Journal of Informatics, Computer Science, Data Science And Artificial Intelligence
Volume 4 No. 1 June 2026, ISSN : 3031-9145

IMPLEMENTATION OF DCT (DISCRETE COSINE TRANSFORM) AND MEDIAN CUT ALGORITHMS FOR CONVERTING JPG IMAGE FILES INTO GIF FORMAT

Muhammad Fauzi Mukmin Nasution[1] , Khairuddin Nasution[2],Tasliyah Haramaini[3]
[1]Universitas Islam Sumatera Utara, fauzynasution04@gmail.com
[2]Universitas Nahdlatul Ulama Sumatera Utara, khairuddin_nst@uisu.ac.id
[3]Universitas Nahdlatul Ulama Sumatera Utara, tazlie@gmail.com

ARTICLE INFO

ARTICLE HISTORY :

Received : 5 June 2026

Revised : 20 June 2026

Accepted : 30 June 2026

***Keywords**: Discrete Cosine Transform, DCT, Median Cut, Image Compression, Color Quantization, GIF, JPG, Python, Digital Image Processing*

ABSTRACT

Digital image compression plays an important role in reducing storage requirements and improving transmission efficiency while maintaining acceptable visual quality. This study implements the Discrete Cosine Transform (DCT) and Median Cut algorithms to convert JPG images into GIF format through a desktop-based application developed using Python. The proposed method applies DCT to transform image data from the spatial domain into the frequency domain, enabling the reduction of high-frequency coefficients that contribute less to visual perception. The reconstructed image is subsequently processed using the Median Cut algorithm for color quantization to generate an optimized color palette suitable for the GIF format. The application was developed with a graphical user interface (GUI) using Tkinter to facilitate image selection, parameter configuration, processing visualization, and performance evaluation. Experimental testing was conducted using multiple color palette configurations (256, 128, 64, 32, and 8 colors). Performance was evaluated based on file size, compression ratio, reduction size, mean squared error (MSE), and peak signal-to-noise ratio (PSNR). The results demonstrate that the proposed approach effectively reduces image file size while preserving acceptable visual quality. A smaller color palette produces higher compression efficiency but increases reconstruction error, resulting in higher MSE and lower PSNR values. Overall, the combination of DCT and Median Cut provides an effective approach for JPG-to-GIF image conversion by achieving a balance between compression efficiency and image quality.

INTRODUCTION

The rapid advancement of digital technology has significantly increased the use of digital images across various fields, including multimedia, photography, social media, information systems, and web-based applications. As the resolution of digital cameras and mobile devices continues to improve, image file sizes have also increased, resulting in greater storage requirements and longer transmission times. Consequently, image compression has become an essential solution for reducing file size while preserving acceptable visual quality. Digital image processing employs various transformation and optimization techniques to generate more efficient image representations, thereby reducing storage and transmission resource requirements (Putra, 2021).

One of the most widely used image formats is JPEG (JPG), as it provides relatively small file sizes while maintaining good visual quality. This efficiency is achieved through a lossy compression mechanism that utilizes the Discrete Cosine Transform (DCT) to convert image data from the spatial domain into the frequency domain. This transformation concentrates most of the image energy in low-frequency coefficients, while high-frequency components, which have less impact on human visual perception, can be reduced through the quantization process. As a result, DCT remains one of the most widely adopted algorithms in modern digital image compression standards (Satria & Wijaya, 2020; Hidayat et al., 2020).

On the other hand, the Graphics Interchange Format (GIF) continues to play an important role in various digital applications, including icons, simple illustrations, short animations, digital stickers, and social media content. Unlike JPG, GIF supports a maximum of only 256 colors, making the conversion from JPG to GIF highly dependent on an effective color quantization technique capable of selecting a representative color palette without significantly degrading image quality. The choice of color quantization algorithm is therefore a critical factor influencing the visual quality of the converted image, as the original color distribution must be represented using a substantially smaller number of colors (Wahyuni & Prabowo, 2022; Dewantara, 2021).

One of the most commonly used color quantization algorithms is Median Cut. This algorithm recursively partitions the RGB color space according to the largest color range until the desired number of color clusters is obtained. The average color of each cluster is then used as the representative color in the final image. Compared with several other color quantization techniques, Median Cut offers relatively low computational complexity, generates representative color distributions, and is particularly suitable for GIF images due to their limited color palette (Febrianto & Hakim, 2021; Nuraini & Maulana, 2022).

