World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(RESEARCH ARTICLE)

Design of GUI-based digital image processing applications in python using OpenCV and Tkinter

Andysah Putera Utama Siahaan *, Afif Yasri, Ramlan Marbun, Muhammad Daffa Alfikri and Khairil Putra

Information Technology, Pembangunan Panca Budi University, Medan, Indonesia.

World Journal of Advanced Engineering Technology and Sciences, 2025, 16(01), 049-055

Publication history: Received on 20 June 2025; revised on 01 July 2025; accepted on 04 July 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.16.1.1196

Abstract

Digital image processing is an important field in computer science that enables the manipulation and analysis of visual data for various practical applications. This study presents the development of an interactive desktop application for digital image processing using Python, OpenCV, and Tkinter. The purpose of this research is to provide a user-friendly graphical interface that simplifies the use of common image processing techniques, especially for non-technical users. The application includes features such as image upload, grayscale conversion, binary thresholding, brightness adjustment, logical AND operation, image sharpening, dilation, RGB histogram visualization, image reset, and result saving. The prototyping development model was adopted to ensure iterative refinement based on functionality and usability. The core methods were implemented using the OpenCV library for image processing operations, while Tkinter was used to design the GUI components. Experimental results demonstrated that the application successfully processed digital images and displayed both original and processed outputs side by side. Each feature functioned as expected and contributed to enhancing image quality or extracting relevant information. The integration of a visual interface with underlying technical operations enables users to interact more intuitively with image processing tasks. This application offers a practical tool for education, experimentation, and further development in the field of digital image processing.

Keywords: Image Processing; OPENCV; Application Gui; Python

1. Introduction

Digital image processing is one of the branches of computer science that has made a significant contribution to the development of modern information technology [1]. This technique allows the processing of visual information in the form of digital images for various purposes, such as object identification, pattern analysis, and image quality improvement [2] . The need for image processing technology is increasing along with advances in various sectors, such as healthcare, agriculture, manufacturing, transportation, to security and defense [3].

The rapid development of software and hardware has prompted researchers to continue to explore new approaches to processing digital imagery [4]. One widely used approach is to utilize the Python programming language, which is known to be flexible and supported by a large community [5]. Python has many supporting libraries for image processing, one of which is OpenCV (Open Source Computer Vision Library), which has been widely used in image and video processing due to its completeness of features and ability to process visual data in real-time [6].

However, the use of image processing libraries is often limited to text-based or command-line environments, which are not friendly to non-technical users (Scott, 2023). This is a separate obstacle in the learning process and implementation at the end-user level. Therefore, an approach is needed that is able to bridge technical functions with the need for more

* Corresponding author: Andysah Putera Utama Siahaan

Copyright © 2025 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0.

intuitive user interaction [8]. One of the solutions offered is the use of a graphical user interface that is able to display the processing results visually and provide direct control over the processes carried out.

In the Python ecosystem, Tkinter is a simple but powerful graphical interface library for building visual interactions. The combination of Tkinter and OpenCV allows users to carry out various image processing operations such as color conversion, rotation, filtering, and object detection directly through an easy-to-understand visual display [9]. This kind of interaction not only simplifies the process of experimentation, but also speeds up the process of understanding the basic principles of image processing.

Previous studies have shown that the use of visual methods in image data processing can increase work effectiveness and encourage collaboration between technical and non-technical users [10]. For example, in educational and research environments, the presence of a visual interface integrated with an image processing library has been proven to increase efficiency in the learning process as well as system modeling [11]. Nonetheless, there is still little development that integrates OpenCV's technical capabilities with an interface designed specifically for interactive purposes.

Based on these problems, this study aims to develop an interactive solution in digital image processing that combines OpenCV and Python-based Tkinter libraries. This solution is expected to be an effective means of supporting the exploration and implementation of basic and advanced techniques in digital image processing, and can be accessed by various users, both technical and non-technical.

