Ambidextrous: Journal of Innovation, Efficiency and Technology in Organizations

https://journal.takaza.id/index.php/ambidextrous

Vol. 5 , no. 01 , 2026 , pp. 105-116
E-ISSN: 3031-7002

E-mail:ambidex@takaza.id



Effectiveness Analysis of Run-Length Encoding and Huffman Coding Combination in Digital Image Compression

Muhammad Ar Rauuf Mudafri[1], Muhammad Harris[2]
Program Studi Teknologi Informasi Fakultas Ilmu Komputer Dan Teknologi Informasi
Universitas Muhammadiyah Sumatera Utara
Corresponding Author e-mail : muhammadarrauuff@gmail.com, muhammadharis@umsu.ac.id

Article History:
Received : 07-10-2026
Revised : 06-11-2026
Accepted : 07-11-2026

Keywords: Digital Image, Compression Ratio, Huffman Coding, Lossless Compression, Run-Length Encoding

Abstract : *The increasing use of digital images has led to a greater need for efficient data storage and transmission. This study aims to implement and analyze the effectiveness of the combination of Run-Length Encoding (RLE) and Huffman Coding in digital image compression based on Compression Ratio, Space Saving, processing time, and image integrity. The study uses an experimental method with a quantitative and implementative approach. The research population is two-dimensional digital images with a sample of 20 images, consisting of 10 RGB images and 10 grayscale images from Kaggle and DIV2K. The research instrument is a computer with a Python-based system, while the analysis is carried out quantitatively and descriptively through Black Box Testing and Performance Testing. The results of the study show that the compression and decompression processes were successful and the decompressed images remained intact. However, the size efficiency is not optimal. The RGB1 image produces a Compression Ratio of 0.177 and Space Saving of -464.84%, while RGB3 produces 0.0363 and -2654.78%. In conclusion, the combination of RLE and Huffman Coding is able to maintain image integrity, but is not yet effective in reducing file size in the tested samples.*

How to Cite : Muhammad Ar Rauuf Mudafri, Muhammad Harris. (2026). *Effectiveness Analysis of Run-Length Encoding and Huffman Coding Combination in Digital Image Compression*. 5(01). Pp.105-116. https://doi.org/10.61536/ambidextrous.v5i01.929

https://doi.org/10.61536/ambidextrous.v5i01.929

This is an open-access article under theCC-BY-SA License.

Introduction

The development of digital technology in recent years has driven the increased use of digital images in various aspects of life. Digital images have been widely used in social media, security systems, telemedicine, education, industry, and even cloud computing-based services. This growth is supported by the use of various digital devices such as smartphones, surveillance cameras (CCTV), drones, and Internet of Things (IoT) devices, which are capable of producing increasingly high-resolution image data (Lubis et al., 2022). This condition has caused the amount of visual data stored and transmitted to continue to increase over time. In fact, developments in image and video data processing indicate that

the need for compression methods is increasingly important as the volume of visual data that must be processed, stored, and transmitted increases (Dong & Pan, 2023). This phenomenon indicates that the increasing use and quality of digital images directly impacts the growing need for efficient visual data management.

The increasing number and size of digital images has become a challenge in data management, primarily because large image files require higher storage capacity and transmission bandwidth (Fitrani & Puspitaningrum, 2023). This situation can increase storage space requirements, increase network load, reduce data transfer speeds, and increase server storage costs, especially in environments with limited internet connections. This problem is increasingly relevant in the fields of telemedicine, security surveillance systems, digital archives, and medical image data processing, which require continuous storage of large amounts of data. Research on image compression also shows that the large size of image data is one of the main reasons for the need for compression techniques to improve data storage and transmission efficiency (Yu et al., 2023). Therefore, the increasing use of digital images needs to be accompanied by data size management that can support storage and transmission efficiency without compromising the information contained in the images.

One of the challenges in reducing image size is the use of lossy compression methods. These methods can produce smaller file sizes by removing some information from the image, but the consequence can be a decrease in image quality. This makes it less suitable for fields that require accurate visual data, such as medical images, digital forensics, and important archival documents (He et al., 2022). Therefore, the need for compression methods is not only related to the ability to reduce file size but also to retain the original image information after the decompression process. In this context, lossless compression is an approach that can reduce data size without losing the original information from the image. Lossless approaches allow the compressed data to be reconstructed accurately, thus remaining suitable for data that requires information integrity (Wang et al., 2023). Therefore, the selection of a compression method needs to consider the balance between data efficiency and the ability to preserve the original information.

