REPORT

Assignment 3 – A Practical Application with a Graphical User Interface (GUI).

Name : Dubba Mahathi

Roll number : 22671A7370

JBIET

AIML - B

Subject name : Computer Vision

Summary :

This project presents an **Interactive Image Processing Toolkit** built with Streamlit and OpenCV. It allows users to perform **color conversions, transformations, filtering, enhancement, edge detection, and compression** with real-time visualization. The toolkit, along with a Jupyter notebook and detailed report, provides both theoretical insights and practical demonstrations for learning fundamental image processing concepts.

Introduction :

Computer Vision is a rapidly growing field that enables machines to interpret and process visual information. One of its foundations is digital image processing, which involves manipulating images to improve quality, extract features, and prepare them for advanced tasks such as recognition and classification.
This project presents an Interactive Image Processing Toolkit developed using Streamlit and OpenCV. The toolkit demonstrates essential concepts of computer vision, including color space conversions, geometric transformations, filtering, morphological operations, image enhancement, edge detection, and compression. By combining theoretical notes with practical implementation, the project provides a strong foundation for understanding and applying core image processing techniques.

Theoretical Background :

Computer Vision is built upon the principles of digital image processing, which involves acquiring, representing, and manipulating images to extract meaningful information. The following theoretical concepts provide the foundation for the operations implemented in this project :

1. CMOS vs CCD Sensors
Image formation begins at the hardware level with sensors. CCD (Charge-Coupled Device) sensors have traditionally been used in applications requiring high-quality images and low noise. hey transfer charge across the sensor array to a single output node, producing cleaner signals but consuming more power. TOn the other hand, CMOS (Complementary Metal-Oxide-Semiconductor) sensors use multiple amplifiers at each pixel, resulting in lower cost, reduced power consumption, and higher speed. This makes CMOS sensors the preferred choice in most modern cameras and mobile devices, even though CCDs are still used in scientific and medical imaging.

2. Sampling and Quantization

When an image is captured, it exists in a continuous form. To process it digitally, two steps are required: sampling and quantization.

- Sampling divides the image into a discrete grid of pixels, determining spatial resolution. A higher sampling rate captures more detail, while a lower rate may cause aliasing.
- Quantization assigns intensity or color values to each sampled pixel. The number of quantization levels (e.g., 8-bit = 256 levels) directly affects image quality. Poor quantization leads to banding and loss of detail. Together, sampling and quantization convert a real-world scene into a digital format suitable for computer processing.

3. Point Spread Function (PSF)

Every imaging system has inherent imperfections that cause blurring. The Point Spread Function (PSF) mathematically represents how a single point of light is spread in the final image. A narrow PSF corresponds to a sharper system, while a wider PSF indicates blur due to lens aberrations, motion, or diffraction. Understanding PSF is important in image restoration, deblurring, and super-resolution, as it provides a model of how images degrade during acquisition.

4. Relevance to Computer Vision

These concepts are fundamental because they define how images are captured, represented, and degraded before any higher-level computer vision task such as recognition, segmentation, or classification can be performed. By studying CMOS vs CCD, Sampling & Quantization, and PSF, one gains insight into the origin and limitations of digital images, which is crucial for implementing effective processing algorithms.

Implementation :

The project was implemented as an Interactive Image Processing Toolkit using Streamlit for the graphical user interface and OpenCV for image processing operations. Streamlit provides an intuitive and interactive web-based environment where users can upload images, choose operations from a sidebar, adjust parameters with sliders or dropdowns, and instantly visualize results.