2. Literature review

Digital image processing is a dynamic and rapidly evolving field within the realms of computer science and engineering. It encompasses a wide range of techniques and methodologies aimed at enhancing, analyzing, and manipulating digital images. The significance of digital image processing has grown substantially with advancements in technology, leading to its application across various domains such as medical imaging, security surveillance, and multimedia content creation. For instance, in the medical field, digital image processing techniques are employed to improve the quality of images obtained from MRI or CT scans, facilitating better diagnosis and treatment planning. In security, image processing algorithms are used for facial recognition and motion detection, enhancing surveillance systems. Overall, digital image processing plays a crucial role in transforming raw image data into meaningful information.

Python has emerged as one of the most popular programming languages for image processing tasks, primarily due to its user-friendly syntax and the extensive range of libraries available for various applications. Among these libraries, OpenCV (Open-Source Computer Vision Library) stands out as one of the most widely utilized tools for image processing and computer vision tasks. OpenCV offers a comprehensive suite of functions that facilitate a variety of image processing operations, including but not limited to edge detection, face recognition, image filtering, and video analysis. The library is designed to be efficient and optimized for real-time applications, making it suitable for both academic research and commercial projects. Its versatility allows developers to implement complex algorithms with relative ease, thereby accelerating the development process.

Tkinter serves as the standard library for creating graphical user interfaces (GUIs) in Python. It provides a robust framework that enables developers to design and implement interactive desktop applications with minimal effort. Tkinter's simplicity and ease of use make it an ideal choice for developers who wish to create user-friendly interfaces for their applications. By integrating Tkinter with OpenCV, developers can build image processing applications that not only perform complex image manipulations but also present a visually appealing and intuitive interface for users. This combination allows for the development of applications that can handle tasks such as image editing, real-time video processing, and object detection, all while ensuring that the user experience remains seamless and engaging.

3. Existing System

The existing systems for digital image processing often rely on standalone software applications that provide limited functionality and require significant technical expertise to operate. Many of these applications are designed for specific tasks, such as image editing or analysis, and may not offer a comprehensive suite of tools for users. Additionally, these systems can be expensive and may not be accessible to all users, particularly in educational or research settings. As a result, users often face challenges in integrating various tools and workflows, leading to inefficiencies and a steep learning curve.

Furthermore, traditional image processing systems typically lack user-friendly interfaces, making it difficult for non-technical users to engage with the software effectively. Many applications require users to input complex commands or navigate through intricate menus, which can be daunting for those without a background in programming or image processing. This barrier to entry limits the potential user base and hinders the widespread adoption of advanced image processing techniques. Consequently, there is a growing demand for more accessible, intuitive solutions that combine powerful image processing capabilities with easy-to-use graphical interfaces, enabling a broader audience to leverage the benefits of digital image processing technology.

4. Proposed System

The proposed system aims to develop an interactive digital image processing application that seamlessly integrates the capabilities of OpenCV with the user-friendly interface provided by the Tkinter library in Python. This application is designed to cater to both technical and non-technical users, enabling them to explore and implement various image processing techniques without the need for extensive programming knowledge. By providing a graphical user interface (GUI), the system will facilitate intuitive interactions, allowing users to perform tasks such as image uploading, processing, and visualization of results in a straightforward manner.

The application will feature a range of image processing techniques, including grayscale conversion, binarization, brightness adjustment, logical AND operations, image sharpening, and morphological dilation. Users will be able to upload images and apply these techniques with just a few clicks. The interface will display the original and processed images side by side, providing immediate visual feedback on the effects of the applied processing methods. Additionally, the application will include functionality to display the RGB histogram of the processed image, allowing users to analyze the color distribution and make informed decisions regarding further adjustments.

To ensure a responsive and efficient user experience, the application will be developed using a multithreaded approach, allowing image processing tasks to run concurrently without freezing the interface. The design will incorporate a gradient background and will be responsive to window resizing, enhancing the overall aesthetic and usability of the application. Key features will include buttons for uploading images, dropdown menus for selecting processing methods, and options to process and save images. The RGB histogram will be displayed in a separate popup window, ensuring that users can focus on the main interface while still accessing important analytical tools.