The next issue concerns how to achieve compression efficiency through methods that maintain image integrity. Run-Length Encoding (RLE) works by storing data values and the number of repetitions, making it effective for data with recurring patterns, while Huffman Coding provides a shorter binary code representation for data with higher frequency of occurrence (Lubis et al., 2022). Both methods have different mechanisms for reducing redundancy and optimizing data representation. The combination of the two is expected to improve digital image compression efficiency while maintaining the original image quality. Research on the development of RLE- and Huffman-based compression methods also shows that data distribution characteristics and repetition patterns influence the method's ability to produce more concise data representations (Zhu et al., 2025). Furthermore, both methods have relatively simple algorithmic complexity and low computational resource requirements, making them still relevant for application in various modern digital systems (Angelita & Rosandy, 2025).

These issues highlight the need for testing to determine the extent to which the combination of Run-Length Encoding and Huffman Coding can produce effective compression across various digital image characteristics. The effectiveness of the method is assessed not only by its ability to produce smaller file sizes, but also by its compression ratio, storage space savings, compression and decompression processing times, and the quality and integrity of the decompressed image. Evaluation of lossless compression algorithms can generally be performed by considering the compression ratio, encoding time, decoding time, and compression efficiency characteristics (Hanif et al., 2023). Different image characteristics can affect compression results because the level of redundancy and data distribution in each image are not always the same. Therefore, an analysis of the combination of the two algorithms is necessary to determine the level of compression efficiency, file size reduction ratio, and method performance across various digital image characteristics.

This study aims to implement a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms in a digital image compression system, analyze its effectiveness in reducing file size based on compression ratio and space saving parameters, and evaluate the quality and integrity of the decompressed image according to the characteristics of lossless compression. This research is important because the increasing use of digital images demands efficient data storage and transmission, while the

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

need for the integrity of image information must still be maintained. The novelty of this research lies in the application of a combination of Run-Length Encoding and Huffman Coding to analyze the effectiveness of digital image compression by considering file size, compression ratio, space saving, compression and decompression time, and the integrity of the decompressed image on various characteristics of digital images. Thus, this study is directed to provide an overview of the level of effectiveness of the combination of the two algorithms in producing efficient image compression while maintaining the original image information.

Method

This research is an experimental study with a quantitative and implementative approach. The study was conducted by applying a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms to the digital image compression process to determine its effectiveness and performance (Bastian et al., 2023; Amalia et al., 2024). A quantitative approach was used by measuring file size, compression ratio, space saving, compression time, decompression time, and the quality of the decompressed image. The research process includes dataset collection, preprocessing, algorithm implementation, compression and decompression, and evaluation of test results.

The research instrument is a computer device used for the implementation and testing of a Python-based digital image compression system with the help of NumPy, OpenCV, and Pillow. Data were collected through literature and documentation studies, while system testing used Black Box Testing and Performance Testing. Data analysis was carried out quantitatively and descriptively by comparing file sizes before and after compression, compression ratio, space saving, compression time, decompression time, and the suitability of the decompressed image with the original image. The results of the analysis are presented in the form of tables, graphs, and descriptive descriptions.

The research population consists of two-dimensional digital images used as testing objects for the compression system. The research sample consists of 20 digital images, namely 10 RGB images and 10 grayscale images. The datasets were obtained from Kaggle and DIV2K (DIVerse 2K Resolution Dataset) which have a variety of objects and image characteristics. These samples are used to test the effectiveness of the combination of RLE and Huffman Coding on different image characteristics.

The research procedure begins with the collection of digital image datasets, followed by preprocessing to adjust the images for processing by the system. Next, the image pixel values are processed using RLE to reduce redundancy based on repetition patterns, followed by Huffman Coding to produce a more efficient bit representation. After compression is complete, Huffman Decoding and RLE Decoding processes are carried out to restore the image to its original form. The final stage is carried out through testing and evaluation based on file size, compression ratio, space saving, compression time, decompression time, and the integrity of the decompressed image.

Results and Discussion

System Interface Implementation

The system interface implementation is designed to facilitate users in performing the digital image compression process using a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms. The interface was developed using the Python programming language with the Tkinter library so that the entire testing process can be carried out through a single application window. As shown in Figure 4.1, the application provides several main menus consisting of Select Image to select the image to be processed, Compress to run the compression process which is automatically followed by decompression and evaluation, Save Compressed to save the compressed file in .rlehuff format, Reset to return the application to its initial state, Clear History to delete the test history, and Save History to CSV to save all test results into a CSV file as research analysis data.