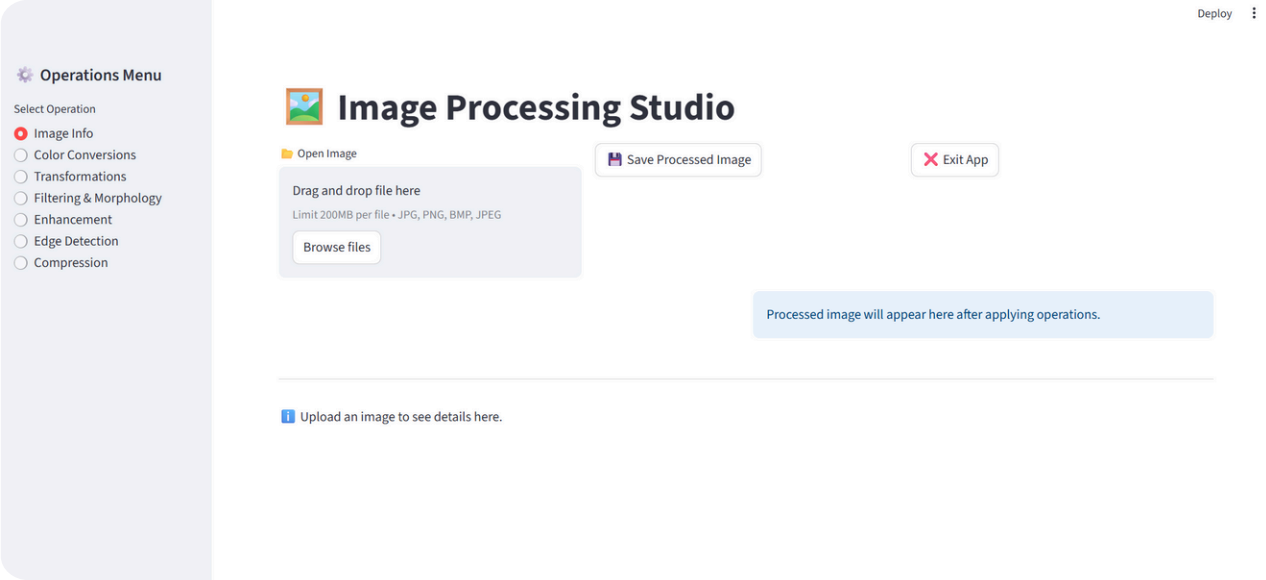
Toolkit Workflow :

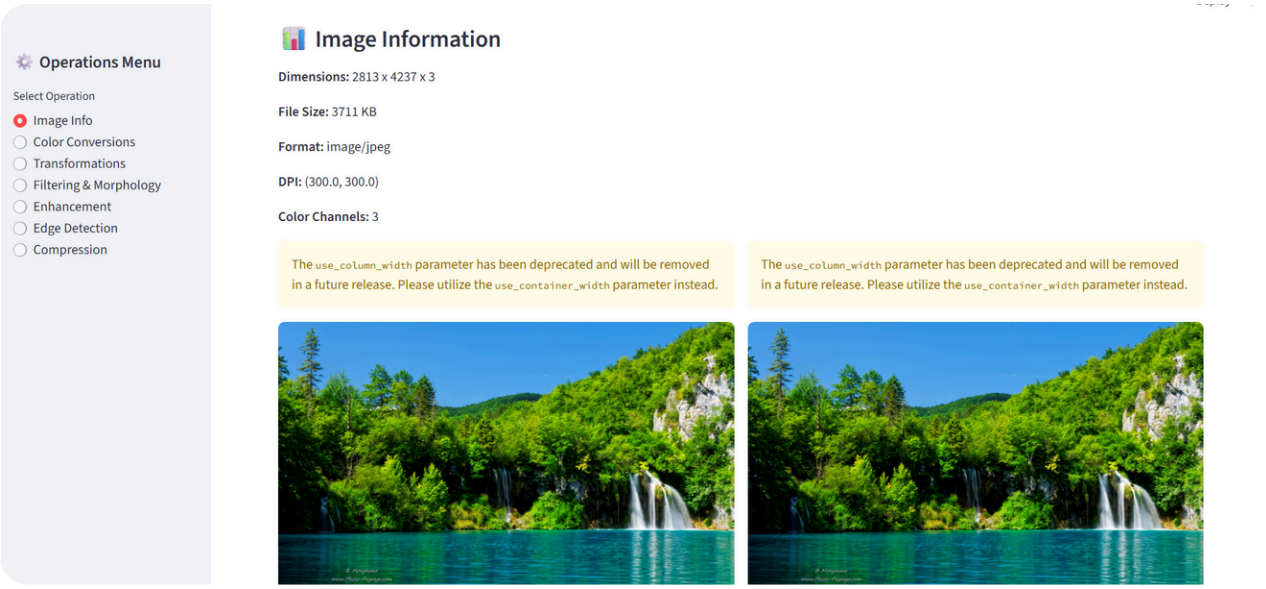
1. Image Upload – Users upload an image through the Streamlit interface.
2. Operation Selection – The sidebar presents multiple categories of image processing operations.
3. Parameter Adjustment – Interactive widgets (sliders, checkboxes, dropdowns) allow tuning parameters such as kernel size, scale factor, or thresholds.
4. Real-Time Output – Processed images are displayed alongside the original for direct comparison.
5. Save Option – Final results can be saved in formats such as JPG, PNG, or BMP.

