

Accelerated framework for image compression and reconstruction based on compressive sensing

Tasneem M. Yousif¹, Mohamed M. Ahmed²

¹Department of Electronic and Electrical Engineering, Faculty of Engineering, University of Nottingham, Nottingham, United Kingdom

²Faculty of Computers and Information Technology, Future University in Egypt, Cairo, Egypt

Article Info

Article history:

Received Oct 26, 2024

Revised Aug 24, 2025

Accepted Oct 15, 2025

Keywords:

Compression
Compressive
FPGA
OMP
Pipelined
Reconstruction

ABSTRACT

Image compression is a crucial field driven by advancements in communication and imaging technologies. Its primary goal is to achieve low bit rates while maintaining high-quality image reconstruction. Compression is essential in digital image processing, multimedia applications, and medical imaging. Various algorithms exist for image compression and reconstruction, each differing in efficiency. Compressive sensing (CS) algorithms, commonly used for radar data reconstruction, require iterative computations that demand significant processing power and time, limiting real-time applications. To overcome these challenges, this study proposes a parallel-pipelined processing approach to enhance compression and reconstruction efficiency. The method accelerates processing speeds, increases data throughput, and optimizes performance by reducing data size. The proposed approach divides image data into multiple parallel processing branches, significantly reducing computational cycles. This results in faster execution and improved real-time applicability. MATLAB simulations and field-programmable gate array (FPGA) hardware implementations have been conducted to validate the system's effectiveness. The results demonstrate that the parallel-pipelined method significantly enhances efficiency compared to traditional approaches, making it suitable for applications requiring high-speed image processing, such as satellite imaging and medical diagnostics.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Tasneem M. Yousif

Department of Electronic and Electrical Engineering, Faculty of Engineering, University of Nottingham
Nottingham, United Kingdom

Email: tasneem.yousif@nottingham.ac.uk

1. INTRODUCTION

Advancements in media technologies and the growing demand for efficient data handling have made image compression a critical research area. The primary goal is to reduce storage requirements and enhance data transmission rates while maintaining high visual fidelity [1]. Various methods, including wavelet-based algorithms, are widely used for compression in applications such as HDTV, satellite communications, internet teleconferencing, and digital image storage. Without effective compression, these technologies would struggle to meet performance expectations [2].

Classical compression algorithms, though effective in personal computing, are computationally intensive and unsuitable for real-time applications or systems with limited resources. To address these challenges, compressive sensing (CS) techniques have emerged, initially applied in radar signal reconstruction, offering advantages like reduced computational complexity and power consumption [3].

CS techniques show promise for image reconstruction but face computational burdens due to iterative algorithms. As the complexity of the data increases, so does the processing time. To reduce this, the study proposes a parallel-branch pipelined processing approach for the orthogonal matching pursuit (OMP) algorithm, aiming to accelerate compression and reconstruction processes. This method improves system efficiency by reducing computational cycles, offering potential performance improvements over traditional approaches [4]. This paper is organized as follows: section 2 provides an overview of the OMP algorithm. Section 3 describes the proposed methodology in detail. Section 4 discusses the test hardware configuration employed for evaluation, and section 5 presents the simulation results. Finally, section 6 concludes the study, summarizing the key contributions and implications of the proposed method.

2. THE ORTHOGONAL MATCHING PURSUIT

OMP is a greedy iterative algorithm used in CS to recover sparse signals from a small number of measurements [5]. The process begins with a compression step. Suppose we have a high-dimensional signal represented as a sparse vector $X \in \mathbb{R}^N$, where most elements are zero. To reduce dimensionality, we multiply this signal with a sensing matrix $A \in \mathbb{R}^{M \times N}$ where $M \ll N$, resulting in a measurement vector $Y \in \mathbb{R}^M$. This relationship is expressed as:

$$Y = AX \quad (1)$$

this equation represents the compression process, where the sparse signal X is transformed into a lower-dimensional measurement vector Y . The goal is then to reconstruct X from Y and A , even though Y contains much less information than the original signal [6].

The reconstruction step is where OMP comes into play. OMP starts with an initial residual $r^{(0)} = Y$ and an empty index set $\Lambda^{(0)} = \emptyset$ [7]. In each iteration, the algorithm identifies the column of A that is most correlated with the current residual. This is done by computing the inner product between each column a_j of A and the residual vector $r^{(t)}$, and selecting the index λ_t with the maximum absolute value [8]:

$$\lambda_t = \arg \max_j | \langle a_j, r^{(t)} \rangle | \quad (2)$$

the selected index is then added to the support set: $\Lambda^{(t+1)} = \Lambda^{(t)} \cup \{\lambda_t\}$. Next, a least squares problem is solved over the current support to estimate the non-zero coefficients of X . The solution is:

$$\hat{X}^{(t+1)} = \arg \min_z \|Y - A_{\Lambda^{(t+1)}} z\|_2^2 \quad (3)$$

here, $A_{\Lambda^{(t+1)}}$ is a submatrix of A containing only the columns indexed by $\Lambda^{(t+1)}$. Once the estimate is found, the residual is updated as (4).

$$r^{(t+1)} = Y - A_{\Lambda^{(t+1)}} \hat{X}^{(t+1)} \quad (4)$$

This residual reflects the part of Y that is still unexplained by the current solution. The algorithm continues iterating by selecting a new column, updating the support set, solving the least squares problem, and updating the residual, until a stopping condition is met [9]. This can be a fixed number of iterations K , corresponding to the sparsity level of X , or when the residual norm $\|r^{(t+1)}\|_2$ becomes sufficiently small [10]. OMP provides an efficient way to reconstruct a sparse signal from limited measurements by greedily selecting the most significant components in each iteration. The key advantage is that, despite the reduced number of measurements, the sparse signal X can still be accurately reconstructed by exploiting its sparsity and the properties of the sensing matrix A .