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

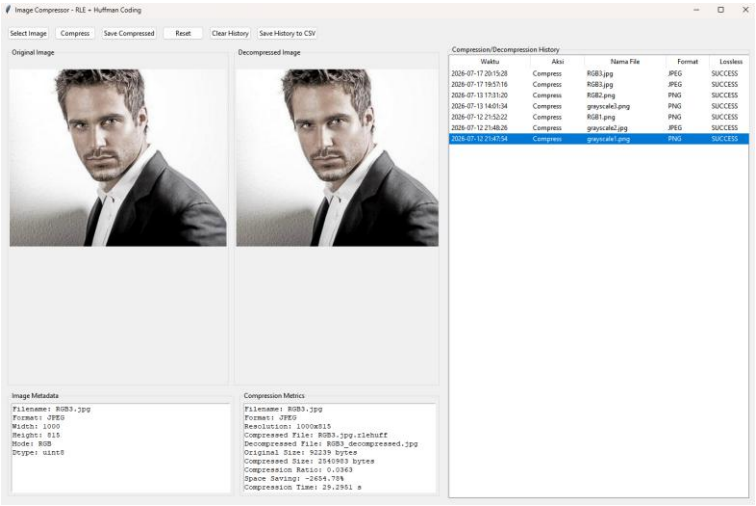


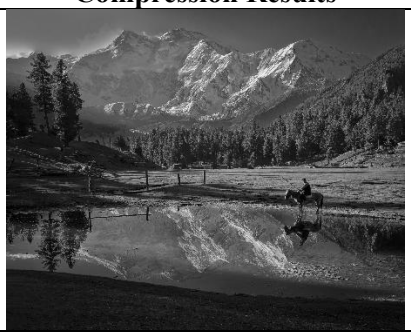
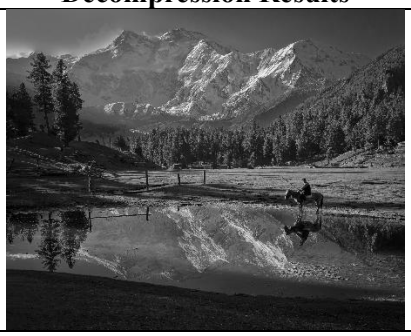
Figure 1. System View

In addition to providing the main menu, the interface also displays several integrated information panels. The Original Image panel is used to display the original image selected by the user, while the Decompressed Image panel displays the decompressed image as a form of visual verification that the compression process is lossless. At the bottom of the interface is the Image Metadata panel that displays image information such as file name, format, resolution, color mode, and pixel data type. Next to it is the Compression Metrics panel that presents evaluation results in the form of original file size, compressed file size, Compression Ratio value, Space Saving, compression time, decompression time, and lossless compression success status. Meanwhile, the Compression/Decompression History panel functions to record all testing activities along with information on time, file name, image format, and compression result status so that each test can be documented and reviewed. With this interface design, the entire process from image selection, compression, decompression, evaluation, to storing test results can be carried out in an integrated manner in one application, thus facilitating the process of analyzing the effectiveness of the combination of RLE and Huffman Coding algorithms.