Features Implemented :

1.Image Info :

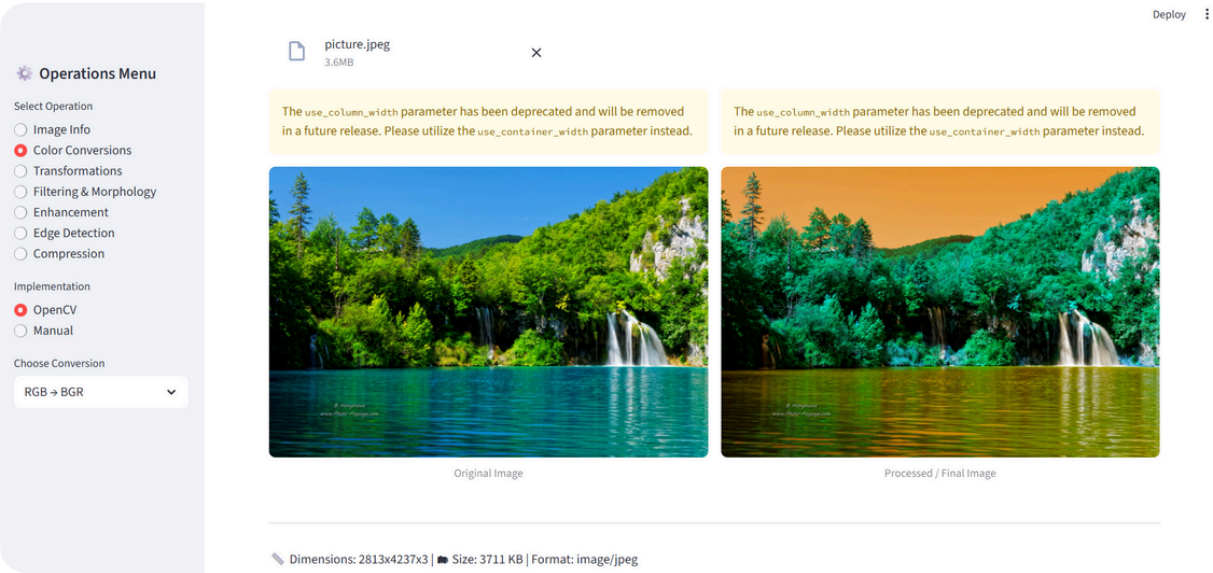
- Show resolution, shape, DPI, file format, color channels