The implementation process will involve careful interface design, coding of the various image processing functions, and integration of interactive visualization features. By leveraging Python libraries such as OpenCV for image processing, NumPy for matrix operations, Pillow (PIL) for image manipulation and display, and Matplotlib for histogram visualization, the proposed system aims to create a comprehensive and user-friendly platform for digital image processing. This research is expected to contribute significantly to the development of accessible image processing systems and serve as a foundation for future advancements in the field.

5. Methodology

This study employs a software engineering approach using a prototyping-based development model, aiming to design and build a desktop-based digital image processing application using the Python programming language and the Tkinter user interface. The application enables users to upload images and apply various image processing techniques such as grayscale conversion, binarization, brightness adjustment, logical AND operation, image sharpening, and morphological dilation. It displays the original and processed images side by side. Additionally, the application provides features to display the RGB histogram of the processed image and to save the resulting image.

The system is developed using Python libraries such as OpenCV for image processing, NumPy for matrix operations, Pillow (PIL) for image manipulation and interface display, and Matplotlib for displaying color histograms. The application interface includes a gradient background using the gradient image method and is designed to be responsive to window resizing. Key features are integrated into a Tkinter-based interface, including buttons to upload images, dropdown menus for selecting processing methods, buttons to process and save images, and RGB histogram displays that appear in a separate popup window. Input and output images are resized for display purposes, and all processing tasks are executed in a multithreaded manner to maintain interface responsiveness.

The main data used in this study are digital images uploaded by users. The implementation process involves interface design, coding of processing functions, and integration of interactive visualization features. This research is expected to

contribute to the development of interactive and user-friendly image processing systems and serve as a foundation for further research in the field of digital image processing.

6. Results and Discussion

The development was successfully carried out using the Python programming language with the support of the OpenCV library for image processing functions and Tkinter for the creation of a graphical user interface (GUI).

6.1. Grayscale

The application is capable of converting color images into grayscale images using the cv2.cvtColor() function of OpenCV. This feature is useful as an initial stage in further image processing such as binarization or edge detection.

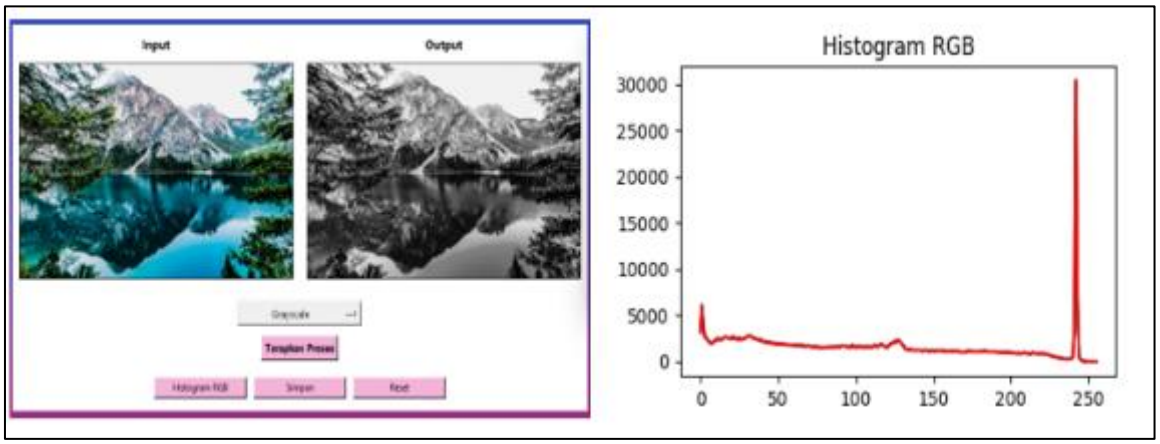

Figure 1 Test results with Grayscale

6.2. Thresholding

The thresholding feature in this application utilizes the Otsu method to automatically divide the image into two regions-black and white-by determining the optimal threshold value that minimizes intra-class variance. This process begins by converting the original color image to grayscale using the cv2.cvtColor() function, as the Otsu method operates on single-channel images. This feature is especially useful for object or document segmentation, such as separating text from the background. Additionally, the application displays the RGB histogram of the original color image in a separate popup window, providing a visual representation of the intensity distribution of red, green, and blue channels. This histogram helps users better understand the image's color composition and contrast levels, which can influence the effectiveness of the segmentation process.