Several previous studies have investigated the application of DCT and Median Cut independently. Satria and Wijaya (2020) demonstrated that DCT remains an effective method for frequency-domain image compression. Lestari et al. (2021) reported that the quantization level significantly affects the visual quality of compressed images. Meanwhile, Febrianto and Hakim (2021) found that the Median Cut algorithm produces a more stable color palette than simpler quantization methods. Khairunnisa et al. (2023) further showed that combining frequency-domain transformation with color quantization can improve the efficiency of digital image representation. Nevertheless, studies specifically integrating DCT and Median Cut for JPG-to-GIF image conversion within an interactive desktop application remain relatively limited.

Based on these considerations, this study implements the Discrete Cosine Transform (DCT) and Median Cut algorithms in a Python-based desktop application to convert JPG images into GIF format. DCT is employed as the image transformation and information reduction stage, while Median Cut is applied to generate an optimized color palette compatible with the characteristics of the GIF format. System performance is evaluated using several objective metrics, namely Compression Ratio, Reduction Size, Mean Squared Error (MSE), and Peak Signal-to-Noise Ratio (PSNR), with color palette variations of 256, 128, 64, 32, and 8 colors. Through this approach, the proposed method is expected to provide an effective solution for producing GIF images with smaller file sizes while maintaining acceptable visual quality, thereby contributing to the advancement of digital image processing applications based on frequency-domain transformation and color quantization techniques.

RESEARCH METHODS

This study employed a quantitative experimental approach to evaluate the performance of the Discrete Cosine Transform (DCT) and Median Cut algorithms in converting JPG images into GIF format. The research focused on measuring the effectiveness of the proposed method in reducing file size while maintaining acceptable image quality. The experimental approach enabled the evaluation of compression performance using objective image quality metrics under different color palette configurations.

The proposed system was implemented as a desktop application using the Python programming language with a graphical user interface (GUI) developed using the Tkinter library. Several supporting libraries were utilized during the implementation process, including Pillow (PIL) for image input/output operations, NumPy for numerical computation, SciPy for implementing the Discrete Cosine Transform (DCT) and Inverse Discrete Cosine Transform (IDCT), and Scikit-image for calculating image quality metrics such as Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR). The application was developed using Visual Studio Code and executed on a computer equipped with an Intel Core i5 processor, 8 GB RAM, and the Windows 11 operating system.

The input data consisted of digital images in JPG format with various image characteristics. Each image was processed through several sequential stages. First, the selected image was loaded into the application and converted into grayscale before being divided into non-overlapping blocks of 8 x 8 pixels. Each block was transformed from the spatial domain into the frequency domain using the two-dimensional Discrete Cosine Transform (DCT), expressed as :

$$F(u,v) = \frac{2}{\sqrt{NM}} C(u)C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x,y) \cos\left(\frac{(2x+1)u\pi}{2N}\right) \cos\left(\frac{(2y+1)v\pi}{2M}\right)$$

where (f(x,y)) represents the pixel intensity in the spatial domain, (F(u,v)) denotes the DCT coefficient in the frequency domain, and (\alpha(\cdot)) is the normalization factor. High-frequency coefficients were reduced to achieve compression, after which the image was reconstructed using the Inverse Discrete Cosine Transform (IDCT).

The reconstructed image was subsequently converted back to the RGB color space and processed using the Median Cut algorithm for color quantization. This algorithm recursively partitions the RGB color space according to the largest color range until the desired number

of color clusters is obtained. The average color of each partition is then used to generate the final color palette for the GIF image. To evaluate the influence of color reduction on compression performance, five palette sizes were tested, namely 256, 128, 64, 32, and 8 colors.

The performance of the proposed method was evaluated using both compression efficiency and image quality metrics. Compression efficiency was measured using Compression Ratio (CR) and Reduction Size (RS), while reconstruction quality was assessed using Mean Squared Error (MSE) and Peak Signal-to-Noise Ratio (PSNR). The MSE was calculated as :

$$MSE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [I(i,j) - K(i,j)]^2$$

where (I(i,j)) and (K(i,j)) represent the original and reconstructed image pixels, respectively. The PSNR was calculated using :

$$PSNR = 10 \log_{10} \left(\frac{255^2}{MSE} \right)$$

Finally, the experimental results obtained from different color palette configurations were compared quantitatively to analyze the trade-off between compression efficiency and visual image quality. The analysis was supported by visual comparisons between the original image, the DCT-processed image, and the final GIF image generated using the Median Cut algorithm.