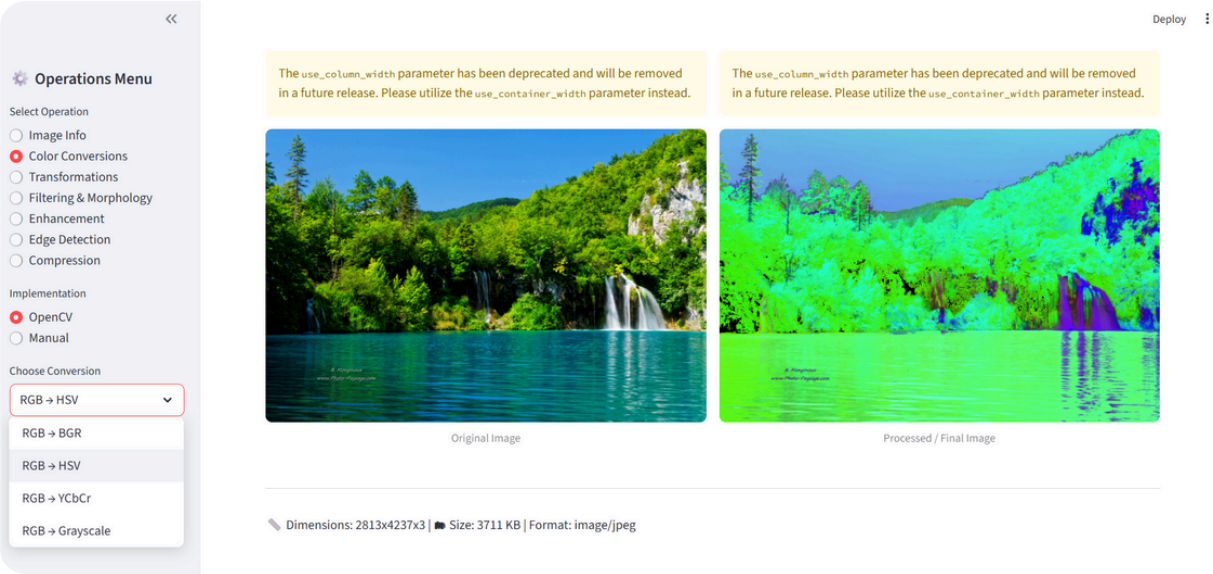




2. Color Conversion :

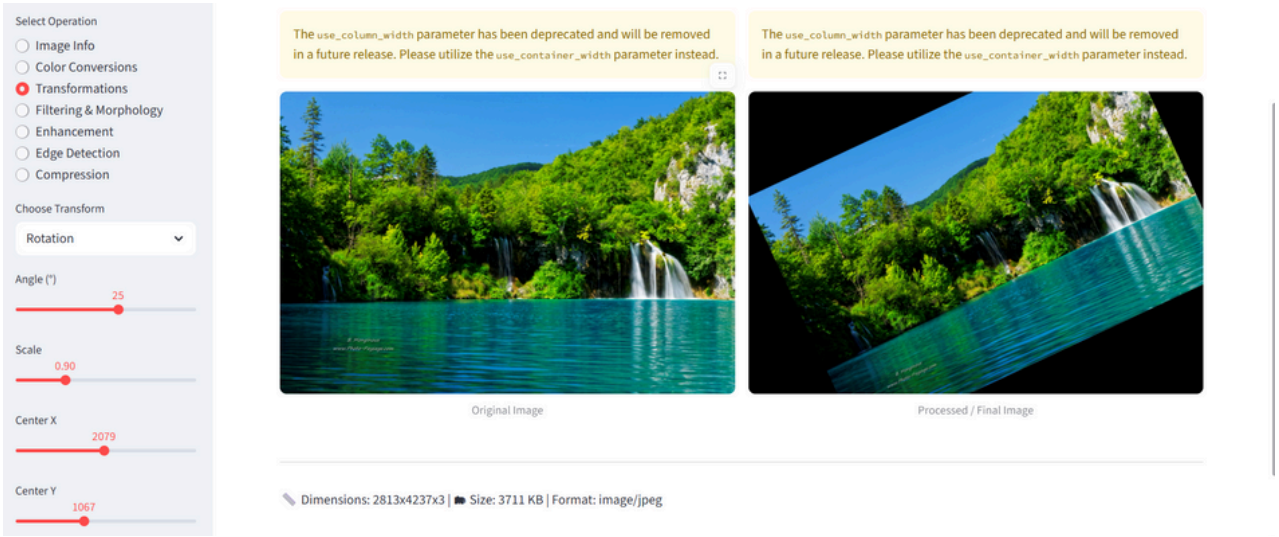
- RGB ↔ BGR
- RGB ↔ HSV
- RGB ↔ YCbCr
- RGB ↔ Grayscale



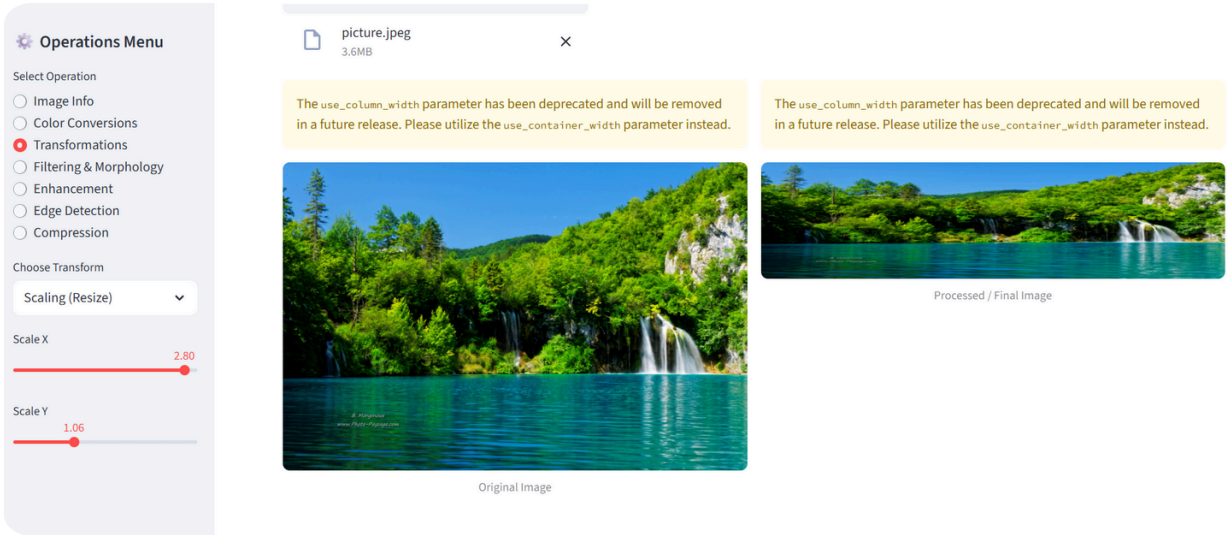


3 Transformations :

- Rotation
- Scaling (up/down)
- Translation (shift)
- Affine Transform
- Perspective Transform

