of random samples per block			4	10	16	22
Lena (256×256)	Entire image	ABCS	20.05	22.26	27.37	29.91
		BCS		24.51	26.98	28.50
	PLAIN	ABCS		27.76	28.34	29.66
		BCS		26.93	31.05	33.46
	EDGE	ABCS		17.31	27.46	29.15
		BCS		21.37	24.16	26.39
	TEXTURE	ABCS		15.53	22.79	32.71
		BCS		18.33	20.99	23.22
Cameraman (256×256)	Entire image	ABCS	18.04	21.09	25.95	29.59
		BCS		22.41	24.24	25.70
	PLAIN	ABCS		29.07	28.97	29.95
		BCS		28.27	30.28	32.19
	EDGE	ABCS		17.69	25.24	28.40
		BCS		18.63	20.95	21.75
	TEXTURE	ABCS		14.62	20.06	29.00
		BCS		16.94	17.92	20.18
Barbara (512×512)	Entire image	ABCS	20.12	22.50	24.88	28.14
		BCS		23.39	24.26	25.31
	PLAIN	ABCS		25.14	27.53	28.22
		BCS		25.11	28.71	30.37
	EDGE	ABCS		20.14	26.51	29.85
		BCS		20.78	24.13	28.56
	TEXTURE	ABCS		14.58	19.55	27.17
		BCS		17.38	18.39	19.83

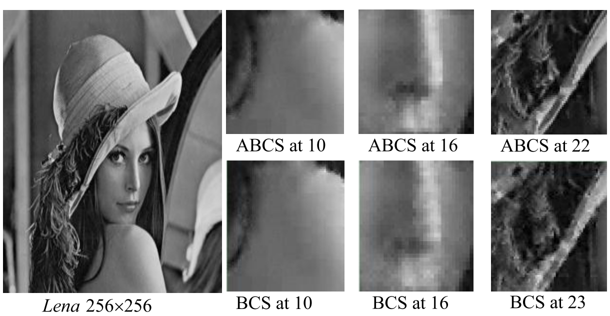

Fig. 4 Visual comparisons between BCS and ABCS for *Lena*: three portions are shown at various stages.

required during the feedback link; thus we have added one more measurement to the fixed-rate BCS scheme. Additionally, when compared to the BCS scheme, the extra complexity happens at block classification at the decoder. However, the computation consumed in this process is ignorable relative to CS reconstruction, but delivering a higher performance.

5. Conclusions

This paper proposed an adaptive block-wise compressive sensing (ABCS) technique that is driven by human visual perception. One unique feature is that our visual analysis is carried out using the DCT coefficients of each block reconstructed at the decoder side; and the analysis result is sent back to the CS encoder (assuming that an up-link channel is available – which is true in most applications) to help choose a new CS-sampling rate adaptively. Our proposed technique keeps the simplicity of the CS coding, takes into account the human eyes' sensitivities to different block types, and offers a progressive reconstruction. When compared with the BCS method at a fixed sampling rate, our ABCS scheme has achieved a remarkable improvement.

Acknowledgments

This work was supported in part by Sino-Singapore JRP (2010DFA11010), NSFC (No.61073142, No.61272262, No.61210006, No.61272051), DSF-TYUST(No.20092011), ICP-SP (No.2011081055), FLDS-SP (No.20111022), Shanxi province Talent Introduction and Development Fund (2011), NSF-SP (2012011014-3).

References

[1] D.L. Donoho, "Compressed sensing," IEEE Trans. Inf. Theory, vol.52, pp.1289–1306, July 2006.

[2] E. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" IEEE Trans. Inf. Theory, vol.52, no.12, pp.5406–5425, Dec. 2006.

[3] E.J. Candes and M.B. Wakin, "An introduction to compressive sampling," IEEE Signal Process. Mag., vol.25, no.2, pp.21–30, March 2008.

[4] M.A.T. Figueiredo, R.D. Nowak, and S.J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," IEEE J. Sel. Top. Signal Process., vol.1, no.4, pp.586–597, Dec. 2007.

[5] T.T. Do, L. Gan, N. Nguyen, and T.D. Tran, "Sparsity adaptive matching pursuit algorithm for practical compressed sensing," Asilomar Conf. on Signals, Systems, and Computers, Pacific Grove, California, pp.581–587, Oct. 2008.

[6] J. Haupt and R. Nowak, "Signal reconstruction from noisy random projections," IEEE Trans. Inf. Theory, vol.52, no.9, pp.4036–4048, Sept. 2006.

[7] L. Gan, "Block compressed sensing of natural images," Proc. Int. Conf. on Digital Signal Process., pp.403–406, Cardiff, UK, July 2007.

[8] S. Mun and J.E. Fowler, "Block compressed sensing of images using directional transforms," Proc. Int. Conf. on Image Process., pp.3021–3024, Cairo, Egypt, Nov. 2009.

[9] A. Wang, L. Liu, B. Zeng, and H. Bai, "Progressive image coding based on an adaptive block compressed sensing," IEICE Electronics Express, vol.8, no.8, pp.575–581, April 2011.

[10] H.H.Y. Tong and A.N. Venetsanopoulos, "A perceptual model for JPEG applications based on block classification, texture masking, and luminance masking," Proc. Int. Conf. on Image Process., pp.428–432, Chicago, IL, USA, March 1998.