Sumer Journal for Pure Science (ISSN: 2790-7031)
SJPS 2025 (June), Vol. 4, No. 1, p.p.: 67-77

The Impact of Parallel Processors on Image Compression: A Review

Oras Nasef Jasim[1]*, Noor Abdulkaadhim Hamad[2], Heba Yousef Ateaa[3], Ahmed Basim Adnan[4]

[1] *General Directorate of Education in Thi-Qar Governorate, Thi-Qar, Nasiriyah, Iraq*
[2] *General Directorate of Education in Thi-Qar Governorate, Thi-Qar, Nasiriyah, Iraq*
[3]*Computer Science, Computer Science and Information Technology, Sumer University, Dhi Qar, Iraq*
[4] *Computer Science, Computer Science and Information Technology, Sumer University, Dhi Qar, Iraq*

* *noor_abd@utq.edu.iq*

Abstract

The rapid growth of digital data on the Internet has led to significant challenges, including high costs for storage and transmission. Image compression is a critical solution to reduce memory requirements and associated expenses. Researchers have focused on improving compression algorithms to enhance performance, speed, and efficiency by leveraging parallelism at the task level. The primary goal of compression is to reduce file sizes for easier storage and transmission while maintaining acceptable image quality. Parallel processors play a vital role in enhancing the performance of image compression algorithms, particularly for processing large datasets and meeting the demands of modern applications. This review examines the impact of parallel processors on improving compression efficiency compared to traditional methods, addressing challenges such as data distribution among processors, image quality, and compression ratios. It explores various techniques, including block-level and multi-layer processing, and evaluates their effects on computational complexity, image quality, and interprocessor communication. By analyzing previous studies, this review provides insights into effective strategies for optimizing image compression using parallel processing, considering factors such as computational efficiency and quality preservation. Future work should focus on developing advanced parallel techniques to further enhance compression performance and address existing limitations.

Keywords: GPU, CPU, CUDA, NVIDIA, Parallel Processing, DCT, JPEG Compression Algorithms.

تأثير المعالجات المتوازية على ضغط الصورة: مراجعة

اوراس نصيف جاسم[1]*, نور عبدالكاظم حامد[2], هبه يوسف عطية[3] ,احمد باسم عدنان[4]

[1]مديرية تربية ذي قار, ذي قار, الناصرية, العراق
[2]مديرية تربية ذي قار, ذي قار, , الناصرية, العراق
[3]علوم الحاسوب,علوم الحاسوب و تكنولوجيا المعلومات, جامعة سومر, ذي قار, العراق
[4]علوم الحاسوب,علوم الحاسوب و تكنولوجيا المعلومات, جامعة سومر, ذي قار, العراق

الخلاصة

أدى النمو السريع للبيانات الرقمية على الإنترنت إلى تحديات كبيرة، بما في ذلك ارتفاع تكاليف التخزين والنقل. يعد ضغط الصور حلاً حاسمًا لتقليل متطلبات الذاكرة والنفقات المرتبطة بها. ركز الباحثون على تحسين خوارزميات الضغط لتحسين الأداء والسرعة والكفاءة من خلال الاستفادة من التوازي على مستوى المهمة. الهدف الأساسي من الضغط هو تقليل أحجام الملفات لتسهيل التخزين والنقل مع الحفاظ على جودة الصورة المقبولة. تلعب المعالجات المتوازية دورًا حيويًا في تحسين أداء خوارزميات ضغط الصور، وخاصة لمعالجة مجموعات البيانات الكبيرة وتلبية متطلبات التطبيقات الحديثة. تدرس هذه المراجعة تأثير المعالجات المتوازية على تحسين كفاءة الضغط مقارنة بالطرق التقليدية، ومعالجة التحديات مثل توزيع البيانات بين المعالجات وجودة الصورة ونسب الضغط. كما تستكشف تقنيات مختلفة، بما في ذلك المعالجة على مستوى الكتلة والمعالجة متعددة الطبقات، وتقيم آثارها على التعقيد الحسابي وجودة الصورة والاتصال بين المعالجات. من خلال تحليل الدراسات السابقة، تقدم هذه المراجعة رؤى حول الاستراتيجيات الفعالة لتحسين ضغط الصور باستخدام المعالجة المتوازية، مع مراعاة عوامل مثل الكفاءة الحسابية والحفاظ على الجودة. ينبغي أن يركز العمل المستقبلي على تطوير تقنيات موازية متقدمة لتعزيز أداء الضغط بشكل أكبر ومعالجة القيود الحالية.