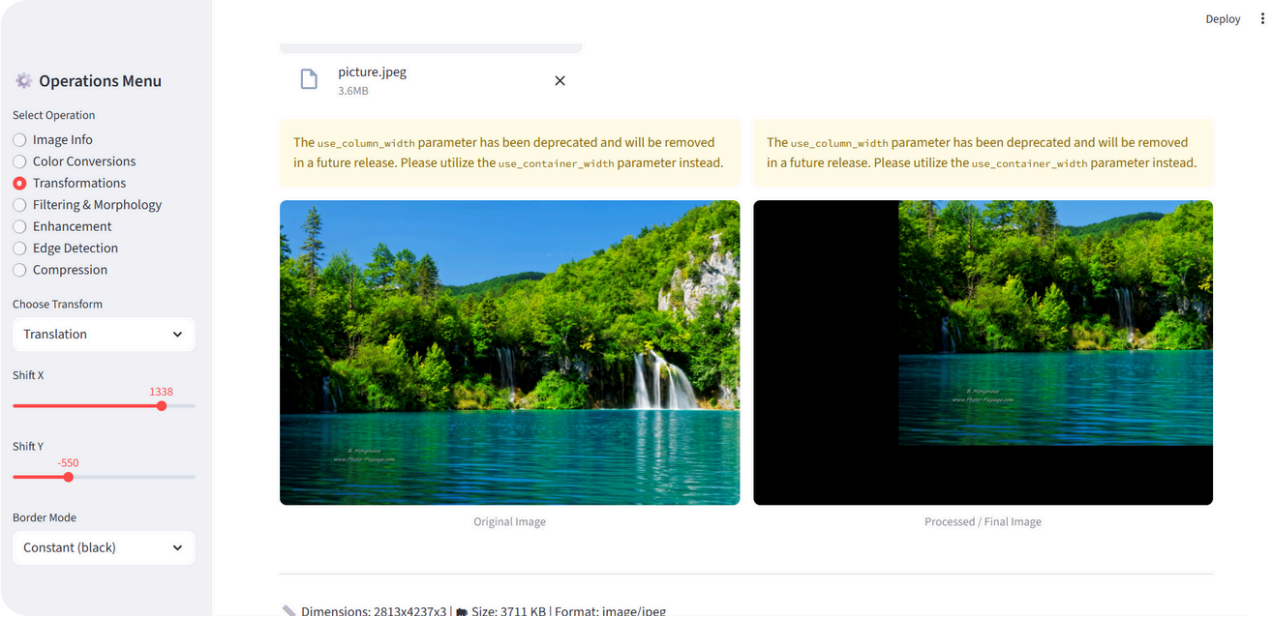


Rotation

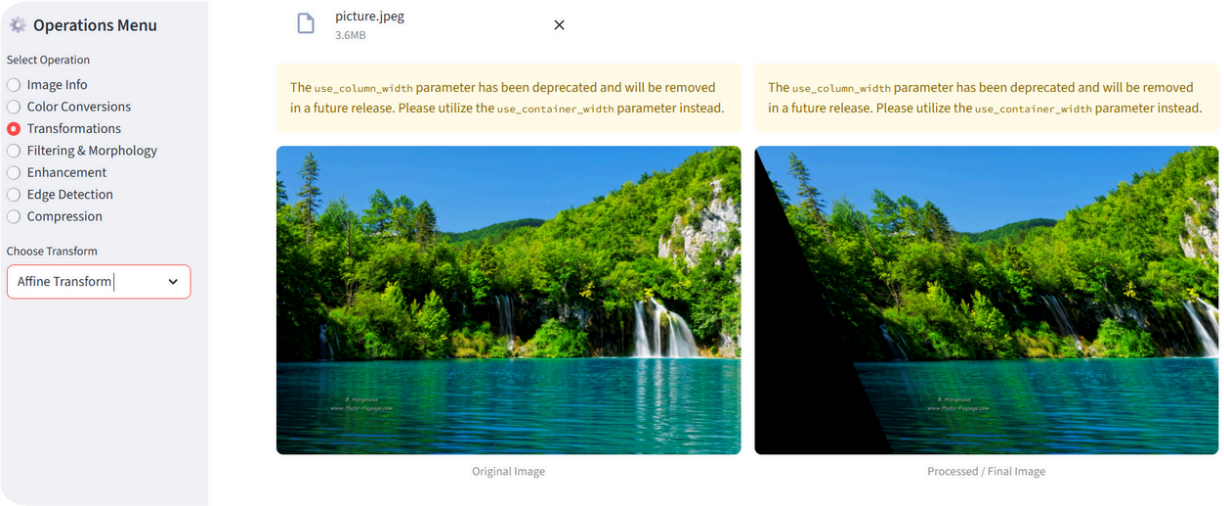


Scaling (Resize)