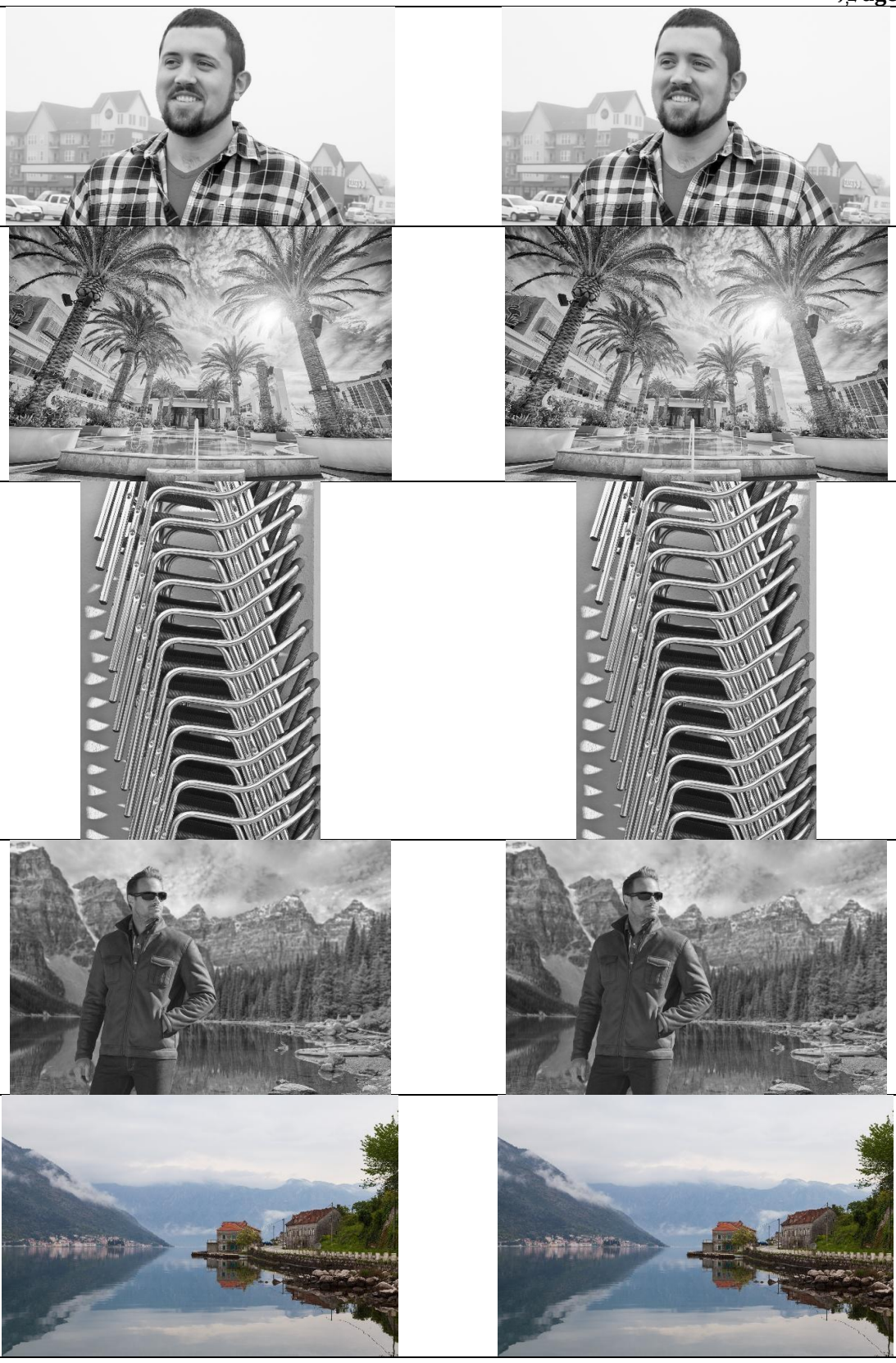
Evaluation of Image Compression Results

Based on the results of the system implementation, the image compression and decompression process was successfully carried out using a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms. The decompression results show that the reconstructed image has the same visual appearance as the original image, so that no significant differences were found when observed directly by the human eye. This indicates that the compression process is lossless, meaning that all pixel information in the original image can be recovered without data loss. The system verification results also show the status Lossless Compression: SUCCESS, which indicates that the decompressed image is identical to the original image.

Table 1. Compression & Decompression Results of Grayscale & RGB Images

Compression Results	Decompression Results
	

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license



https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license



3. Evaluation of Image Metric Value Results

The compression evaluation was conducted to analyze the effectiveness of the Run-Length Encoding (RLE) and Huffman Coding algorithms on the tested digital images. The analysis was conducted based on several evaluation parameters, namely Original Size versus Compressed Size, Compression Ratio, and Space Saving. Through these parameters, changes in file size before and after compression can be determined, while also measuring the level of compression efficiency produced by the implemented method.

1. Analysis of File Compression Size Results

A. File Compression Results on RGB Images

Based on the results of tests conducted on grayscale images, information was obtained regarding changes in file size before and after the compression and decompression processes. Grayscale images were chosen as one of the test objects because they only have one color intensity

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

channel, thus having different characteristics compared to RGB images. Observations on file size were carried out to determine how the implementation of the Run-Length Encoding (RLE) and Huffman Coding algorithms affected the final results of the processed images. Based on file property information, the original image size was recorded at 4.99 MB (5,238,645 bytes), while the decompressed image size was 3.55 MB (3,724,177 bytes). These results indicate that the decompressed file size is smaller than the original file size.

However, visual observation shows that the decompressed image retains the same quality as the original image, with no changes in object shape, decreased sharpness, or loss of detail recognizable to the human eye. The difference in file size is caused by the re-encoding process performed when the decompressed image is saved back in PNG format. This process can result in different file sizes, although the pixel information contained within the image is retained. Therefore, changes in the decompressed file size cannot be used as an indicator of the success of the compression algorithm.