1. Introduction

The compression of images is crucial now due the compression of images is widely used in various fields of the present era, including security Cameras, webcam, area detection cameras, etc. Visual images have to be communicated and dispersed and this calls for techniques that enable the compression of picture files. During the compression it is important to consider the applications which can be useful in practice and cost–effectiveness at the same time without losing the most significant visual data in images [1]. Effective compression methods also help in optimizing storage space by producing smaller file sizes, which is essential to save storage memory and facilitate faster data transfer over communication channels.

The compression methods that have been embraced by today's world are JPEG 2000 and the Web. JPEG 2000 for instance makes uses of wavelet compression, an aspect that makers it to have higher per image compression ratios as well as improved image quality at lower bit rates than the common JPEG format. On the other hand, WebP designed by Google offer the best compression ratio for lossless and lossy file formats, their file sizes are usually smaller thus making them ideal for web use and mobile apps. Another obvious method of doing it is the HEVC also known as High-Efficiency Video coding to compress as images and videos with quite a high efficiency while losing only a negligible amount of data.

However, all of these methods are relatively complicated and still need very large calculations, long processing time, and large spaces for storage. It was thus important to look for a way of dealing with these problems in order to achieve organizational goals. The solution for the above problem was to parallel implementation of the compression algorithm. Thus, the idea of shortening the time used for processing can be set as the goal with considerable time-sparing revealed. Also, the execution of image and data compression algorithms can be enhanced and transmitted with the simplest way, allowing for faster transfer speeds and lower storage requirements, which are essential in today's digital communications landscape [2].

The rest of this paper is structured as follows. Section 2 is dedicated to presenting background information about Parallel Processing. In section 3, we present our formal representation of a comparison of previous studies' works. Section 4 is reserved to show the results of our discussion. In contrast, section 5 draws conclusions and forecasts needed for these algorithms to continue to improve to meet the needs of the accelerating digital age in some future works.

2. Parallel Processing

Parallel processing is the process of executing multiple operations simultaneously. It is a type of computer architecture in which large problems are divided into small tasks, usually related to each other, which can all be processed simultaneously, which is of great importance for efficient computing and computation in various fields, which leads to improving system efficiency, accelerating task solving, and contributing to enhancing productivity by harnessing the power of multiple processors to work together to accomplish the required task [3].

Modern computers, workstations, and mobile devices contain multiple central processing unit (CPU) cores. These cores are independent and can execute different instructions at the same time. Programs that use parallelism to take advantage of multiple cores run faster and allow better use of CPU resources [4].

The use of modern general-purpose computing graphics processing units (GPUs) has presented the option of writing application to run on hundreds of small cores. CUDA is NVIDIA's extension of this parallel architecture to its GPUs, and the API for programmers to write C-based parallel applications. CUDA offers a hardware abstractions called blocks of threads that can share data. These blocks are then mapped to multiple standard processors available with the hardware[5].

In this review , we discuss the hardware and software approaches. We have addressed the use of architecture in modern devices such as CPUs and GPUs to achieve parallel processing. We have also addressed CUDA, the parallel computing platform for graphics processing units. In general, the study has shed the greatest light on the hardware approach to take advantage of the capabilities of graphics processing units with cores for parallel execution, while acknowledging the importance of software, especially the CUDA model. However, the main focus of the review is on hardware-based parallel processing. this review provides new insights into the use of appropriate techniques in image compression using parallel processing and helps in identifying the best strategies based on several factors.

3. Previous Studies

In this review, we explore key literature on image compression utilizing parallel processors, highlighting the significant trends and findings that researchers have uncovered in their studies:

Bheshaj Kumar et al. (2011) This research highlighted the need to implement parallel processors in the image compression process by providing general- purpose parallel algorithms for image encryption and decryption using BTC technique. It works by partitioning of the image to non-overlapping blocks where only the needed block maybe compressed by numerous processors at once. The authors were also devoted to the significant of the parallel approach on the aspects of compression performance and execution time. This study also validates that the proposed parallel algorithms further improved the image compression performance That is the work efficiency achieved it is more than 50 per cent input when using many processors. This proves that parallel approach is not only efficient but also work better than the conventional methods in image compression[6].