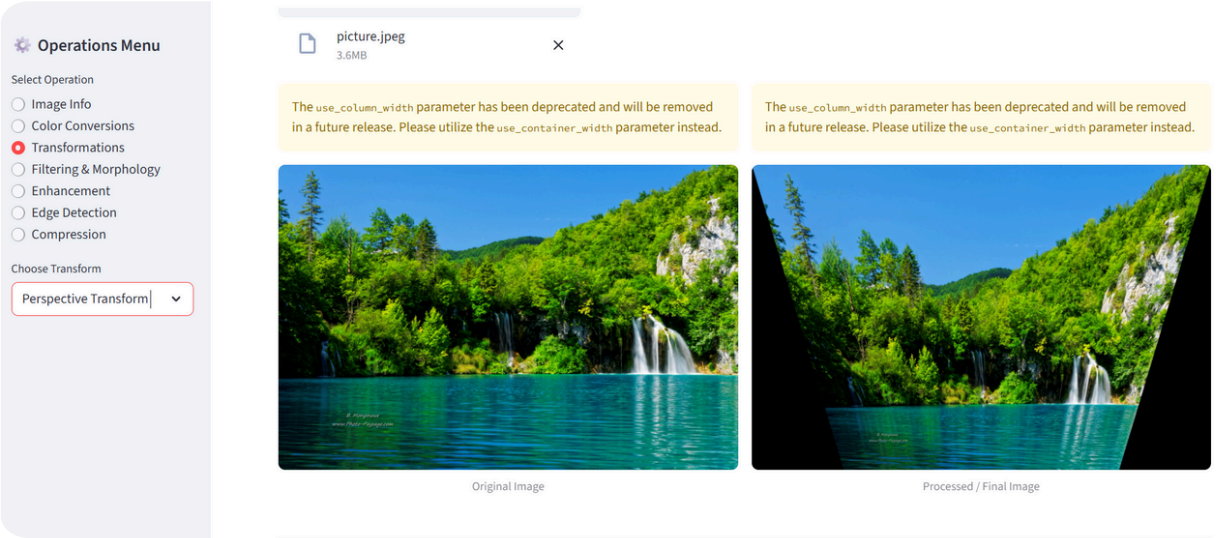
Translation (shift)



