RESEARCH ON HIGH-SPPED IMAGE RECONSTRUCTION BASED ON COMPRESSED SENSING

著者	Xu Jiayao
出版者	法政大学大学院理工学・工学研究科
journal or publication title	法政大学大学院紀要. 理工学・工学研究科編
volume	62
page range	1-6
year	2021-03-24
URL	http://doi.org/10.15002/00023948

RESEARCH ON HIGH-SPPED IMAGE RECONSTRUCTION BASED ON COMPRESSED SENSING

Jiayao Xu
Applied Informatics Major, Graduate School of Science and Engineering,
Hosei University,
Supervisor: Jinjia Zhou

***Abstract*— Compressed sensing (CS), as a signal processing technique, is often used to acquire and reconstruct a sparse signal. It can decrease the difficulty of acquiring signal while increase the difficulty of reconstructing the signal. Recently, block-based intra-prediction algorithms are widely used to further increase the compression ratio of images by using the information of neighboring blocks to predict the current block. However, it is hard to increase the speed by parallel processing due to the dependency among the blocks. Meanwhile, the reconstruction of compressed sensing images is time consuming. A reconstruction algorithm using Zigzag ordering-based parallelism is proposed in this paper to solve these problems. Besides, based on the feature of the chosen sensing matrix, a new method with higher efficiency for choosing the first candidate list in the reconstruction procedure was presented in this paper. The experimental results demonstrated that the proposed algorithm speedups the baseline algorithm for 3.26 to 7.13 times. And the quality of the reconstructed images is not changed. Thus, it is a promising solution for fast reconstruction of compressed images.**

Keywords: Block Compressed Sensing; Parallel Reconstruction; Zigzag Scanning

I. INTRODUCTION

Around 2006, the D Donoho, E Candes, T Tao and other scientists proposed a theory named compressed sensing (CS) [1, 2], which proves that the signal is possible to be reconstructed with even less samples than the sampling theorem requires. The concept of CS is if signal X of N length is k-sparse on a basis or dictionary Ψ, then an accurate reconstruction result is obtained by M times ($k < M \ll N$) linear projection this signal to another basis Φ which is incoherence to Ψ. That is:

$$Y = \Phi_{M*N} X_{N*1} = \Phi_{M*N} \Psi_{N*N} \theta_{N*1} = A_{M*N} \theta_{N*1} \qquad (1)$$

In general, the basis Ψ is called as transform matrix, basis Φ is called as measurement matrix [3], matrix A is called as sensing matrix or sampling matrix, M/N is the sampling rate, Y is the compressed signal is called measurement.

The reconstruction procedure of the greedy algorithms is to find the column vectors of the sensing matrix which can describe the signal as close to the original signal as possible. Fast Sparsity Adaptive Matching Pursuit (FSAMP) [4] is one of these algorithms. This algorithm does not require the information of the sparsity, which makes this algorithm more practical than others. Also, this algorithm needs less iteration times due to the strategy of choosing step size in each iteration. The reconstruction procedure of this algorithm chooses the column vectors twice. The first choice called Preliminary Test is to choose the candidate list by using the dot product which is multiplying the sensing matrix with the residual of the last step. The candidate list contains the vectors which get the top big dot product results. The other choice called Final Test keeps the vectors that obtain high results using the least square solution of the candidate.

There is a trade-off between the quality and the time cost of the reconstruction result in this algorithm. Which means, higher quality higher time cost. To solve this problem, this work proposes the Revised Candidate List (RCL) based on the characteristic of the chosen sensing matrix to decrease the complexity of the FSAMP algorithm. This strategy can be used in the greedy algorithms which has the procedure of dot product.

Usually, to increase the compression ratio, the Block-based intra-prediction [5–8] which uses the correlation between the neighboring blocks of the current block, is adopted. The Fig.1 illustrates the details of the block-based intra-prediction refers the information of the upper block and the left block.