Hao Duan et al. (2012) designed and developed a highly optimized parallel JPEG-LS encoder for lossless hyperspectral compression using parallel computing processing platform NVIDIA CUDA. JPEG-LS is characterized by relatively low computational cost and, hence, can be used in many applications in fields where lossless image compression is needed. This implementation utilizes the block parallelism strategy and the several CUDA techniques such as the global memory access, the parallel prefix sum and the asynchronous data transfer in improving the performance of the image compression operations. Self-produced graphics demonstrated the general trends of GPU speedup with reference to AVIRIS block size, and of the actual compression ratio with reference to block size. Maximum speed up was found using 6 by 6 by 64 by 64 pixel block size of AVIRIS images with a total time enhancement of 26.3 times faster than CPU based code. This approach helped to enhance the performance of encoding in particular with rather large data such as hyperspectral images[7].

A (2013) study by Kgotlaetsile Mathews Modieginyane et al. described where and how CUDA programming model can be applied to enhance image compression. This algorithm under discussion was a development of the Loeffler's technique which in actuality relied on the discrete cosine transform (DCT). The computational efficiency of the system was assessed and compared in terms of CPU as well as GPU. In test conditions, the authors used images from the Marco Schmidt benchmark database to check the effectiveness of the method. To compare how well the images were reconstructed the authors utilized the basic measure of quality known as the peak signal-to-noise ratio (PSNR). They were able to document differences in the quality of reconstructed images through the use of GPU compared to the CPU. This study also validated that as the image size rises, the CPU processing time for an image is greater than the GPU for the same image size rating. As the results too suggest the computational efficiency of DCT for image compression is also very good particularly for computation extensive jobs like image processing using NVIDIA GPUs rather than conventional techniques [1].

Chongke Bi1 et al. (2014) offered a parallel data compression method which can effectively address the data compression needs of supercomputers due to their scalability. It was also as an advantageous POD method that is involved in the data compression approach that selects the relevant data's characteristic from huge amounts of data resulting in a reduction in the data size without loss of data. The study used binary load distribution technique, which is nearly identical to the binary swap image generation process. This approach optimizes parallel efficiency for processing the workload by the processors, cutting on inter-processor communication costs in the compression computations. This method guarantees that all processors are fully utilized, which is critical for turning out a high performing supercomputer. The results have shown that the proposed method successfully decreases the data size with low computational cost and small errors, which prove that the proposed method is useful to treat large-scale simulation data as a solution of the proposed method[8].

Umesh B. Kodgule and B. A. Sonkamble in (2015) In this work, an efficient parallel algorithm for handling fractional image compression based on the application of the NVIDIA GPGPU is presented. For reduction of the block comparison, a new feature detection based on the discrete wavelet transform is applied. The proposed DWT method and parallel block classification are tested on six standard test images: Barbara, Ellen, Bot, Cobble, Bridge, and Ariel, each with a size of 512 × 512 pixels. The experiments are performed serially on CPU and in parallel using GPU (NVIDIA GeForce GTX 480) by writing CUDA C programs. Overall, from the experimental results the proposed algorithm was found to be results in the reduction ofencoding time over the traditional search. The average of the speed up ratio obtained is 298.32 pointing to a remarkable efficiency gain from the use of parallelism. It also ranked the quality of the reconstructed images with other methods and identified that the parallel method enhances the efficiency of rate processing and make this technique useful in real-time applications including image retrieval, image filtering, image comparison, satellite image, and radiography image [9].

I. P. Skirnevskiy1 et al. in (2016) I also delivered a paper on the use of parallel processors in strengthening Computational power; a case study focused on denoising algorithms for Digital Image Processing on GPU. The feature detection also simplifies the denoising aspect of CT image segmentation where the authors suggested the use of NLM algorithm despite computationally expensive. The authors also pointed out that parallel computation can make the denoising process faster by running tasks in multiple cores of a computer chip, as well as in multiple CPUs, without requiring sophisticated hardware. Based on the exploration of previous work and methods with the specific application in CT images, the NLM algorithm was selected for its efficiency in denoising. The findings of the study show that the performance gain of the computational burdens can be escalated by a factor of 5.6 to 6 times by offloading operations to the GPU when parallel computing is implemented. This significant acceleration shows that we can utilize GPU resources when it is required to solve many computational problems and improves the performance of tasks associated with digital image processing recognised here as ImageProc[10].