OMP exploits the sparsity property of signals to reduce the number of required measurements, thereby enabling the design of devices with lower size, weight, power consumption, and manufacturing costs [11]. As established by Donoho and Candès in 2004, a signal that is sparse in some domain can be accurately reconstructed from a minimal number of linear, non-adaptive observations. This foundational principle forms the basis of OMP and its application in CS [12]. The OMP framework transforms a high-dimensional signal S into a lower-dimensional measurement vector Y , leveraging the sparse representation X and a sensing matrix A [13]. Mathematically, this is expressed as $Y = AX$, where $A \in \mathbb{R}^{M \times N}$, with $M \ll N$, and X is a sparse vector. Due to the properties of the sensing matrix and the inherent sparsity of the signal, the original high-dimensional signal can be reconstructed with high probability [14], [15]. OMP is primarily utilized for signal reconstruction and has found applications in image compression and recovery. As illustrated in Figure 1, the

original image is represented by a set of numerical coefficients X . The compression process involves the multiplication of X with the sensing matrix A to obtain the measurement vector Y , such that $Y=AX$. The dimensions of Y correspond to the number of rows in A , thereby resulting in a lower-dimensional representation of the original signal [16]. This operation constitutes the compression stage of the process. The more complex component of the procedure lies in the reconstruction stage [17], [18]. While compression requires only a straightforward matrix multiplication, reconstruction through OMP involves multiple iterative steps. The OMP algorithm functions as an estimator that operates iteratively to recover the original sparse signal. In each iteration, the algorithm identifies the column in A that has the highest correlation with the current residual vector r . Initially, the residual $r^{(0)}$ is set equal to Y , and is subsequently updated in each iteration to reflect the portion of the signal that remains unexplained [19], [20]. This iterative process continues by selecting the best-matching column from A , updating the support set of selected indices, solving a least squares problem to estimate the sparse coefficients, and updating the residual accordingly [21], [22]. The process repeats until a predefined stopping criterion is met, such as reaching the expected sparsity level or achieving a sufficiently small residual norm [23]-[25].

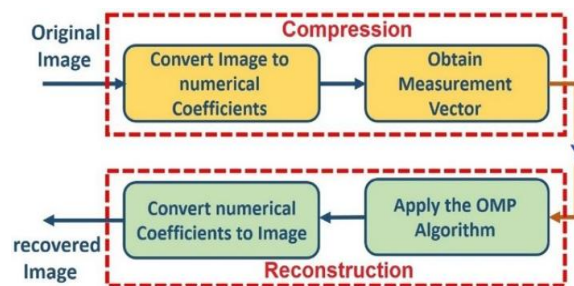


Figure 1. The process of the compression and the reconstruction

3. PROPOSED METHOD

Ma *et al.* [4], introduced pipelined processing as a strategy to enhance computational efficiency, replacing the conventional single-cycle processing paradigm. As illustrated in Figure 2, a comparison between single-cycle and pipelined processing demonstrates the clear performance benefits of the latter. Unlike single-cycle processing, where operations are performed sequentially (one per clock cycle) pipelined processing enables the concurrent execution of multiple operations by dividing them into discrete stages. This approach allows overlapping of operations, thereby significantly reducing the overall execution time and increasing throughput.

The adoption of pipelined processing enhances resource utilization and accelerates data handling, making it especially effective in computationally intensive tasks such as image compression and reconstruction. As demonstrated in Figure 2, pipelined architectures significantly reduce the number of clock cycles required, increasing throughput. However, this gain comes at the cost of increased hardware complexity due to the additional logic needed to manage parallel stages. In this study, 64×256 images (16,384 pixels) are compressed and reconstructed using a block-based method. Each image is divided into 64 blocks of 256 elements, and processing is performed independently for each block. A single-cycle processing approach would require 16,384 clock cycles to complete this task, while a pipelined design reduces it to 319 cycles a substantial performance improvement.

To further enhance processing speed for time-critical applications, a dual-branch parallel pipelined architecture is proposed. In this approach, the input image is split into two equal sections of size 32×256 , and each is processed independently using pipelined techniques. Each branch uses a 256×150 sensing matrix for compression, resulting in 32 vectors of 150 elements each. These vectors serve as compressed representations of the image, allowing for efficient storage and transmission. During reconstruction, the same sensing matrix is used to recover the original image. This parallel-pipelined architecture combines the speed of pipelining with the concurrency of parallel processing, further reducing computation time without sacrificing accuracy. Figure 3 illustrates the structure and operational flow of this enhanced system. The primary goal of this research is to demonstrate efficient parallel processing for 64×256 image compression and reconstruction. A systematic methodology is followed, including segmentation of the input matrix, concurrent processing with sensing matrices, and recovery of the original image. The proposed approach is evaluated through comparative analysis against existing methods. Simulation results highlight its superior performance in both computational efficiency and reconstruction fidelity.