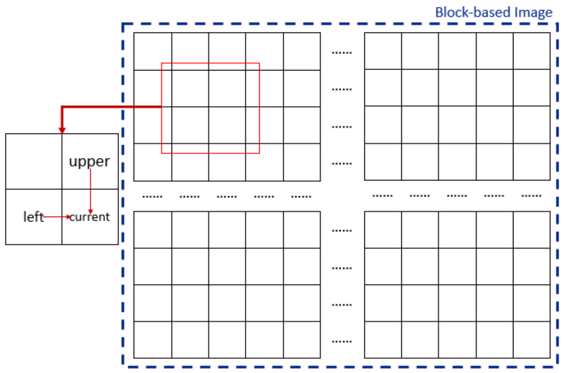


Fig 1. Block-based intra-prediction

The basic of the implement of the intra-prediction is the calculations between the blocks. To improve the practicality of the reconstruction framework and lay the foundation for future work in the compression part, when using the multi-thread method, it is necessary to guarantee the calculations between blocks. Also, the Block Compressed Sensing [9] should be used. However, because of the calculations between the blocks, it is hard to implement the multi-thread. To solve this problem, a method called High-Speed Compressed Sensing Reconstruction using Zigzag based Parallel Processing is proposed. This framework has good generality and can be used in combination with all of the reconstruction algorithms [10] in compressed sensing [11–15].

II. HIGH-SPEED RECONSTRUCTION FRAMEWORK

A. *Revised Candidate List*

The Hadamard matrix and the Fast Walsh-Hadamard transform matrix are chosen in this work, both matrices are hardware friendly. The sensing matrix which obtained by multiplying the Hadamard matrix with the Fast Walsh-Hadamard transform matrix is illustrated in Fig.2. The N is set as 16, M is 4.

16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0

Fig 2. The sensing matrix with N=16, M = 4

As mentioned before, to reconstruct the original signal, it is necessary to find the vectors in the sensing matrix which gets the top result in both first choice and second choice in the FSAMP algorithm. In Fig.2, it is easy to find that when the index of the column vectors is bigger than M, the vectors of the sensing matrix are all zero vectors. Those vectors have no contribution in the reconstruction. Therefore, choosing the first M vectors in the sensing matrix as the revised candidate list, then using the Final Test to gets the reconstructed signal. The other procedures of the reconstruction algorithm follow the FSAMP.

Based on this strategy, the reconstruction efficiency is higher than before, and the quality of the images is same as before.

B. *Parallel framework*

To ensure the implementation of the multi-thread not influence the calculations between the blocks, there are three frameworks that can be taken in consideration.

1. Row-based Parallel

A simple and direct way to reconstruct the image parallelly is using row-based Parallel. This parallel method is setting the row of the blocks as a unit, then putting one unit into one thread to reconstruct the blocks parallelly. To describe it clearly, the details of the row-based parallel is demonstrated in Fig.3.

There are J rows of the blocks reconstructed parallelly. In other words, when reconstructing the current block, it is not guaranteed that the upper block is reconstructed but it is certain that the reconstruction of the left block is finished. Thus, when reconstructing the image blocks, the information of the upper block can not be used. Which means, the compression ratio of the row-based parallel is lower than the compression ratio of the original.

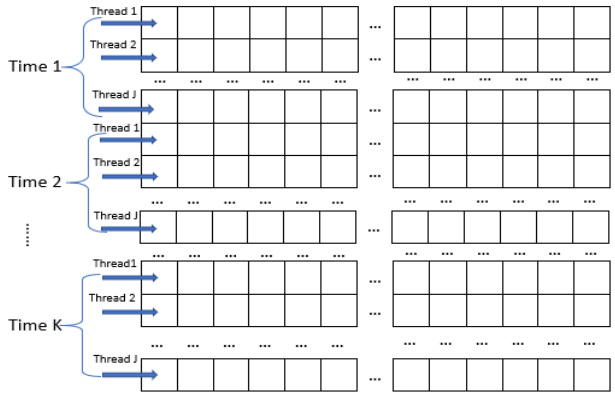


Fig 3. Row-based parallel; J is the number of the threads.