P. S. J. Kumar et al. (2017) offered a paper on the computational complexity of several compression methods and the way in which quantitative increases could be made. They stressed on the performance analysis by putting CPU and GPU in contrast, where both are vital for getting higher efficiency in the image processing tasks. To compare, the study assessed the following still image compression methods: block truncation coding (BTC), discrete cosine transforms (DCT), discrete wavelet transform (DWT), dual-tree complex wavelet transform (DTCWT), segmentation of hierarchical trees (SPIHT), and embedded zero-tree wavelet (EZW). All these methods are different but each of them possesses their strength points and may be improved to demonstrate better characteristics in parallel processing conditions. For each of the algorithms, the coding efficiency, the amount of memory used and the quality of compressed images were determined. PSNR is an important quality assessment of signals, which, when it has its maximum value, tested images were reconstructed by compression. The performance analysis revealed that parallel processing strongly improves the used compression techniques and the processing speed and efficiency for image compression tasks. Analysis of these findings also brought to light the appropriateness of taking memory into account when applying these algorithms[11].

In this paper, Mohammad Ashraf Alam et al. (2018) offered a theoretical work on the LZW (Lempel-Ziv-Welch) algorithm as one of the lossless image compressions where authors benefited from the parallel computation of the GPU in order to enhance the rate of the compression. The paper showed thatif the LZW algorithm is to be implemented on the GPU, then the time saved in achieving the improved time is in terms of millisecond and therefore the chances of increasing the compression speed and efficiency with no compromise in image quality and can retrieve the original images from the compressed data in perfect quality. These results bear implications for multiple industries, specifically software and web design, mobile software application, and desktop software application. Efficient compression with significantly better image quality will help to enhance the clients' satisfaction, especially in applications that are based on image processing [12].

In (2020), Mouna Afif et al. accomplished the successful application of different computer vision algorithms on GPUs emphasizing on parallelism and boosting. This involved several creature activities like feature extraction and image integration computation; Gaussian Blur and others. There has been an increased availability of computation through the use of GPU programming platforms. The research highlighted the need to efficiently control access to GPU memory which decreased or completely eliminated banking conflicts and notably saved time in all CV tasks. This paper focuses on the role of efficient memory management in GPU computing since it is one of the ways of enhancing the performance. This research showed that in practice there was a quite notable speedup when the algorithm was moved to the GPU. The implementation of Sobel filter generated up to 8061 times on an average at 2048 × 2048 images compared to the traditional CPU implementations. This proves the high prospects of GPU acceleration for using in large image resolutions with high co-efficiency and great decrease of all mentioned image resolutions time. In conclusion, the results of the paper affirm that the application of NVIDIA CUDA is efficient in enhancing the current algorithms in computer vision stating improvements in efficiency and speed resulting from the utilization of GPU[13].

In (2020), AI AlSideiri A. et al. proposed and undertook an extension of the Fisher classification scheme in parallel within the CUDA framework with reference to GPUs. They applied the Fisher classification scheme to minimize the encoding time of fractal images, given the fractions by providing the best match. The encoding time, compression ratio, and maximum signal-to-noise ratio were chosen as the performance figures to assess how accurately and effectively the developed algorithm was implemented. For the experiment, 8 images of different size (512 × 512, 1024 × 1024 and 2048 × 2048 were employed. The experiments revealed that for some images a speedup of 6.4× was attained using NVIDIA GPU. The results also proved that utilising CUDA to execute the scheme considerably shorten the amount of time taken to encode a fractal image. This is of great importance because traditional fractal image compression techniques generally suffer from one major drawback: lengthy encoding times, resulting from the unsuitability of fractal images for fast and efficient segmentation during the encoding process [5].

Jiahao Pang et al. (2022), proposed a new approach for block classification of fractal images by using the Discrete Cosine Transform. The given method is developed to classify the image blocks in order to enhance the compression with the help of parallel processors by producing more efficient encoding of fractal images. The proposed framework has been applied to the NVIDIA CUDA architecture to bring improved parallel mode of encoding where simultaneous computations are possible. The results proved that the feature of the image compression speed is faster with quality enhancement, the encoding time is lower and the PSNR is approximately 0.4 less than the current methods making the method useful for the user who needs speed in image compression than the quality to be maintained, especially for the business which involves digital media and communication. The sets reflect the efficiency of parallel processing of the fractal images applied in this paper that is 107 times higher than that of the traditional fractal image compression techniques[14].