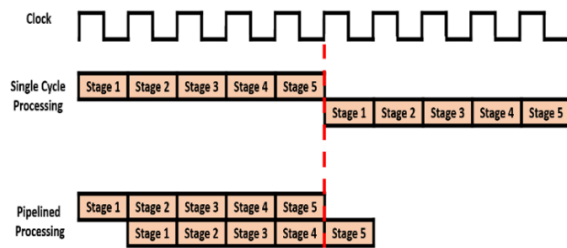


Figure 2. The single-cycle and the pipelined processing

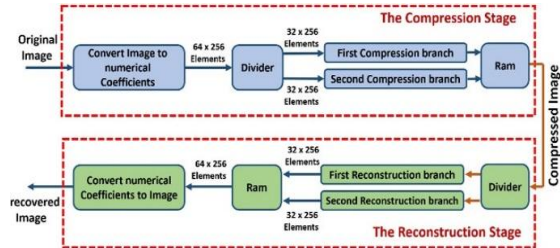


Figure 3. The proposed algorithm

4. HARDWARE IMPLEMENTATION

This section outlines the field-programmable gate array (FPGA) implementation of the proposed algorithm using the Xilinx Spartan 3A/3AN FPGA starter kit (XC3S700A-4C in the VQ100 package). The design is created with custom VHDL code in the Xilinx ISE 14.7 environment, with simulation and verification using ISim. The implementation is designed for ease of assembly and debugging. Figure 4 shows the schematic of the compression framework, highlighting key components and their interconnections within the FPGA platform.

The compression system comprises six modules. The frequency down-conversion module reduces the FPGA clock from 50 MHz to a suitable lower frequency. The serial-to-parallel conversion module uses a shift register to convert 16-bit serial data to parallel format. The divider module splits the image into two 32×256 sections for parallel processing. Two compression modules, each running at 3.125 MHz, process the 16-bit coefficients. Finally, the memory module stores the compressed data for reconstruction. The reconstruction framework mirrors this structure, as shown in Figure 5. It includes a frequency down-conversion module, a serial-to-parallel conversion module for 32-bit coefficients, and a divider module that splits data into two 32×150 sections. Two reconstruction modules restore the original image, and the memory module stores the output. The experimental setup includes a laptop, Spartan 3A/3AN FPGA kit, JTAG cable, and USB-to-serial cable. The laptop connects via JTAG for project download and ChipScope monitoring. The UART protocol transfers image coefficients to the FPGA and sends reconstructed data back. MATLAB handles conversion between image and coefficients. Figure 6 illustrates the complete setup.