Affine Transform

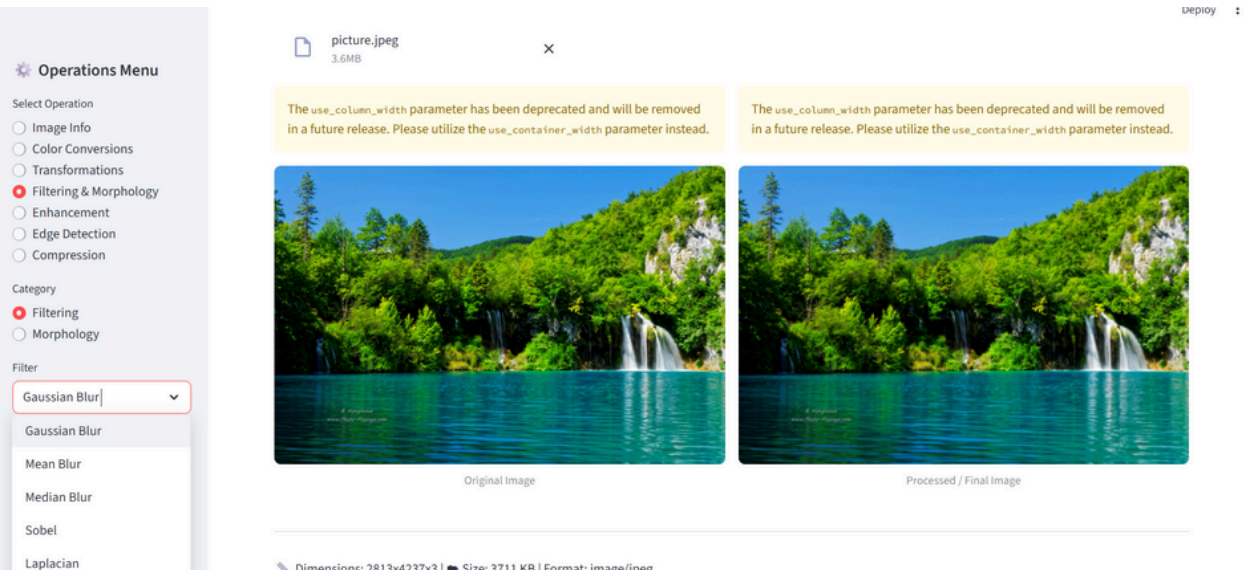


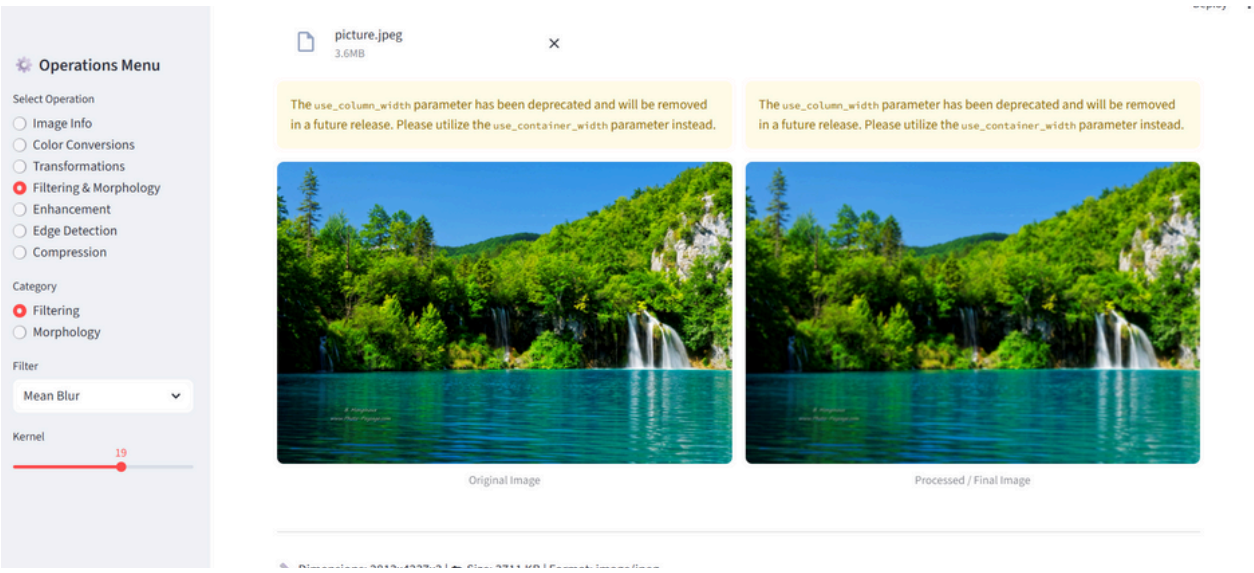
Perspective Transform

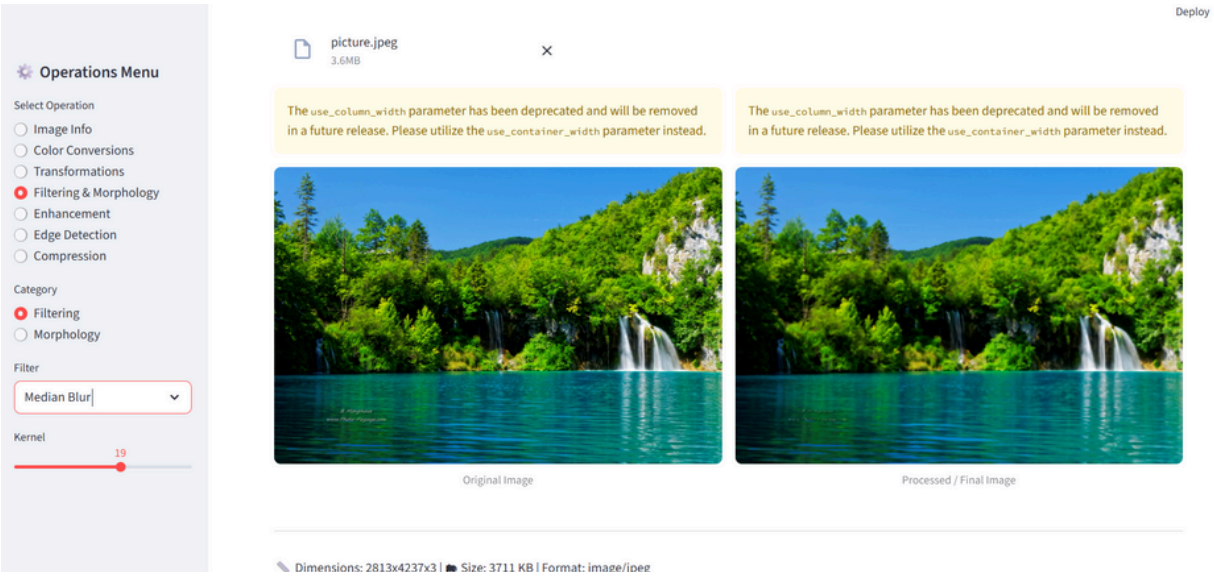


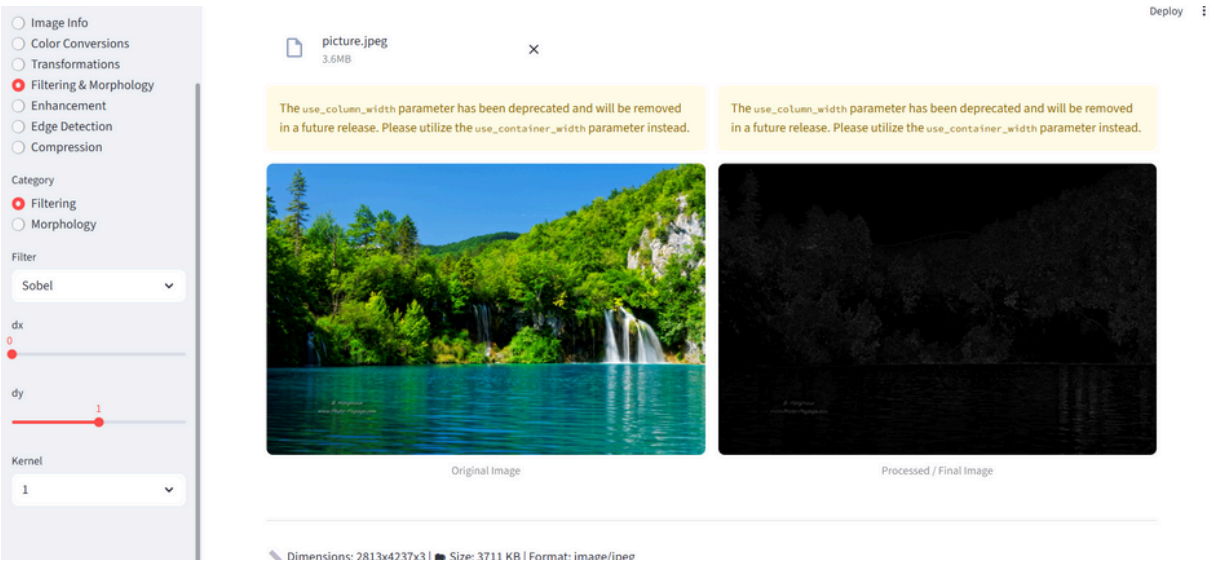