Wang Z et al. (2023) In this study, a parallel BCS scheme for images is introduced to enhance the performance of CS encoder of the image using a complex valued DFT measurement matrix. Theoretical study based on the method using minimum norm of the Frobenius is used to determine the efficiency of different measurement matrices. Next, simulation experiments are performed on the reconstructed performance of a sample of ten test images. Both theoretical analysis and simulation indicate that reconstructed performance using parallel processors based on the proposed DFT measurement matrix accords with PSNR and SSIM superior to other traditional measurement matrices include partial DCT matrix the partial Hadamard matrix the DHT matrix and chaos-based measurement matrix. However, the computational complexity of the DFT measurement matrix has been augmented while the enhanced reconstructed performance of the CS matrix has clearly demonstrated the advantages of our improvement. The emphasis is placed specifically, on the reconstruction of the lost image and the quality of which depends on the complex-valued DFT measurement matrix (1). All the experiments were carried on MATLAB 2014b environment in a laptop with configuration intel core i5, 2.20GHz CPU, 4GB RAM and window 7[15]. The details of the related work comparison are Table 1 below.

Table 1- Detailed table of previous studies.

Year	Researcher's name	Search Name	The method used	Advantages of the method	Negatives
2011	Bheshaj Kumar et a	Parallel Implementation for Fast and Efficient Image Compression in Spatial Domain	Block Truncation Coding (BTC)	-Processing efficiency -Scalability -Reduced execution time -Can be applied to different types of images and datasets -Improved compression performance -Independent block processing	-Implementation complexity -Hardware dependency -Communication overhead -Limited compression performance -Scalability issues Potential loss of quality
2012	Hao, Duan and others	Parallel design of JPEG-LS encoder on graphics processing units	JPEG-LS	-High performance and speed -Efficient memory management -Easy scalability -Reduced processing time -Ability to adapt to different image sizes -Using the capabilities of the GPU architecture, designed for intensive numerical computing.	-Data dependency issues where some operations must wait for the results of other operations -Serial nature of run-length encoding -Implementation complexity -Limited improvement in compression ratio -Resource constraints
2013	Kgotlaetsile and others	CUDA BASED PERFORMANCE EVALUATION OF THE COMPUTATIONAL EFFICIENCY OF THE DCT IMAGE COMPRESSION TECHNIQUE ON BOTH THE CPU AND GPU	Discrete Cosine Transform (DCT)	-Effectively reduces image file size while maintaining quality -CUDA programming model, the method takes advantage of the parallel processing capabilities of GPUs. -Enhanced computational efficiency -Scalability	-Implementation complexity -Hardware dependency -Memory management issues -Specific method of image compression
2014	Chongke Bi and others	A study of parallel data compression using proper orthogonal decomposition on the K computer	Proper Orthogonal Decomposition (POD) Data Compression	-Effective feature extraction -Low computational cost -Linear time decompression -High parallel efficiency -Scalability	-Complexity of implementation as researchers and practitioners may need a deep understanding of these techniques to apply them effectively -Dependence on data characteristics -Potential for information loss

					-Inter-processor communication costs
2015	Umesh B. Kodgule and B. A. Sonkamble	Discrete Wavelet Transform based Fractal Image Compression using Parallel Approach	Discrete Wavelet Transform (DWT)	-High compression ratios -Fast decompression -Real-time application compatibility -Significant acceleration	-Computationally intensive encryption -The quality of the reconstructed images may not always match the quality achieved by other advanced methods -Resource dependency
2016	I. P. Skirnevskiy 1 et al	Digital image processing using parallel computing based on CUDA technology	Using CUDA technology for digital image processing	-Concurrent execution -Efficiency in handling big data -Resource utilization	-Complexity in implementation May not be recommended for all denoising algorithms -Resource intensive algorithms
2017	P., S., Jagadeesh et al	Computational Paradigm and Quantitative Optimization to Parallel Processing Performance of Still Image Compression	Block Truncation Cipher (BTC), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) Partitioning in Hierarchical Trees (SPIHT) Embedded Zero Tree Wave (EZW)	Easy to implement -Provides good compression ratios while maintaining image quality -Efficient compression -Efficient encoding -Adaptable to different types of images	-May result in loss of visual quality -Limited compression -More complex -Higher computational cost -Can require significant memory resources,
2018	Mohammad Ashraf Alam et al	Faster Image Compression Technique Based on LZW Algorithm Using GPU Parallel Processing	LZW (Lempel-Ziv-Welch) algorithm	-Faster computation time -Scalability -Lossless compression	-Implementation complexity -Hardware dependency -Potential overhead
2020	Mouna Afif et al	Computer vision algorithms acceleration using graphic processors NVIDIA CUDA	high-speed computer vision algorithm using graphic processing unit by using the NVIDIA's programming framework compute unified	-High execution speeds -Improved efficiency -The method allows for parallel execution of tasks	-Hardware dependency -Implementation complexity -Potential overhead