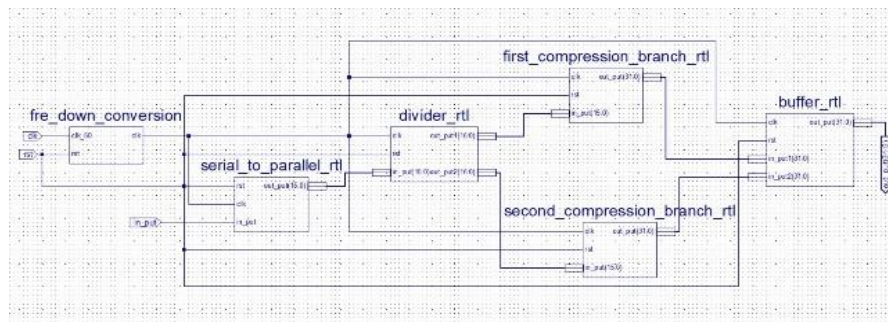


Figure 4. The schematic diagram of the compression

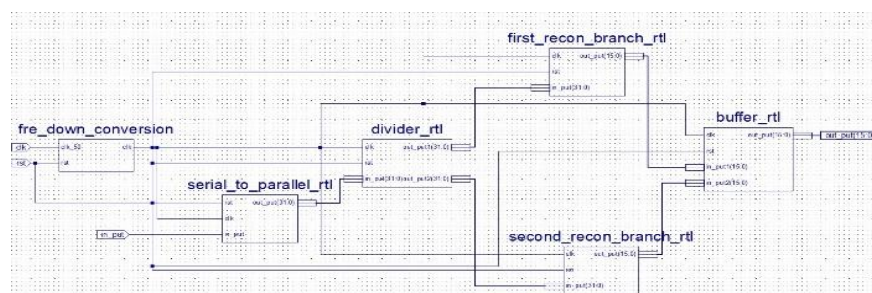


Figure 5. The schematic diagram of the reconstruction

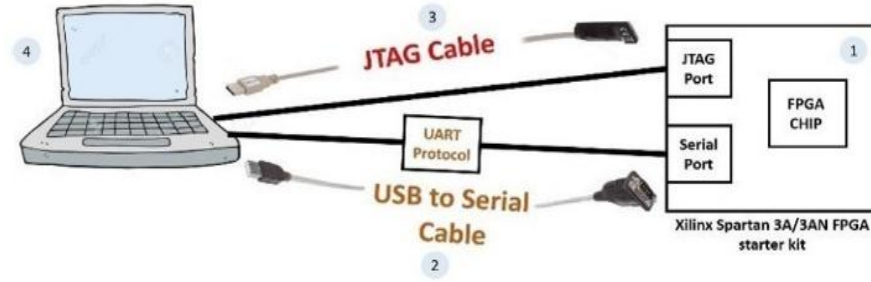


Figure 6. The experimental test preparation

5. SIMULATION AND RESULTS

Ma *et al.* [4] is used as a benchmark, where pipelined processing was introduced to boost performance. The proposed method enhances this by incorporating parallel branches, reducing image processing time from 319 to 287 cycles a 10% improvement. This is crucial for time-sensitive tasks like satellite imaging. For instance, transmitting 100 images would require 1,638,400 cycles using single-cycle processing, 31,900 with pipelining, and only 28,700 with parallel branches saving 3,200 cycles. Though this study uses two branches, the design can be extended further, at the cost of additional hardware. Figure 7 shows peak signal-to-noise ratio (PSNR) values for four compression methods. At a compression ratio (CR) of 0.95 (42% data reduction), PSNR drops below 42 dB. The OMP method consistently performs best, indicating its strength in high-quality image compression. Figure 8 presents mean squared error (MSE) results, showing an inverse relation with CR. Again, OMP achieves the lowest MSE, confirming its superior reconstruction accuracy at higher compression levels.

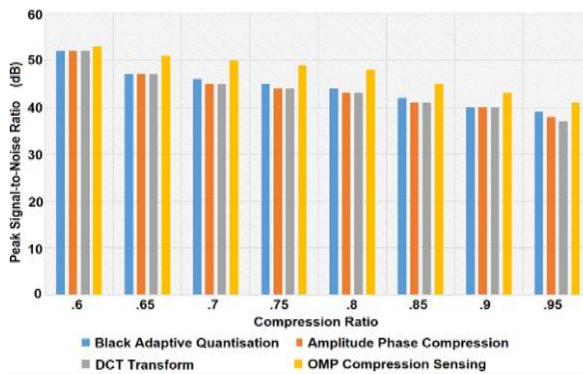


Figure 7. Peak SNR v/s compression ratio

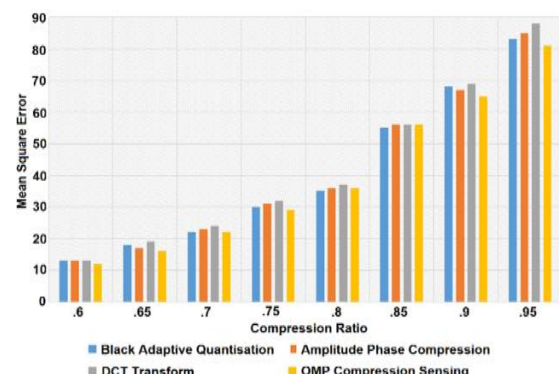


Figure 8. MSE v/s compression ratio

Figures 9 to 13 illustrate the MSE of radar image reconstruction using the OMP algorithm with different under-sampling factors (δ), all evaluated at 50 iterations.

- Figure 9: with $\delta=0.4$, the MSE is 0.015.
- Figure 10: with $\delta=0.5$, the MSE drops to 0.0048.
- Figure 11: with $\delta=0.6$, the MSE further decreases to 0.0042.
- Figure 12: with $\delta=0.7$, the MSE reaches 0.003.
- Figure 13: with $\delta=0.8$, the MSE reaches 0.001.

These results clearly show that as the under-sampling factor increases (i.e., more measurements are taken), the MSE decreases, improving reconstruction accuracy. Figure 14 presents MSE versus sparsity level. As the sparsity level increases (i.e., more non-zero elements or measurements), the MSE continues to decrease. This trend is expected, as more information leads to better reconstruction quality.

Figures 15 to 18 show the signal-to-noise ratio (SNR) versus the number of iterations for different subsampling factors (δ). In all cases, the SNR increases as the number of iterations increases.

- Figure 15: with $\delta=0.5$, the SNR reaches 25 dB at 80 iterations.
- Figure 16: with $\delta=0.6$, the SNR reaches 55 dB at 80 iterations.
- Figure 17: with $\delta=0.7$, the SNR increases to 72 dB at 80 iterations.

– Figure 18: with $\delta=0.8$, the SNR peaks at 110 dB at 80 iterations. These results indicate that a higher subsampling factor (i.e., more measurements) leads to significantly better reconstruction quality, as reflected in the higher SNR values.