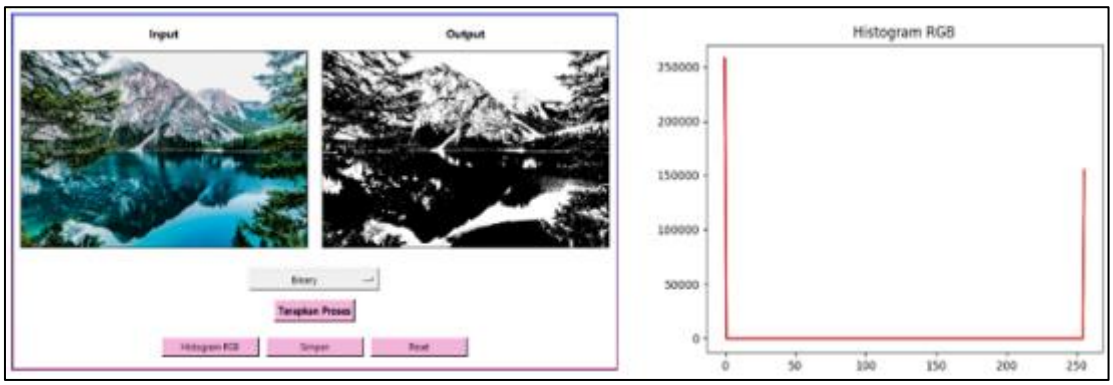

Figure 2 Thresholding Test Results

6.3. Brightness

The system allows the addition of brightness to the image with a fixed value. This feature helps to improve visibility in low-light images, although it's worth noting that excessive brightness can take away from detail.

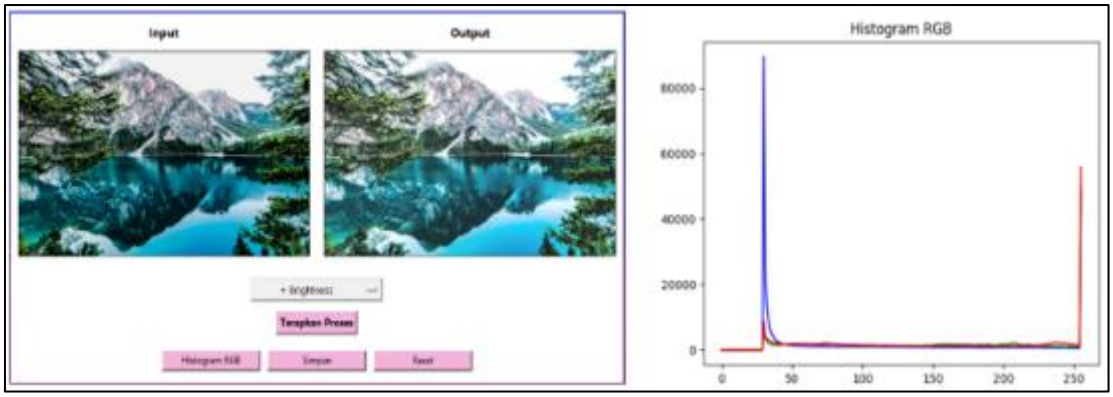

Figure 3 Brightness Test Results

6.4. AND Operations

AND operation is a binary logic operation performed on two binary or grayscale images. Each pixel of the first image is compared to the corresponding pixel in the second image, and the result follows the rules of AND logic.