2. Group-based Parallel

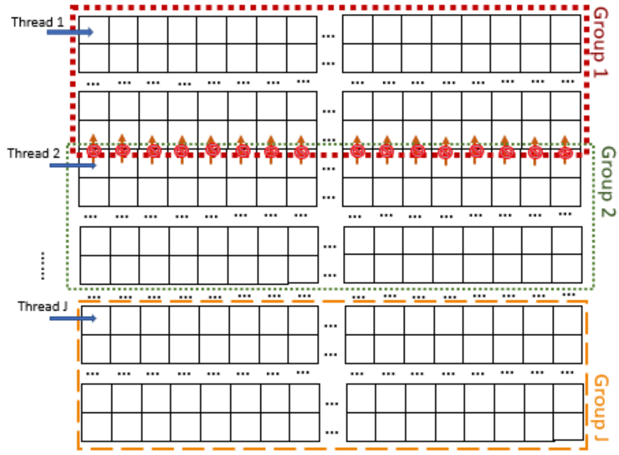


Fig 4. Group-based Parallel; J is the number of the threads.

To increase the compression ratio, there is a parallel method based on group. By this method, the blocks of the images are divided into J groups. One thread reconstructs one group. By this way, the first row of the group cannot refer the information of the upper row. This also lower the compression ratio. The group-based parallel is illustrated in Fig.4.

Because the scan logic of the groups is "top to bottom, left to right", therefore, it is necessary to store the blocks data of the upper row and the data of left block in the same group. Then the storage cost is as the Eq.2 shows.

$$Storage = J_{group} * (Width_{image} + 1) * M * Type \quad (2)$$

The size of the $Type$ depends on the chosen type of data.

In one word, group-based parallel increases the compression ratio than row-based parallel while costs storage. More groups, higher efficiency, lower compression ratio. This illustrates that there is a trade-off

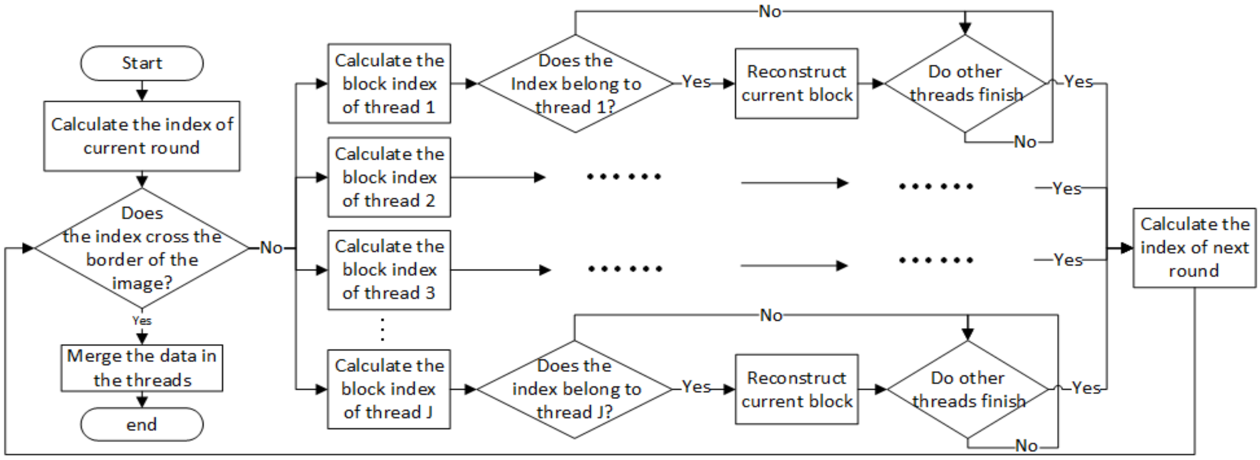


Fig 6. The procedure of parallel reconstruction in our work.

between the compression ratio with the storage when reconstructing the image parallelly. Zigzag Ordering based Parallel is proposed to balance this trade-off.

3. Zigzag Ordering based Parallel (ZOP)

To keep the compression ratio as the original in the same time to save storage, Zigzag scanning based parallel algorithm is proposed. In this algorithm, the scan logic of "top to bottom, left to right" is replaced by "Zigzag scanning". The Zigzag scanning ensures the left and the upper block of the current block must be reconstructed before reconstructing the current block. Combining with the parallel processing, the Zigzag Scanning based Parallel is demonstrated in Fig.5.