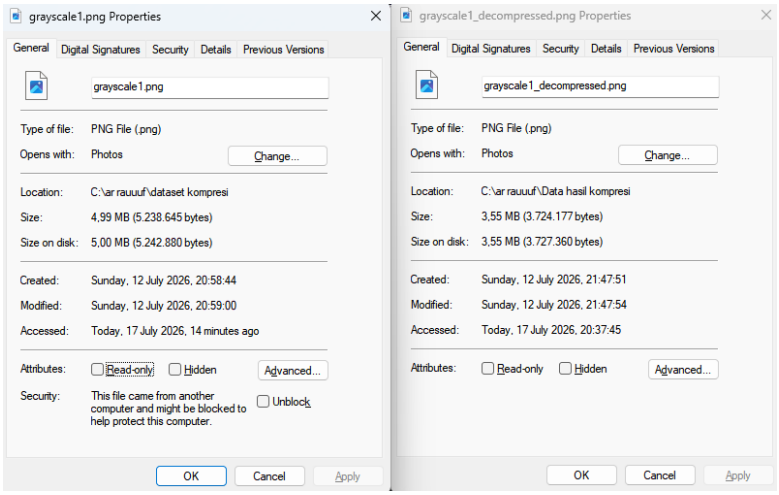


Figure 2. Comparison of Compress & Decompress Grayscale Image Results

Therefore, in this study, the evaluation of the effectiveness of the combination of Run-Length Encoding (RLE) and Huffman Coding algorithms is focused on the size of the compressed file (.rlehuff) along with the Original Size, Compressed Size, Compression Ratio, and Space Saving values, while the decompressed image is used to verify that the compression process still maintains the visual quality and image information intact according to the characteristics of lossless compression.

B. File Compression Results on RGB Images

Based on the results of tests conducted on RGB images, information was obtained regarding changes in file size before and after the compression and decompression processes. RGB images were used as test objects because they have three color channels, namely Red (R), Green (G), and Blue (B), so the amount of data processed is greater than that of grayscale images. Observations on file sizes were carried out to determine how the implementation of the Run-Length Encoding (RLE) and Huffman Coding algorithms affected the final results of the processed images. Based on file property information, the size of the original image and the decompressed image showed a difference in file size.

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

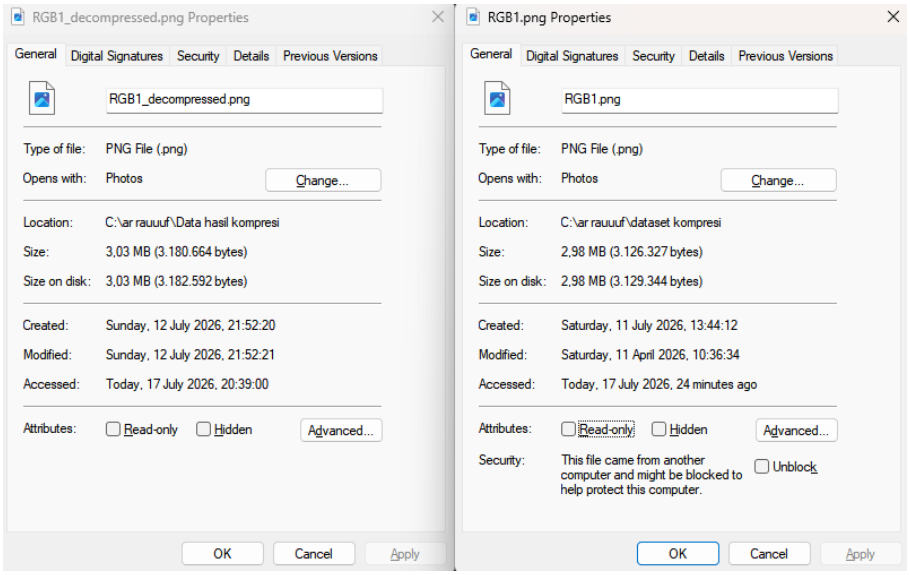


Figure 3. Comparison of RGB Image Compression & Decompression Results

However, visual observation results show that the decompressed image still has the same quality as the original image without any changes in color, sharpness, or object details that can be observed by the human eye. The difference in file size is caused by the re-encoding process that is carried out when the decompressed image is saved back to its original format. This process allows the file size to change even though all pixel information is maintained. Thus, changes in the size of the decompressed file cannot be used as an indicator of the success of the compression algorithm. Therefore, in this study, the evaluation of the effectiveness of the combination of Run-Length Encoding (RLE) and Huffman Coding algorithms focuses on the size of the compressed file (.rlehuff) along with the Original Size, Compressed Size, Compression Ratio, and Space Saving values, while the decompressed image is used to verify that the compression process is lossless, that is, it is able to maintain the visual quality and image information intact after the decompression process.