RESULT AND DISCUSSION

System Implementation

A desktop-based image conversion application was successfully developed using Python with the Tkinter library to implement the proposed combination of the Discrete Cosine Transform (DCT) and Median Cut algorithms. The application enables users to select JPG images, specify the desired color palette size, execute the compression process, and save the converted image in GIF format. In addition, the application automatically calculates quantitative evaluation metrics, including file size, Compression Ratio (CR), Reduction Size (RS), Mean Squared Error (MSE), and Peak Signal-to-Noise Ratio (PSNR).

The main interface integrates the complete image processing workflow, including image loading, DCT transformation, color quantization using Median Cut, performance evaluation, and image visualization. This design allows users to compare the original image, the reconstructed image after DCT, and the final GIF image generated after color quantization. Figure 1 illustrates the main interface of the developed application.

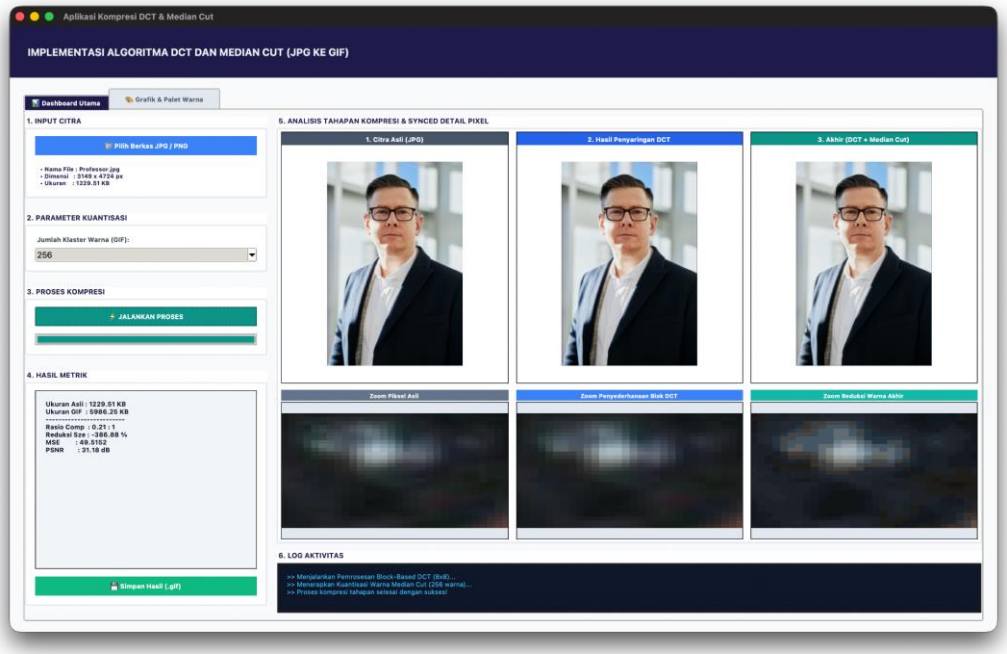

Figure 1. Main Interface of the Proposed Application

Visualization of Image Processing

The image processing begins by transforming the selected JPG image into the frequency domain using the Discrete Cosine Transform. The DCT concentrates most image information into low-frequency coefficients while suppressing less significant high-frequency components. After inverse transformation (IDCT), the reconstructed image maintains most visual characteristics of the original image while preparing the data for efficient color quantization.

The reconstructed image is subsequently processed using the Median Cut algorithm. This algorithm recursively partitions the RGB color space according to the largest color variance until the required number of color clusters is obtained. Each cluster is represented by its average color, producing an optimized color palette suitable for the GIF format.

The visualization demonstrates that the generated GIF image preserves the dominant colors and overall image structure despite using a limited number of colors.