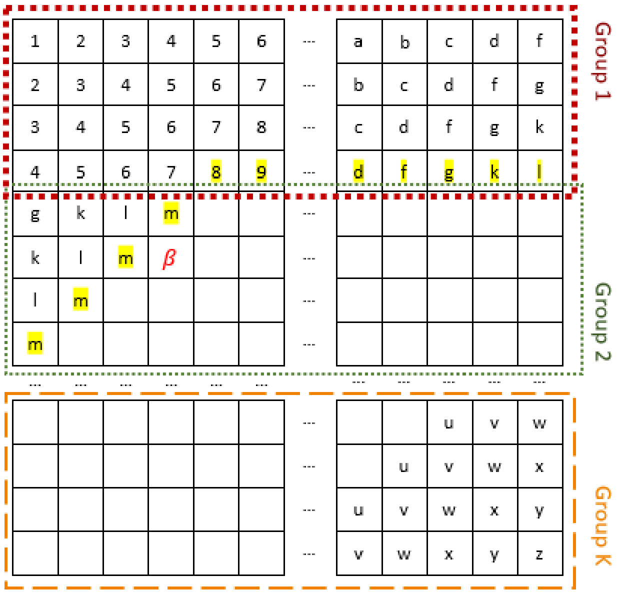


Fig 5. Zigzag scanning based parallel; The image is divided into K groups; There are 4 rows of blocks in each group; The thread number is 4; The blocks have same number or same character (except β) run simultaneously; The yellow highlight means the stored blocks when reconstructing the block β

As the Fig.5 shows, when the reconstruction procedure is at the last 3 rounds of the group, meanwhile, the current group is not the last group of the image, the blocks of the first 3 rounds of the next group are putted into the threads. In this way, only the rounds at the first 3 round and the last 3 rounds of the image, the number of the running threads is less than 4. The more threads used, the more rounds that have idling threads. Which means, the speedup of this part should not higher than the thread number.

Because this algorithm reconstructs the image one group by one group, and the first row of each group needs the information of the upper block in the compression procedure, it is necessary to store the last row of blocks in the group. During the reconstruction procedure, the left block of the current block in each thread should also be stored. Thus, the storage cost during the reconstruction is as the Eq.3 shows:

$$Storage = (Width_{image} + J - Current_x) * M * Type \quad (3)$$

$Current_x$ is the x index of the current block, J is the thread number.

For example, when reconstructing the block β, the details of the positions of the stored blocks are shown in the Fig.5 marked by the yellow highlight. Setting the width of the image as 1024, setting the J_{group} and J as 4, the storage costed by Eq.2 is $4100 * \mathrm{M} * Type$, the storage costed by Eq.3 is $1024 * \mathrm{M} * Type$.

To sum up, the Zigzag scanning based parallel algorithm saves the storage and keeps the compression ratio.

C. The overall framework

The overall framework of Zigzag ordering based parallel using J threads is as Fig.6 shows.

To reconstruct the block parallelly, the first step is calculating the index, then check the value of index to see whether it's out of the image border or not. If yes, merging the data in the threads then quit the procedure, otherwise, calculating the index in every thread. Then reconstructing the blocks in the threads based on the index obtained in the last step. If the reconstruction in the current thread is finished, the thread will be blocked until the others are all finished. After all threads are finished, the loop will continue and run next round till whole image processed.

III. EXPERIMENTS

Experimental simulations are performed by Intel® Core^{TM} i7-7700 CPU @ 3.60 GHz with 4 cores, RAM 16 G, MATLAB R2019a. The block size is 16*16. The thread number is set as 4 in terms of the core number. The sampling rates (SR) are 0.2, 0.4, 0.6, 0.8. The chosen databases are Urban 100 [16], BSD 500 [17], Sun-Hays 80 [18] and the typical images named Aerial, Airplane, Boat, Couple, House, Lena, Mandrill, Peppers, Sailboat, Splash. Because of the limitation of the block size, both the width and the height of the image should be multiples of 16. Therefore, 200 images with the size of 320*480, 10 images with the size of 512*512, 1024*672, 1024*768 as well as 1024*1024 are chosen. The details of the dataset for experiment of this work is presented in Tab.1.