2. Compression Ratio Analysis

Before analyzing all test results, an example of the Compression Ratio calculation process using one of the tested image samples is presented. The sample used in this calculation example is the RGB1 image, which is one of the 10 RGB images used as test data in this study. The presentation of one calculation example aims to provide a clear picture of the stages of obtaining the Compression Ratio value, starting from using the Original Size data as the original file size to the Compressed Size as the compressed file size. Thus, readers can understand how the Compression Ratio value is obtained based on the formula explained in the previous chapter. It should be noted that the calculation process shown in the RGB1 image represents the same calculation method applied to all other RGB image samples. Therefore, differences in the Compression Ratio value for each image are only influenced by the characteristics of each image, such as file size and pixel data distribution, while the calculation stages remain using the same procedure. The calculation results for all samples are then presented in tabular form and further analyzed to determine the effectiveness of the combination of Run-Length Encoding (RLE) and Huffman Coding algorithms in compressing digital images.

As an example of the application of the Compression Ratio calculation, this study uses the test results on the RGB1 image, which is one of the ten RGB image samples used in the testing process. The presentation of this calculation example aims to show the stages of obtaining the Compression Ratio value systematically based on the original file size (Original Size) and the compressed file size (Compressed Size). The calculation steps shown in the RGB1 image are also applied to all other image samples to produce different Compression Ratio values according to the characteristics of each image. Thus, this calculation example can be used as a reference in understanding the evaluation process carried out in this study.

The formula for solving the Compression Ratio used is as follows:

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

$$CompressionRatio = \frac{OriginalSize}{CompressedSize} = \frac{3.126.327}{17.658.639} = 0{,}177 \quad (4)$$

Information:

- *Original Size*= image file size before compression process (bytes).
- *Compressed Size*= image file size after compression process (bytes).

Based on the test results on the RGB1 image, the following data was obtained.

- *Original Size*= 3,126,327 bytes
- *Compressed Size*= 17,658,639 bytes

Based on the calculation results, the Compression Ratio value is 0.177. This value indicates that the compressed file size is larger than the original file size. This is due to the characteristics of the tested image and the encoding process using a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms, which in this sample resulted in a compressed file size larger than the initial file size. Therefore, a Compression Ratio value below 1 indicates that the compression process on the RGB1 image has not been able to produce an effective file size reduction. The same calculation results were then applied to nine other RGB image samples to obtain a comprehensive picture of the performance of the proposed compression method.

3. Space Saving Analysis

To provide an understanding of the Space Saving calculation process, this study presents an example calculation using test results on the RGB1 image. This image was selected as a representative sample of the 10 RGB images used in the compression method testing. The presentation of this example is intended to allow readers to follow each step of the Space Saving calculation in detail based on the Original Size and Compressed Size data obtained after the compression process is completed. Through this example, the calculation mechanism can be more easily understood before discussing the overall test results. It should be understood that the calculation procedure used on the RGB1 image is also applied identically to the other nine RGB image samples. The differences in the resulting Space Saving values for each image are not caused by changes in the calculation method, but are influenced by the characteristics of each image, such as the initial file size, pixel distribution pattern, and level of data redundancy. All calculation results are then summarized in tabular form for further analysis to evaluate the ability of the combination of Run-Length Encoding (RLE) and Huffman Coding algorithms to optimize storage space usage in the digital image compression process.

As part of the compression effectiveness evaluation, this study also calculated the Space Saving value. This parameter is used to determine the percentage change in file size after the compression process compared to the original file size. Unlike the Compression Ratio, which shows the comparison between the original file size and the compression result, Space Saving describes the amount of storage space that is successfully saved or even increased after the compression process is carried out. Therefore, the Space Saving value can be used as an indicator to assess the level of efficiency of the applied compression method. In this study, the Space Saving calculation example uses the test results on the RGB1 image, while the same calculation process is applied to all other image samples.

The Space Saving solution formula used is as follows.

$$\begin{aligned} SpaceSaving &= \frac{OriginalSize - CompressedSize}{OriginalSize} x100\% \\ &= \frac{3.126.327 - 17.658.639}{3.126.327} \\ &= \frac{-14.532.312}{3.126.327} x100\% \\ &= -4{,}6484 \times 100\% \\ &= -464{,}84\% \end{aligned} \quad (5)$$

Based on the calculation results, the Space Saving value is -464.84%. A negative value indicates that the compressed file size is larger than the original file size, so there is no storage space savings. On the contrary, the compression process actually causes the file size to increase by 464.84% of the initial file size. This condition indicates that the combination of the Run-Length Encoding (RLE) and Huffman