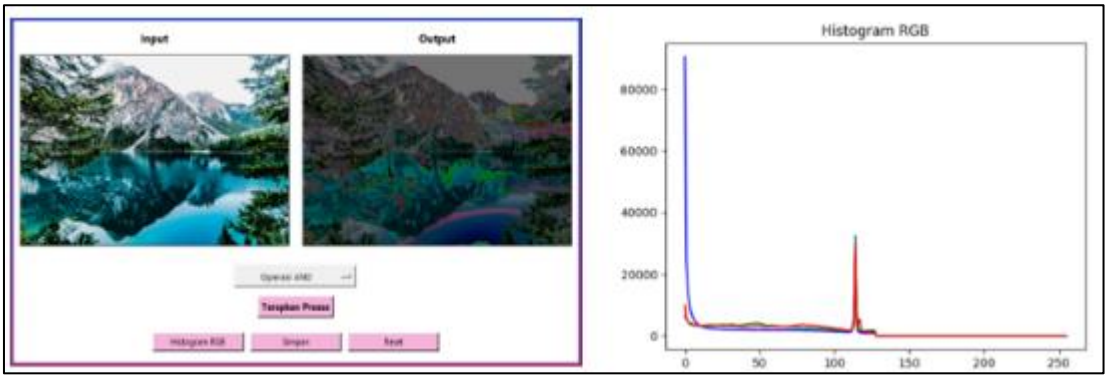

Figure 4 AND Operation Test Results

6.5. Sharpen

The sharpen feature in digital image processing is designed to enhance image clarity by emphasizing the edges or contours of objects within the image. This is achieved through a mathematical operation known as convolution, where the original image is processed using a special kernel or filter matrix. The kernel typically contains a high positive value at the center and negative values surrounding it, allowing it to highlight intensity differences between adjacent pixels.

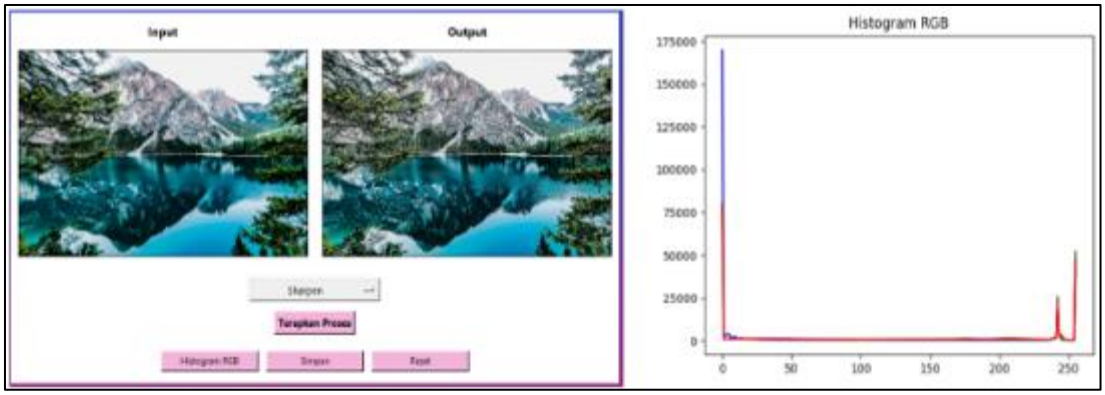

Figure 5 Image Sharpening Test Results

6.6. Dilation

The morphological operation of dilation successfully enlarges the white region of the binary image. This feature is useful for reducing noise and connecting disconnected parts of objects.