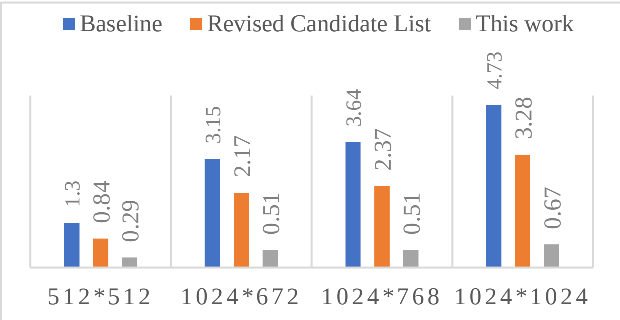
Table 1. Dataset details

Size of the images	Number of the images
320*480	200
512*512	10
1024*672	44
1024*768	54
1024*1024	10

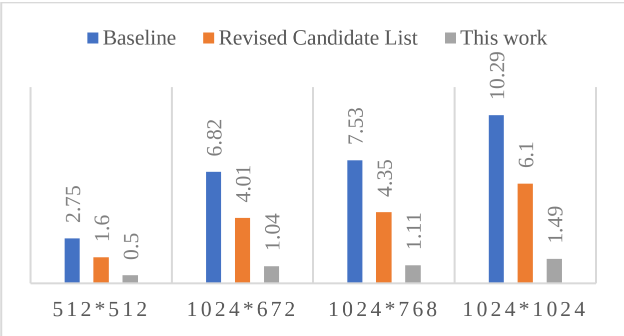
Table 2. Reconstruction Time (RT) comparison between different Sampling Rates (RT); The Baseline is FSAMP

RT(s)	Sampling Rate			
	0.2	0.4	0.6	0.8
Baseline	0.75	1.62	3.31	6.35
+RCL	0.48	0.88	1.77	3.39
Speedup of RCL	×1.56	×1.84	×1.87	×1.87
+RCL+ZOP	0.23	0.34	0.66	1.31
Speedup of ZOP	×2.09	×2.59	×2.68	×2.59
Total Speedup	×3.26	×4.76	×5.02	×4.85

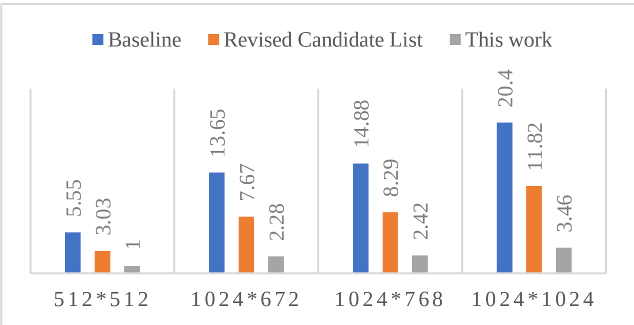
The Tab.2 explicates the average reconstruction time of the test images with the size of 320*480. The speedup of RCL is the reconstruction time of the baseline divided by the reconstruction time of using the revised candidate list. The speedup of ZOP is the reconstruction time of using the revised candidate list divided by the reconstruction time of this work The total speedup is the reconstruction time of the baseline divided by the reconstruction time of this work which combines the zigzag ordering based parallel with the revised candidate list. In the Table.2, it is easy to find that using RCL, the reconstruction time almost saves half of the original. Based on the total speed up, the reconstruction efficiency increases obviously. Comparing the result on sampling rate of 0.6 with the result on sampling rate of 0.8, the speed up results of RCL are same. On the contrary, the speed up results of ZOP are different. This is because there is an idle option for threads during multi-thread reconstruction procedure. If the difference between the least time cost with the most time cost of the thread in the same loop is too large, the speed up result will be worse. In other words, the larger difference on reconstruction time cost of the blocks in the same round, the longer time thread stays idle, the lower reconstruction efficiency. The time cost difference is influenced by the feature of the measurement, and it cannot be predicted before the reconstruction procedure.