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

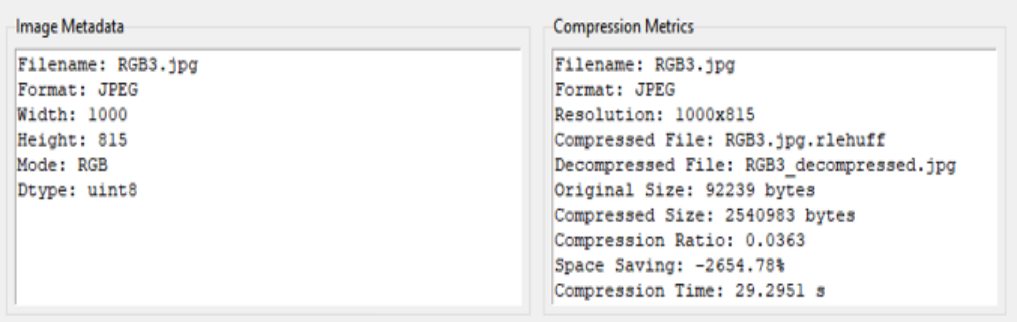
Coding algorithms on the RGB1 image sample has not been able to produce efficient compression. This may be influenced by the characteristics of the image data that already has a high compression level or does not have enough redundancy patterns to be optimally utilized by the Run-Length Encoding (RLE) algorithm before encoding using Huffman Coding. The same calculation process is then applied to all other RGB image samples to obtain a comprehensive picture of the level of storage space efficiency produced by the proposed compression method.

4. Compression & Decompression Time Analysis

Based on the test results on the RGB1 image, the compression time was 142.0235 seconds, while the decompression time was 68.8071 seconds. These results indicate that the compression process takes longer than the decompression process, with a time difference of approximately 73.2164 seconds. This difference occurs because in the compression stage the system must carry out a series of processes, starting from reading the image data, identifying the pixel sequence using the Run-Length Encoding (RLE) algorithm, building a Huffman tree based on the frequency of data occurrence, generating the Huffman code, to saving the compression results into a file with the extension .rlehuff. All of these stages require a relatively more complex computational process so that the time required is longer. On the other hand, the decompression process requires a shorter time because the system only reads the compressed data, the Huffman decoding process to return the data to the Run-Length Encoding (RLE) result form, then carries out the Run-Length Decoding process to reconstruct the pixel data back into the original image. Based on these results, it can be concluded that the combined implementation of the Run-Length Encoding (RLE) and Huffman Coding algorithms has the characteristic of longer processing times in the compression stage compared to the decompression stage. This indicates that the majority of the computational burden is on the data encoding process, while the process of restoring the data to its original form can be done relatively quickly.

5. Analysis of the RLE + Huffman Algorithm Compression Process

The image shows the results of the RGB3.jpg image compression process using a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms on the developed system. The image used is in JPEG format with a resolution of 1000 × 815 pixels and an original file size of 92,239 bytes. After the compression process is carried out, the system produces a compressed file with the name RGB3.jpg.rlehuff. This information indicates that the compression process is carried out on the image pixel data, then the RLE encoding results are processed using Huffman Coding and saved in a special .rlehuff format. Thus, the compressed file size is the actual size of the .rlehuff file produced by the system after all compression stages are completed.



Compression Metrics Results

Based on the test results, the compressed file size was recorded at 2,540,963 bytes. This value indicates that the compressed file size is actually larger than the original image size. This condition can be seen from the Compression Ratio value of 0.0363, which is obtained by comparing the original file size to the compressed file size. The calculation can be shown as follows:

$$CR = \frac{92.239}{2.540.963} = 0{,}0363 \quad (6)$$

https://doi.org/10.61536/ambidextrous.v5i01.929
This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

The Compression Ratio value which is far below 1 indicates that in RGB3 images the combination of RLE and Huffman processes has not been able to produce a reduction in file size. This condition is also reinforced by the Space Saving value of -2654.78%, which indicates that after the compression process the file size has increased significantly compared to the initial file size. This indicates that the representation of the compression results along with the information needed for the decoding process requires more storage space than the original data in the RGB3 image. This characteristic can occur when the image data has a low level of redundancy or the distribution of pixel values is less suitable for effective utilization by RLE and Huffman Coding.

In addition to the file size, the system recorded a Compression Time of 29.2951 seconds. This time includes a series of compression processes starting from reading image data, processing using RLE, forming Huffman representations, encoding processes, to saving the results into a .rlehuff file. On the other hand, the interface displays the original image and the decompressed image with the same visual appearance, while the test history shows the status Lossless: SUCCESS. This shows that although the compression process on RGB3 images does not result in storage space savings, the decompression process still successfully restores the image so that image information can be maintained. Thus, the results on RGB3 show that the success of a lossless compression method is not only determined by the ability to restore data intact, but also by the algorithm's ability to produce a data representation that is smaller than the original size.