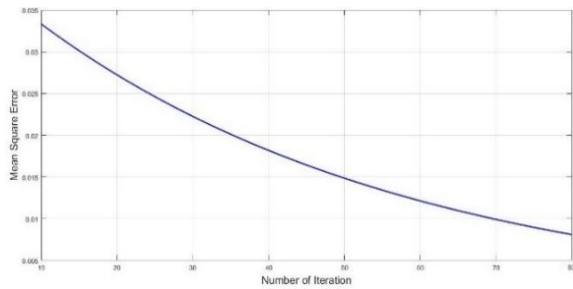
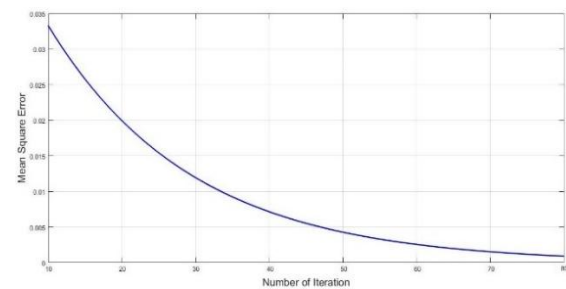
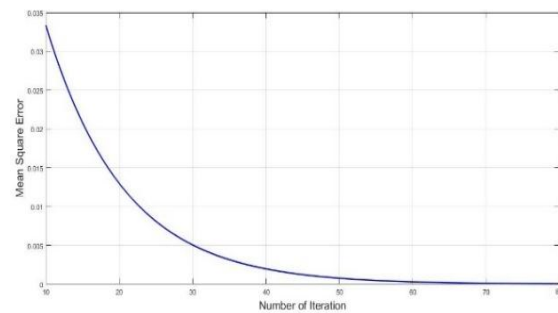
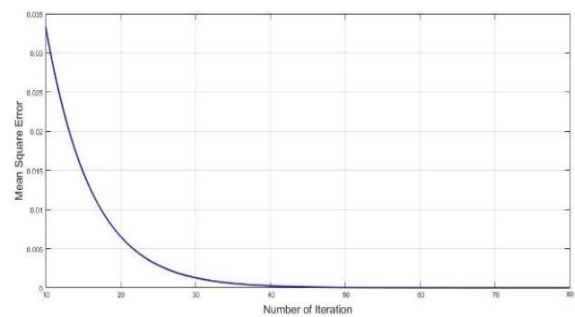
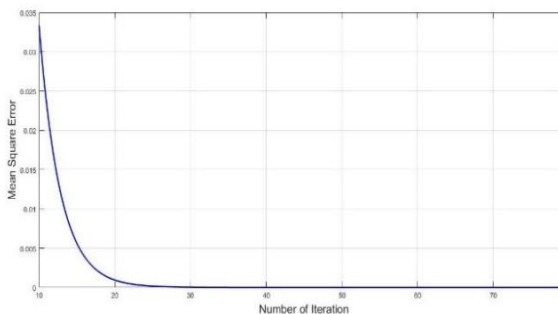
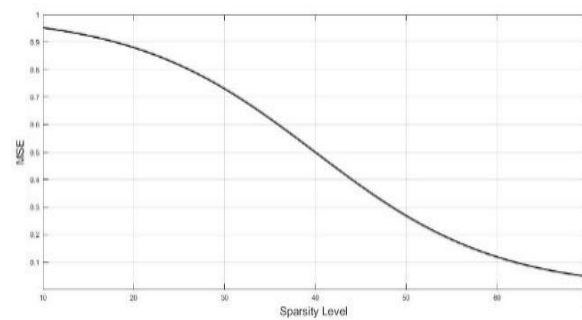
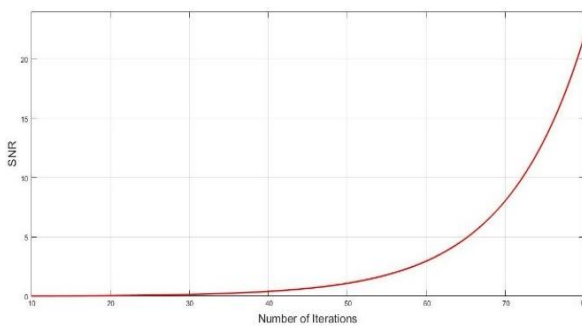
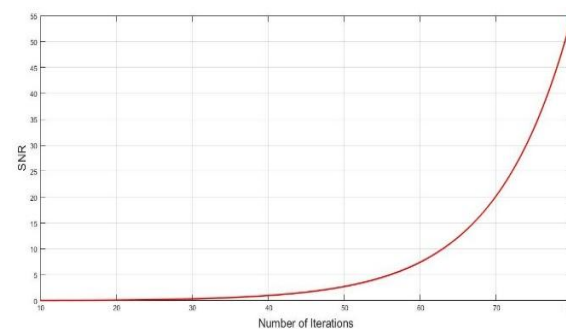
Figure 9. The MSE under-sampling factor $\delta=0.4$ Figure 10. The MSE under-sampling factor $\delta=0.5$ Figure 11. The MSE under-sampling factor $\delta=0.6$ Figure 12. The MSE under-sampling factor $\delta=0.7$ Figure 13. The MSE under-sampling factor $\delta=0.8$ 

Figure 14. The MSE vs the sparsity level

Figure 15. SNR vs no. iterations for $\delta=0.5$ Figure 16. SNR vs no. iterations for $\delta=0.6$