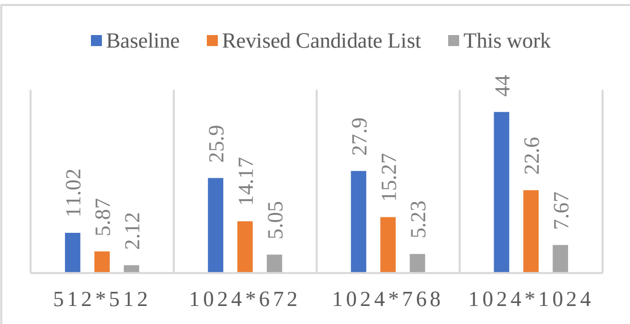
(a)



(b)



(c)



(d)

Fig 7. The results of reconstruction time using different sizes of images in sampling rate of (a) 0.2, (b) 0.4, (c) 0.6, (d) 0.8

The time cost details using different sizes of images are given in the Fig.7.

In Fig.7, when image size is 1024*1024, sampling rate is 0.8, the speed up of RCL is highest. When image size is 1024*1024, sampling rate is 0.2, the speed up of ZOP is highest. As mentioned before, the speedup between this work and revised candidate list should not higher than thread number 4. But in the Fig.7, some speedups are higher than 4, the reason is that if using the multithread processing in the MATLAB, MATLAB will optimize the program itself. Therefore, the multithread processing in MATLAB will increase the efficiency of the reconstruction.

Because the theory of the RCL is selecting the candidate list based on the characteristic of the chosen sensing matrix, the quality of the reconstructed images is not changed. In addition, the ZOP is combining multi-thread with the reconstruction procedure, the quality of the reconstruction is also not changed. In one word, this work does not influence the quality of the reconstruction results. The details of the reconstruction results are illustrated in the Figure.8.

SR = 0.2 SR = 0.4 SR = 0.6 SR = 0.8

PSNR = 23.9 PSNR = 27.6 PSNR = 30.0 PSNR = 34.6

(1) Original

PSNR = 23.9 PSNR = 27.6 PSNR = 30.0 PSNR = 34.6

(2) This work

(a)

SR = 0.2 SR = 0.4 SR = 0.6 SR = 0.8

PSNR = 26.1 PSNR = 31.4 PSNR = 33.8 PSNR = 37.5

(1) Original

PSNR = 26.1 PSNR = 31.4 PSNR = 33.8 PSNR = 37.5

(2) This work

(b)

SR = 0.2 SR = 0.4 SR = 0.6 SR = 0.8

PSNR = 16.9 PSNR = 20.5 PSNR = 23.1 PSNR = 28.0

(1) Original

PSNR = 16.9 PSNR = 20.5 PSNR = 23.1 PSNR = 28.0

(2) This work

(c)

SR = 0.2 SR = 0.4 SR = 0.6 SR = 0.8

PSNR = 21.5 PSNR = 26.6 PSNR = 29.2 PSNR = 33.3

(1) Original

PSNR = 21.5 PSNR = 26.6 PSNR = 29.2 PSNR = 33.3

(2) This work

(d)

SR = 0.2 SR = 0.4 SR = 0.6 SR = 0.8

PSNR = 20.1 PSNR = 22.3 PSNR = 24.4 PSNR = 28.9

(1) Original

PSNR = 20.1 PSNR = 22.3 PSNR = 24.4 PSNR = 28.9

(2) This work

(e)

*Fig 8. The quality of the reconstruction results using different sampling rate (SR) on different size of (a) 320*480, (b) 512*512, (c) 1024*672, (d) 1024*768, (e) 1024*1024*

As the result shows, this work keeps the quality of the reconstruction results and decreases the reconstruction time cost.

V. CONCLUSION

In this paper, the theory of Compressed Sensing is introduced. The key point of the Compressed Sensing is the reconstruction part. One of the reconstruction algorithms is Fast Sparsity Adaptive Matching Pursuit, which not requires the sparsity of the signal in the transform domain, needs fewer iteration times. The Fast Sparsity Adaptive Matching Pursuit is the baseline algorithm of this work.

A new algorithm called High-speed Compressed Sensing Reconstruction using Zigzag Ordering based Parallel Processing is proposed. This algorithm combines the muti-thread method with the reconstruction procedure of the compressed sensing together. Also, this algorithm is friendly to compression ratio by using Zigzag ordering based parallel. Furthermore, this algorithm modifies the baseline algorithm based on the feature of the chosen matrices by using the Revised Candidate List to replace the dot product in the baseline algorithm.