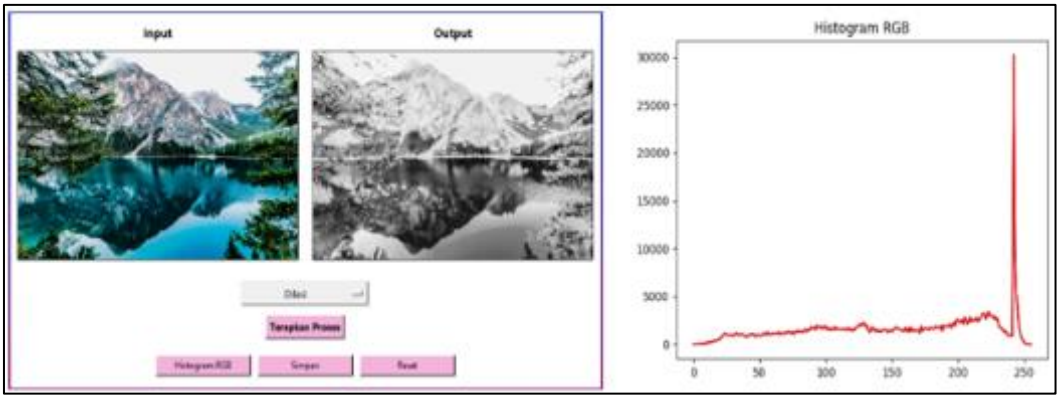

Figure 6 Dilation Test Results

7. Conclusion

This research successfully developed an interactive GUI application for digital image processing by utilizing Python, OpenCV, and Tkinter. The resulting application is able to perform various basic image processing functions such as conversion to grayscale, binerization, brightness adjustment, AND logic operations, image sharpening, and morphological dilation well. Each feature provides results that conform to the principles of digital image processing, demonstrated through visual testing that supports the analysis of the respective function. In addition, the app also provides an RGB histogram display as well as a result storage feature, which adds practical value to the image exploration process. The successful integration between technical libraries such as OpenCV and Tkinter's visual interface proves that the development of a user-friendly image processing system is possible, even for non-technical users. Therefore, these applications are not only relevant for educational purposes, but also have the potential to be further developed in research and industrial applications that require digital image analysis.

Compliance with ethical standards

Disclosure of conflict of interest

There is no conflict of interest.

References

[1] J. Jumadi, Y. Yupianti, and D. Sartika, “Pengolahan Citra Digital Untuk Identifikasi Objek Menggunakan Metode Hierarchical Agglomerative Clustering,” JST (Jurnal Sains dan Teknol., vol. 10, no. 2, pp. 148–156, 2021.

[2] A. P. U. Siahaan, “Image Similarity Test Using Eigenfaace Calculation,” vol. 3, no. 6, 2017.

[3] S. Mustakim et al., PENGOLAHAN CITRA DIGITAL. Cendikia Mulia Mandiri, 2025.

[4] G. S. Mahendra et al., Tren Teknologi AI: Pengantar, Teori, dan Contoh Penerapan Artificial Intelligence di Berbagai Bidang. PT. Sonpedia Publishing Indonesia, 2024.

[5] D. Abdullah et al., “Application of interpolation image by using bi-cubic algorithm,” in Journal of Physics: Conference Series, IOP Publishing, 2018, p. 12066.

[6] M. R. Qisthiano and A. O. Pratiwi, “Deteksi Tepi Pada Citra Objek Benda Menggunakan Algoritma Sobel Dan Prewitt Dengan Python,” J. Inform. dan Tek. Elektro Terap., vol. 13, no. 2, 2025.

[7] D. Satria, Pengantar Teknik Komputer: Konsep dan Prinsip Dasar. PT. Sonpedia Publishing Indonesia, 2023.

[8] C. Hodijah, N. W. M. N. Yani, and S. E. Mohamad Sajili, Komunikasi Bisnis dalam Era Artificial Intelligence. Takaza Innovatix Labs, 2025.

[9] A. I. Warnilah, H. Sutisna, A. Jaya-Mulyana, F. Siti-Nuraeni, and T. Aninditya-Widianto, "Program aplikasi pendeteksi masker dengan menggunakan algoritma Haarcascade," EVOLUSI J. Sains dan Manaj., vol. 10, no. 1, 2022.

[10] L. Judijanto et al., Sistem Informasi: Teori dan Penerapannya di Berbagai Bidang. PT. Sonpedia Publishing Indonesia, 2025.

[11] W. N. Afifah and V. Lusiana, "KLASIFIKASI JENIS BATIK SEMARANGAN MENGGUNAKAN METODE CONVOLUTION NEURAL NETWORK (CNN)," JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform., vol. 10, no. 1, pp. 542–553, 2025.