			device architecture (CUDA)		
2020	AI AlSideiri A. et al	CUDA implementation of fractal image compression	Fisher classification scheme using CUDA	-Significant reduction in encoding time -Efficient use of resources -Scalability -Preservation of image quality	-Complex implementation -Dependence on hardware -Sacrifices some image quality for compression
2022	J. Pang, J. He, H. Guo, and M	A parallel compression framework for fractal images using DCT block classification	Uses discrete cosine transform (DCT)	-Significant speed improvement -Efficient block sorting -Minimum quality loss -Practical application for real-world use -Using modern hardware by leveraging NVIDIA's CUDA architecture	-Implementation complexity -The method's reliance on the NVIDIA CUDA architecture means it is dependent on specific hardware capabilities -Its effectiveness may vary across image types or datasets -Possibility of over fitting, which can lead to performance degradation when applied to new or unseen images
2023	Wang Z et al	Image parallel block compressive sensing scheme using DFT measurement matrix	Parallel Block Compression Sensing (BCS)	-Reconstruction quality -Minimum quantization effect -Method allows parallel recovery of each column of transformed data -Processing speed and efficiency	-Complexity of compressed data -Reliance on sparse representation -Computational resources

Based on the literature we have presented, we will recommend the best method for parallel image compression by analyzing different methods and their advantages and disadvantages.

After reviewing the research from 2011 to 2023, it became clear to us that the DCT transform used in CUDA technology is the best and most effective method for parallel image compression. We based this on factors including:

Repetition and effectiveness have been consistently demonstrated in multiple studies such as (Kgotlatsile et al. 2013, P. Jagadeesh et al. 2017, J. Pang et al. 2022) and always give positive results.

In addition to the balance of advantages, it reduces image file sizes while maintaining image resolution and quality. The aforementioned method has been improved over time in 2022 by the researcher Pang et al. It showed significant development and improvement in speed with minimal loss of image quality.

4. Discussion

Looking at many such studies as [6, 7], we noted that all called for better methods of image compression using parallel processors. In these studies, Graphics Processing Unit (GPU) were utilized to enhance the acceleration ratios in terms of compression, as reflected from the test outcomes that involved a refined execution speed and better reconstructed image quality than CPU in the traditional models. These studies also confirmed that with the use of parallel processors not only do we get a faster executing program but the value of 'P', the computational cost is minimized without much loss in image quality.

Furthermore, other related works, for example, [1] have exhibited that the DCT with parallel processors helps enhance the efficiency of computing time. This reveals why these processors should be utilised in enhancing the quality of the reconstructed picture and time that is otherwise used by the central processing unit. But in the case of some researches for example [15]there are some difficulties connected with applying certain techniques such as DFT which may in some cases augment the computational complexity. However, the results were shown that do not exclude the increase of the complexity in the described algorithms but does not decrease the quality of the reconstructed image which proves the tendency to find the balance between the time consumption and image quality.

In all, the study focusses on the need to employ parallel processors in image compression, notably the GPUs based ones. Image compression can be made faster and more efficient through parallel processing; for instance, those that employ the use of illustrative cards, GPUs. These devices have hundreds of cores where the computations can be run in parallel and, therefore, image compression is much faster than when using the conventional microprocessors known as the CPUs. Despite the fact that all the reviewed studies focus on image compression, and different methods depend on adopting parallel processors since they enable the execution of more than one process than. These results align with the initiatives that have been made towards enhancing the realization of the image compression operation hence contributing to the advancement of practicable solutions in use in areas including medical, space and communication where parallel processors increase the level of image compression performance.