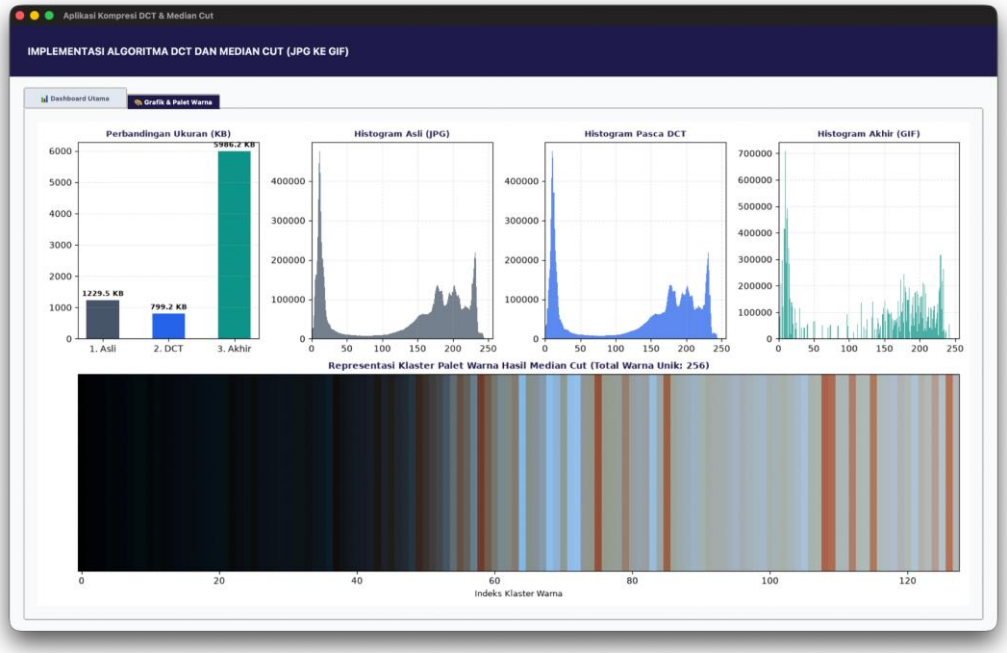

Figure 2. Image Processing Results Using DCT and Median Cut

Performance Evaluation

The proposed method was evaluated using several color palette configurations, namely 256, 128, 64, 32, and 8 colors. Each configuration produced different compression performance and image quality characteristics.

The evaluation employed five objective performance indicators:

- File Size
- Compression Ratio (CR)
- Reduction Size (RS)
- Mean Squared Error (MSE)
- Peak Signal-to-Noise Ratio (PSNR)

The experimental results indicate that reducing the number of colors significantly decreases the resulting GIF file size. Consequently, the compression ratio increases as the palette size becomes smaller. However, aggressive color reduction also introduces larger reconstruction errors, reflected by increasing MSE values and decreasing PSNR values.

This behavior demonstrates the expected trade-off between compression efficiency and image quality. Larger color palettes (e.g., 256 colors) preserve more image details but achieve lower compression efficiency, whereas smaller palettes (e.g., 8 colors) generate much smaller file sizes at the expense of visible color degradation.

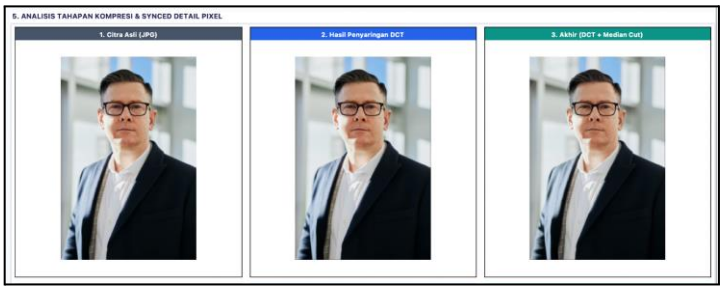

Figure 3. Comparison of Original Image, DCT Output, and Final GIF Image

Discussion

The experimental findings confirm that the proposed integration of the Discrete Cosine Transform and Median Cut algorithms effectively performs JPG-to-GIF image conversion while reducing storage requirements.

The DCT algorithm contributes by transforming image information into the frequency domain, allowing insignificant frequency components to be discarded without severely affecting perceptual image quality. This preprocessing stage simplifies the image representation before color quantization.

Subsequently, the Median Cut algorithm efficiently generates an optimized color palette that satisfies the 256-color limitation of the GIF format. The recursive partitioning strategy successfully preserves dominant image colors while reducing color redundancy.

The quantitative evaluation further demonstrates that increasing the compression level inevitably increases reconstruction error. This observation is consistent with the characteristics of lossy image compression, where higher compression generally results in reduced visual fidelity. Nevertheless, images processed using 256 and 128 colors still maintain satisfactory visual quality while achieving meaningful reductions in storage size.

Overall, the proposed method provides a practical balance between compression efficiency and reconstructed image quality. The developed application successfully integrates DCT-based frequency compression with Median Cut color quantization into an interactive desktop environment, making it suitable for educational purposes and practical image processing applications requiring efficient JPG-to-GIF conversion.