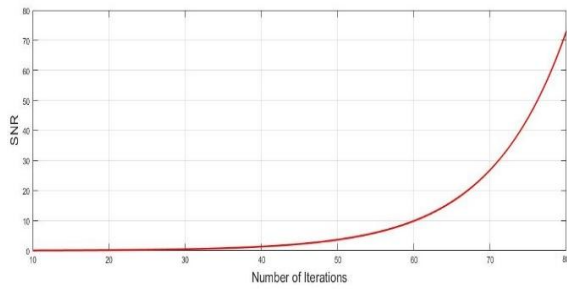


Figure 17. SNR vs no. iterations for $\delta=0.7$

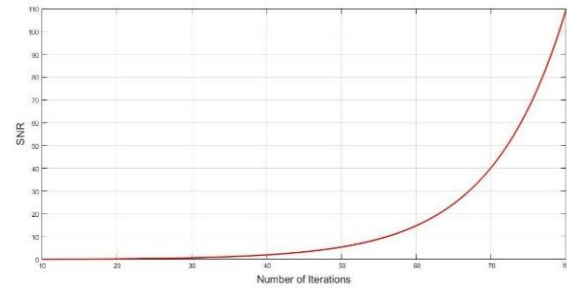


Figure 18. SNR vs no. iterations for $\delta=0.8$

Figure 19 shows the compression simulation using ISim. After compression, a single 32-bit coefficient is visible within the white triangle. The input data is provided in serial form, while the output is delivered in 32 parallel bits. The time interval between the vertical white and yellow lines, highlighted in a circle at the bottom of Figure 19, represents a low-frequency nanosecond-scale timeframe. As indicated, this period is 320 nanoseconds, consistent with the hardware implementation details discussed earlier. Figure 20 presents the image reconstruction simulation using ISim. After reconstruction, two coefficients appear in the white triangle, each represented with 16 bits. Similar to the compression process, the input is serial, while the output is provided in 16 parallel bits. The time interval between the vertical white and yellow lines is marked at the bottom of the Figure 20 and corresponds to 640 nanoseconds, as previously mentioned in the hardware implementation section. Figures 21 and 22 display the ChipScope results for compression and reconstruction, respectively. These results validate the effectiveness of the proposed method and confirm its practical applicability. The output data is presented in both listing and waveform formats, reinforcing the system's functional reliability.

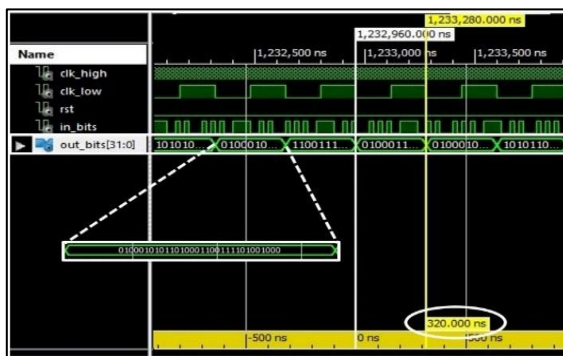


Figure 19. ISim simulation for the compression

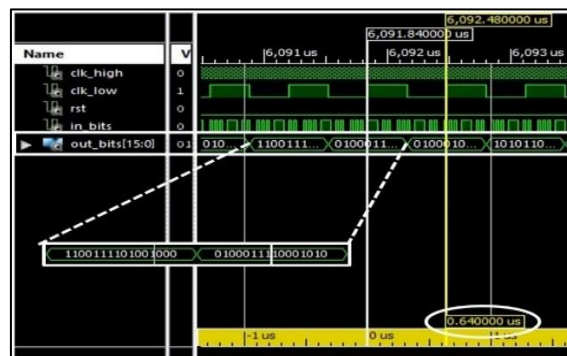


Figure 20. ISim simulation for the reconstruction

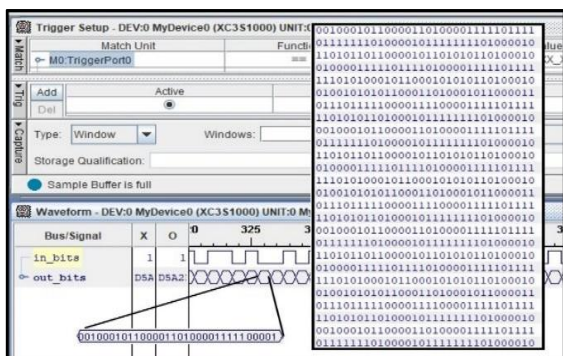


Figure 21. The ChipScope results of the compression

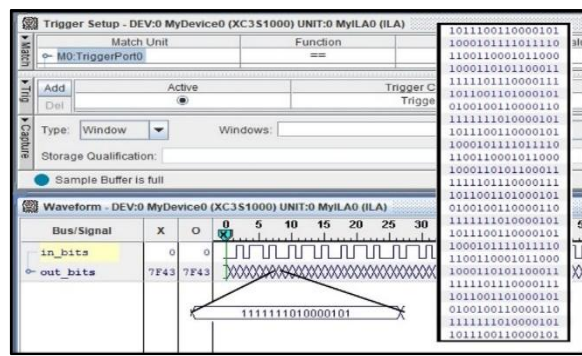


Figure 22. The ChipScope results of the reconstruction

Figures 23 to 26 show the test image before compression and after reconstruction using different numbers of iterations. Slight color variations in the reconstructed images are due to the estimation process. The accuracy of the recovered image depends on the sensing matrix and the number of iterations generally, higher iteration counts lead to better reconstruction quality, up to a certain limit.