Based on the experimental results, comparing with the baseline, this proposal decreases the reconstruction time obviously and keeps the quality of the reconstructed images. Also, because it is possible to do calculations between the blocks, the compression procedure with intra-prediction can be integrated with this work. And this work doesn't influence on the compression ratio.

REFERENCES

[1] Emmanuel J Candes, Justin Romberg, and Terence Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," IEEE Transactions on information theory, vol. 52, no. 2, pp. 489–509, 2006.

[2] David L Donoho et al., "Compressed sensing," IEEE Transactions on information theory, vol. 52, no. 4, pp. 1289–1306, 2006.

[3] Youness Arjoune, Naima Kaabouch, Hassan El Ghazi, and Ahmed Tamtaoui, "A performance comparison of measurement matrices in compressive sensing," International Journal of Communication Systems, vol. 31, no. 10, pp. e3576, 2018.

[4] Shihong Yao, Qingfeng Guan, Sheng Wang, and Xiao Xie, "Fast sparsity adaptive matching pursuit algorithm for large-scale image reconstruction," EURASIP Journal on Wireless Communications and Networking, vol. 2018, no. 1, pp. 78, 2018.

[5] Jianbin Zhou, Jinjia Zhou, and Li Guo, "Angular intra prediction based measurement coding algorithm for compressively sensed image," in 2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). IEEE, 2018, pp. 1–6.

[6] Jirayu Peetakul, Jinjia Zhou, and Koichi Wada, "A measurement coding system for block-based compressive sensing images by using pixel-domain features," in 2019 Data Compression Conference (DCC). IEEE, 2019, pp. 599–599.

[7] Diana Mandache, Ali Akbari, and Maria Trocan, "Image compressed sensing recovery using intra-block prediction," in 2015 23rd Telecommunications Forum Telfor (TELFOR). IEEE, 2015, pp. 748–751.

[8] Jianbin Zhou, Dajiang Zhou, Li Guo, Yoshimura Takeshi, and Satoshi Goto, "Measurement-domain intra prediction framework for compressively sensed images," in 2017 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2017, pp. 1–4.

[9] Lu Gan, "Block compressed sensing of natural images," in 2007 15th International Conference on Digital Signal Processing, July 2007, pp. 403–406.

[10] Yonina C Eldar and Gitta Kutyniok, Compressed sensing: theory and applications, Cambridge university press, 2012.

[11] Hui Luo, Ning Zhang, and Yuandong Wang, "Modified smoothed projected landweber algorithm for adaptive block compressed sensing image reconstruction," in 2018 International Conference on Audio, Language and Image Processing (ICALIP). IEEE, 2018, pp. 430–434.

[12] Thong T Do, Lu Gan, Nam Nguyen, and Trac D Tran, "Sparsity adaptive matching pursuit algorithm for practical compressed sensing," in 2008 42nd Asilomar Conference on Signals, Systems and Computers. IEEE, 2008, pp. 581–587.

[13] Sungkwang Mun and James E Fowler, "Block compressed sensing of images using directional transforms," in 2009 16th IEEE international conference on image processing (ICIP). IEEE, 2009, pp. 3021–3024.

[14] Scott Shaobing Chen, David L Donoho, and Michael A Saunders, "Atomic decomposition by basis pursuit," SIAM review, vol. 43, no. 1, pp. 129–159, 2001.

[15] Linyu Wang, Xiangjun Yin, and Jianhong Xiang, "Backtracking adaptive matching pursuit reconstruction algorithm based on improved matching criterion," in Proceedings of the 2nd International Conference on Digital Signal Processing, 2018, pp. 22–26.

[16] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja, "Single image super-resolution from transformed selfexemplars," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5197–5206.

[17] David Martin, Charless Fowlkes, Doron Tal, Jitendra Malik, et al., "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," Iccv Vancouver:, 2001.

[18] Libin Sun and James Hays, "Super-resolution from internet-scale scene matching," in 2012 IEEE International Conference on Computational Photography (ICCP). IEEE, 2012, pp. 1–12.