5. Conclusion

Therefore, in conclusion of this study, the authors can confirm that utilization of parallel processors on the GPU has been effective in enhancing the image compression process. This paper also discovered interesting facts, that are, the parallel processing techniques are much faster, more computational and have enhanced quality of reconstructed images than the original CPU-based literature. Several papers have provided empirical evidence that these approaches can lead towards the reduction of the encoding time and the performance of the compression with minimal loss of quality for application in fields that demand large data performance under short amount of time. The results also show the integration between image compression algorithms and parallel processing techniques based on technologies such as CUDA, DCT, and DWT, where compression processes are accelerated by distributing tasks across multiple processors. Moreover, we find that taking advantage of parallel image compression techniques has a positive impact on reducing communication costs between processors and improving memory efficiency. Given the importance of these processors in areas such as medical imaging, communications, and space, this study suggests that these algorithms need to continue to improve to meet the needs of the accelerating digital age, with a focus on reducing computational complexity and balancing performance speed with image quality.

References

[1] K. M. Modieginyane, Z. P. Ncube, and N. Gasela, "CUDA based performance evaluation of the computational efficiency of the DCT image compression technique on both the CPU and GPU," *arXiv preprint arXiv:1306.1373*, 2013.

[2] M. A. Joshi, M. S. Raval, Y. H. Dandawate, K. R. Joshi, and S. P. Metkar, "Image and video compression," *(No Title)*, 2014.

[3] A. A. KE and N. Arivazhagan, "Performance Optimization of Parallel Programming."

[4] M. Wilde, M. Hategan, J. M. Wozniak, B. Clifford, D. S. Katz, and I. Foster, "Swift: A language for distributed parallel scripting," *Parallel Computing*, vol. 37, no. 9, pp. 633-652, 2011.

[5] A. Al Sideiri, N. Alzeidi, M. Al Hammoshi, M. S. Chauhan, and G. AlFarsi, "CUDA implementation of fractal image compression," *Journal of Real-Time Image Processing*, vol. 17, no. 5, pp. 1375-1387, 2020.

[6] B. Kumar, K. Thakur, G. Sinha, B. C. Patel, and S. Choubey, "Parallel implementation for fast and efficient image compression in spatial domain," in *3rd International Conference on Machine Learning and Computing (ICMLC 2011)*, 2011, vol. 4, pp. 378-381.

[7] H. Duan, Y. Fang, and B. Huang, "Parallel design of JPEG-LS encoder on graphics processing units," *Journal of Applied Remote Sensing*, vol. 6, no. 1, pp. 061508-061508, 2012.

[8] C. Bi, K. Ono, K.-L. Ma, H. Wu, and T. Imamura, "A Study of Parallel Data Compression Using Proper Orthogonal Decomposition on the K Computer," in *EGPGV@ EuroVis*, 2014, pp. 1-8.

[9] U. B. Kodgule and B. Sonkamble, "Discrete wavelet transform based fractal image compression using parallel approach," *International Journal of Computer Applications*, vol. 122, no. 16, 2015.

[10] I. Skirnevskiy, A. Pustovit, and M. O. Abdrashitova, "Digital image processing using parallel computing based on CUDA technology," in *Journal of Physics: Conference Series*, 2017, vol. 803, no. 1: IOP Publishing, p. 012152.

[11] P. J. Kumar, T. L. Huan, and Y. Yung, "Computational Paradigm and Quantitative Optimization to Parallel Processing Performance of Still Image Compression," *Circulation in Computer Science*, vol. 2, no. 4, pp. 11-17, 2017.

[12] M. A. Alam, F. Ahsan, M. Subhani, F. F. Hossain, M. S. Islam, and K. N. Ruma, "Faster image compression technique based on LZW algorithm using GPU parallel processing," in *2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, 2018: IEEE, pp. 272-275.

[13] M. Afif, Y. Said, and M. Atri, "Computer vision algorithms acceleration using graphic processors NVIDIA CUDA," *Cluster Computing*, vol. 23, no. 4, pp. 3335-3347, 2020.

[14] J. Pang, J. He, H. Guo, and M. Chen, "A parallel compression framework for fractal images using DCT block classification," in *2022 IEEE 21st International Conference on Ubiquitous Computing and Communications (IUCC/CIT/DSCI/SmartCNS)*, 2022: IEEE, pp. 245-253.

[15] Z. Wang, Y. Jiang, and S. Chen, "Image parallel block compressive sensing scheme using DFT measurement matrix," *Multimedia Tools and Applications*, vol. 82, no. 14, pp. 21561-21583, 2023.