Figure 23. The original image

Figure 24. The reconstruction using 20 iterations



Figure 25. The reconstruction using 40 iterations

Figure 26. The reconstruction using 60 iterations

Table 1 presents the device utilization summary for three different processing architectures: single-cycle processing, pipelined processing, and the proposed parallel-branch approach. As illustrated, the parallel-branch method while offering improved performance requires more hardware resources compared to the single-cycle and pipelined designs. This increase in resource consumption is expected, as the parallel-branch architecture involves multiple concurrent processing units to accelerate computation. Despite the higher utilization, the trade-off results in significantly faster execution and better suitability for real-time image compression and reconstruction applications.

Table 1. The device utilization summary

Logic utilization	Single cycle	Pipelined	Parallel branches
Number of slice flip flops	2,015	2,965	3,658
Number of 4 input LUTs	103	189	248
Number of occupied slices	1,965	2,658	3,956
Number of slices containing only related logic	1,403	1,963	2,657
Total number of 4 input LUTs	1,965	2,658	3,956
Number of bonded IOBs	68	76	153

6. CONCLUSION

There are different algorithms for the image compression and the reconstruction. One of these methods is the CS algorithm. This algorithm is an iterative algorithm that needs high computational operations and high numbers of iterations to achieve the function. These iterations lead to high delay and high time consumption. In order to speed up processing, this research suggested a practical parallel-pipelined processing solution for the OMP CS algorithm. According to the aim of this work, the time needed for OMP reconstruction and compression can be decreased. Additionally covered in this article are the design, analysis, and application of the suggested method. The paper succeeded in achieving its goal and reduced the consumed cycles that are used for image compression and image reconstruction by using parallel-pipelined processing branches. The performance is enhanced but the consumed hardware resources will be increased. Finally, to demonstrate the differences between the two images, the test image is shown before compression and after reconstruction. The FPGA implementation and MATLAB simulation are used to confirm the paper's goal.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Tasneem M. Yousif	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Mohamed M. Ahmed	✓	✓	✓		✓		✓	✓		✓			✓	

C : Conceptualization

M : Methodology

So : Software

Va : Validation

Fo : Formal analysis

I : Investigation

R : Resources

D : Data Curation

O : Writing - Original Draft

E : Writing - Review & Editing

Vi : Visualization

Su : Supervision

P : Project administration

Fu : Funding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.




REFERENCES

- [1] A. Bekit, C.-I. Chang, B. Lampe, C. J. D. Porta, and C.-C. Wu, "N-FINDER for finding endmembers in compressively sensed band domain," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 2, pp. 1087–1101, Feb. 2020, doi: 10.1109/TGRS.2019.2943448.
- [2] F. Zhang, X. Liang, R. Cheng, Y. Wan, L. Chen, and Y. Wu, "Building corner reflection in MIMO SAR tomography and compressive sensing-based corner reflection suppression," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 3, pp. 446–450, 2020, doi: 10.1109/LGRS.2019.2926301.
- [3] K. Sekar, K. S. Devi, P. Srinivasan, T. Dheepa, B. Arpita, and L. D. Singh, "Joint correlated compressive sensing based on predictive data recovery in WSNs," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Feb. 2020, pp. 1–5, doi: 10.1109/ic-ETITE47903.2020.181.
- [4] Y. Ma, H. Hong, and X. Zhu, "Multiple moving-target indication for urban sensing using change detection-based compressive sensing," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 3, pp. 416–420, Mar. 2021, doi: 10.1109/LGRS.2020.2977168.
- [5] C. J. D. Porta and C.-I. Chang, "Progressive compressively sensed band processing for hyperspectral classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 3, pp. 2378–2390, Mar. 2021, doi: 10.1109/TGRS.2020.3000873.
- [6] J. Peetukul and J. Zhou, "Temporal redundancy reduction in compressive video sensing by using moving detection and inter-coding," in *2020 Data Compression Conference (DCC)*, Mar. 2020, pp. 387–387, doi: 10.1109/DCC47342.2020.00052.
- [7] J. Zammit and I. J. Wassell, "Adaptive block compressive sensing: toward a real-time and low-complexity implementation," *IEEE Access*, vol. 8, pp. 120999–121013, 2020, doi: 10.1109/ACCESS.2020.3006861.
- [8] H. Yang, C. Chen, S. Chen, F. Xi, and Z. Liu, "Non-common band SAR interferometry via compressive sensing," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 6, pp. 4436–4453, 2020, doi: 10.1109/TGRS.2020.2964701.
- [9] Y. Guo, S. Yang, N. Li, and X. Jiang, "Device-free localization scheme with time-varying gestures using block compressive sensing," *IEEE Access*, vol. 8, pp. 88951–88960, 2020, doi: 10.1109/ACCESS.2020.2993576.
- [10] C. Gan, X. Yan, Y. Wu, and Z. Zhang, "A two-branch convolution residual network for image compressive sensing," *IEEE Access*, vol. 8, pp. 1705–1714, 2020, doi: 10.1109/ACCESS.2019.2961369.
- [11] W. Song and W. Wang, "Compressive sensing based multiuser detector for massive MBM MIMO uplink," *Journal of Systems Engineering and Electronics*, vol. 31, no. 1, pp. 19–27, 2020, doi: 10.21629/JSEE.2020.01.03.
- [12] Y. Shi, X. X. Zhu, and R. Bamler, "Nonlocal compressive sensing-based SAR tomography," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 5, pp. 3015–3024, May 2019, doi: 10.1109/TGRS.2018.2879382.
- [13] Y. Wang, H. Bai, and Y. Zhao, "Image reconstruction from patch compressive sensing measurements," in *2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM)*, Sep. 2018, pp. 1–4, doi: 10.1109/BigMM.2018.8499088.
- [14] H. Sayadi, H. M. Makrani, O. Randive, S. M. P.D., S. Rafatirad, and H. Homayoun, "Customized machine learning-based hardware-assisted malware detection in embedded devices," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, Aug. 2018, pp. 1685–1688, doi: 10.1109/TrustCom/BigDataSE.2018.00251.
- [15] L. Ma, H. Bai, M. Zhang, and Y. Zhao, "Edge-based adaptive sampling for image block compressive sensing," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E99A, no. 11, pp. 2095–2098, 2016, doi: 10.1587/transfun.E99.A.2095.
- [16] M. A. Hadi, S. Alshebeili, K. Jamil, and F. E. A. El-Samie, "Compressive sensing applied to radar systems: an overview," *Signal, Image and Video Processing*, vol. 9, no. S1, pp. 25–39, Dec. 2015, doi: 10.1007/s11760-015-0824-y.