Conclusion

This study successfully implemented a combination of Run-Length Encoding (RLE) and Huffman Coding algorithms for digital image compression and decompression, addressing the research objective of evaluating compression effectiveness in terms of file size reduction and image integrity. The results demonstrate that the proposed method can reconstruct RGB and grayscale images while maintaining their original pixel information. However, its storage efficiency was not optimal across the tested samples. Specifically, the RGB1 image produced a Compression Ratio of 0.177 and Space Saving of -464.84%, while RGB3 produced a Compression Ratio of 0.0363 and Space Saving of -2654.78%. These negative Space Saving values indicate that the compressed files were larger than their original counterparts. Therefore, the findings confirm that successful image reconstruction does not necessarily correspond to effective file size reduction. The effectiveness of the combined algorithms depends on image characteristics, particularly pixel repetition patterns and the level of data redundancy that can be utilized by RLE and Huffman Coding. The main contribution of this study is demonstrating that the combination of these algorithms can preserve image information while its compression efficiency varies depending on the characteristics of the input images.

This study has several limitations, including the relatively small dataset of 20 images, consisting of 10 RGB and 10 grayscale images, and the limited diversity of image formats and data characteristics. Furthermore, the evaluation did not include comparative testing against standalone RLE, standalone Huffman Coding, or other compression algorithms. Although the reconstructed images were visually identical to the original images, objective verification using pixel-by-pixel comparison or image integrity metrics should be incorporated to strengthen the evaluation. Future research should therefore examine larger and more diverse image datasets, compare the combined method with standalone RLE and Huffman Coding, and investigate alternative algorithm combinations to improve compression efficiency. Additional evaluations should also include objective image integrity verification, compression ratio, space saving, and processing time to provide a more comprehensive assessment of compression performance.

Bibliography

Amalia, F., Fiqih, R., & Sena, Y. (2024). *Metodologi penelitian ilmu komputer*. Buku Metodologi Penelitian Ilmu Komputer. https://www.teknokrat.ac.id

Angelita, A., & Rosandy, T. (2025). Implementasi algoritma Run Length Encoding (RLE) untuk efisiensi penyimpanan data teks berbasis Python. *Jurnal Mnemonic*, *8*(2), 266–270. https://doi.org/10.36040/MNEMONIC.V8I2.14758

This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license

Bastian, A., Mardiana, A., Berliani, M., & Firmansyah, M. B. (2023). Run Length Encoding compression on virtual tour campus to enhance load access performance. *JOIN: Jurnal Online Informatika*, *8*(1), 27–35. https://doi.org/10.15575/JOIN.V8I1.1000

Dong, Y., & Pan, W. D. (2023). A survey on compression domain image and video data processing and analysis techniques. *Information*, *14*(3), 184. https://doi.org/10.3390/info14030184

Fitrani, L. D., & Puspitaningrum, A. C. (2023). Utilization of UML (Unified Modeling Language) in the design of academic information systems based on the OOAD method. *Sistemasi: Jurnal Sistem Informasi*, *12*(2), 614–625. https://doi.org/10.32520/STMSI.V12I2.2871

Hanif, A., Wahyudi, E., Adianto, H., & Martanto, L. (2023). Komparasi performa algoritma kompresi data lossless menggunakan rasio kompresi dan penghematan ruang. *Journal of Information and Technology*, *11*(1), 148–157. https://doi.org/10.32664/J-INTECH.V11I1.863

He, D., Yang, Z., Peng, W., Ma, R., Qin, H., & Wang, Y. (2022). ELIC: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5718–5727. https://doi.org/10.1109/CVPR52688.2022.00563

Lubis, H., Srisulistiowati, D. B., & Ramdhani, A. I. (2022). Huffman coding pada image compression. *Jurnal Gerbang STMIK Bani Saleh*, *12*(1), 16–25. https://jurnal.stmik.banisaleh.ac.id/ojs2/index.php/JIST/article/view/89

Wang, R., Liu, J., Sun, H., & Katto, J. (2023). Learned lossless image compression with combined channel-conditioning models and autoregressive modules. *IEEE Access*, *11*, 73462–73469. https://doi.org/10.1109/ACCESS.2023.3291591

Yu, L., Li, H., Li, J., & Li, W. (2023). Lossless compression of large aperture static imaging spectrometer data. *Applied Sciences*, *13*(9), 5632. https://doi.org/10.3390/app13095632

Zhu, Y., Liu, Y., Zhu, Y., Huang, M., Jiang, J., & Zhang, Y. (2025). Lossy infrared image compression based on wavelet coefficient probability modeling and run-length-enhanced Huffman coding. *Sensors*, *25*(8), 2491. https://doi.org/10.3390/s25082491

This open-access article is distributed under a Creative Commons Attribution (CC-BY-SA) 4.0 license