CONCLUSION

This study successfully implemented the Discrete Cosine Transform (DCT) and Median Cut algorithms in a Python-based desktop application for converting JPG images into GIF format. The developed application integrates image loading, DCT-based frequency transformation, color quantization using the Median Cut algorithm, performance evaluation, and visualization of processing results within a single graphical user interface. The implementation demonstrates that the proposed approach is capable of performing image conversion while providing users with both visual comparisons and quantitative evaluation metrics.

Experimental results indicate that the combination of DCT and Median Cut effectively reduces image file size while maintaining acceptable visual quality. The DCT algorithm efficiently transforms image information into the frequency domain, enabling data reduction by suppressing less significant high-frequency components before reconstruction.

Subsequently, the Median Cut algorithm generates an optimized color palette suitable for the GIF format by reducing the number of colors while preserving the dominant visual characteristics of the original image.

Performance evaluation using Compression Ratio (CR), Reduction Size (RS), Mean Squared Error (MSE), and Peak Signal-to-Noise Ratio (PSNR) confirms the expected trade-off between compression efficiency and image quality. Smaller color palettes produce higher compression ratios and smaller output file sizes but increase reconstruction error, resulting in higher MSE and lower PSNR values. Among the evaluated configurations, larger color palettes (such as 256 and 128 colors) provide a better balance between compression efficiency and visual quality, whereas lower palette sizes are more suitable for applications that prioritize storage efficiency over image fidelity.

Overall, the proposed method provides an effective and practical solution for JPG-to-GIF image conversion by balancing compression performance and visual quality, making it suitable for digital image processing applications that require efficient image storage and transmission.

REFERENCES

Amanati, N., Setyawan, W., & Sari, D. (2021). Implementasi Discrete Cosine Transform (DCT) pada kompresi citra digital untuk peningkatan efisiensi penyimpanan. Jurnal Teknologi Informasi dan Terapan, 9(2), 145–154.

Budiman, A., & Kurniawan, H. (2022). Analisis performa algoritma kompresi citra berbasis transformasi frekuensi. Jurnal Sains Komputer Indonesia, 7(1), 23–32.

Dewi, R., & Pratama, F. (2021). Optimasi pengolahan citra menggunakan metode Discrete Cosine Transform pada media digital. Jurnal Informatika dan Komputer, 10(3), 189–198.

Gifani, R. (2023). An improved median cut color quantization for efficient GIF encoding. International Journal of Image Processing, 17(1), 55–64.

Halim, C., & Wahono, R. (2020). Studi komparatif teknik kuantisasi warna dalam pembuatan GIF. Jurnal Sistem dan Teknologi Informasi, 8(4), 301–312.

Kusuma, Y., & Lestari, D. (2021). Evaluasi kualitas kompresi citra menggunakan metrik PSNR dan MSE pada berbagai algoritma transformasi. Jurnal Multimedia dan Komputer, 5(2), 77–86.

Putra, I., & Santoso, A. (2022). Penerapan algoritma median cut pada proses kuantisasi warna citra digital berformat GIF. Journal of Digital Imaging Technology, 4(1), 11–22.

Rahmadi, B., & Nugroho, S. (2023). Analisis format JPG dan GIF pada sistem penyimpanan multimedia modern. Jurnal Teknologi Media Digital, 6(1), 41–52.

Sundari, M., & Riyadi, R. (2020). Pengaruh jumlah warna terhadap kualitas hasil konversi GIF pada citra digital. Jurnal Informatika Nusantara, 3(2), 98–108.

Wijaya, R., & Handayani, T. (2022). Penerapan transformasi DCT pada pemrosesan citra untuk efisiensi kompresi. Jurnal Rekayasa Elektronika dan Informatika, 12(1), 34–43.

Zhafira, N., & Malik, A. (2021). Review perkembangan metode kompresi citra berbasis transformasi dari tahun 2015–2020. Jurnal Ilmu Komputer Terapan, 9(2), 120–134.

Ardiansah, G., dkk. (2024). Penggunaan bahasa pemrograman Python dalam pembelajaran kalkulus fungsi dua variabel. Algoritma. https://doi.org/10.62383/algoritma.v2i3.67.

Kadarina, T. M., & Hajar, M. H. I. (2024). Edukasi pengenalan bahasa pemrograman Python pada SMA Methodist Binjai. Cendekia: Jurnal Pengabdian Masyarakat. https://doi.org/10.31004/cdj.v5i4.30999.

Wilyani, T., et al. (2024). Pengenalan dasar pemrograman Python dengan Google Colaboratory. Jurnal Pelayanan dan Pengabdian Masyarakat Indonesia. https://doi.org/10.55606/jppmi.v3i1.108.