- [17] A. Massa, P. Rocca, and G. Oliveri, "Compressive sensing in electromagnetics - a review," *IEEE Antennas and Propagation Magazine*, vol. 57, no. 1, pp. 224–238, Feb. 2015, doi: 10.1109/MAP.2015.2397092.
- [18] C. A. Metzler, A. Maleki, and R. G. Baraniuk, "From denoising to compressed sensing," *IEEE Transactions on Information Theory*, vol. 62, no. 9, pp. 5117–5144, Sep. 2016, doi: 10.1109/TIT.2016.2556683.
- [19] M. Hossiny, "Real-time implementation of radar signal processing," Cairo, 2014.
- [20] E. Ashraf, A. A. M. Khalaf, and S. M. Hassan, "Real time FPGA implementation of SAR radar reconstruction system based on adaptive OMP compressive sensing," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 20, no. 1, p. 185, Oct. 2020, doi: 10.11591/ijeecs.v20.i1.pp185-196.
- [21] M. M. Ahmed, H. Bedour, and S. M. Hassan, "FPGA implementation of an imagecompression and reconstruction system for the onboard radar using the compressive sensing," in *2019 14th International Conference on Computer Engineering and Systems (ICCES)*, Dec. 2019, pp. 163–168, doi: 10.1109/ICCES48960.2019.9068155.
- [22] M. M. Ahmed, H. Bedour, and S. M. Hassan, "Design and implementation of a multistage image compression and reconstruction system based on the orthogonal matching pursuit using FPGA," in *2019 14th International Conference on Computer Engineering and Systems (ICCES)*, Dec. 2019, pp. 174–179, doi: 10.1109/ICCES48960.2019.9068151.
- [23] A. Matin, B. Dai, Y. Huang, and X. Wang, "Ultrafast imaging with optical encoding and compressive sensing," *Journal of Lightwave Technology*, vol. 37, no. 3, pp. 761–768, Feb. 2019, doi: 10.1109/JLT.2018.2880816.
- [24] L. B. Montefusco, D. Lazzaro, S. Papi, and C. Guerrini, "A fast compressed sensing approach to 3D MR image reconstruction," *IEEE Transactions on Medical Imaging*, vol. 30, no. 5, pp. 1064–1075, May 2011, doi: 10.1109/TMI.2010.2068306.
- [25] D. Liang, B. Liu, J. Wang, and L. Ying, "Accelerating SENSE using compressed sensing," *Magnetic Resonance in Medicine*, vol. 62, no. 6, pp. 1574–1584, Dec. 2009, doi: 10.1002/mrm.22161.

BIOGRAPHIES OF AUTHORS



Tasneem M. Yousif    is a Ph.D. student at the Nottingham Geospatial Institute. She received her M.Sc. degree in electronics and communication engineering at the University of Nottingham in 2019. She holds a bachelor's degree in computer engineering from the University of Bahrain. She has multiple of research papers in reputed journals and conferences. Her research interest areas are digital signal processing, GNSS interference detection and mitigation, space, and reconfigurable computing using FPGA. She can be contacted at email: tasneem.yousif@nottingham.ac.uk.



Mohamed M. Ahmed    holds a bachelor's degree in electrical engineering and is currently working on his Ph.D. degree in digital signal processing (DSP). His academic and professional focus is on signal processing and analysis. He has high FPGA and MATLAB expertise, along with solid experience in system analysis and design through simulation and modeling tools. He can be contacted at email: mohamed.mahran@fue.edu.eg.