4.Filtering & Morphology :

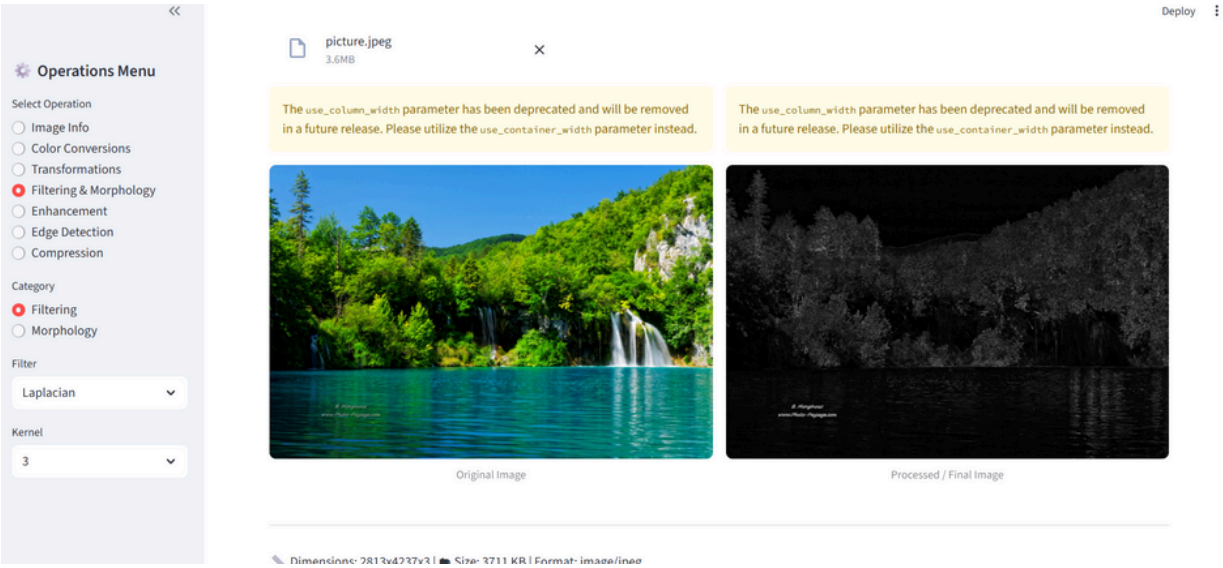
- Gaussian / Mean / Median filter
- Sobel / Laplacian edge filters
- Dilation, Erosion
- Opening, Closing





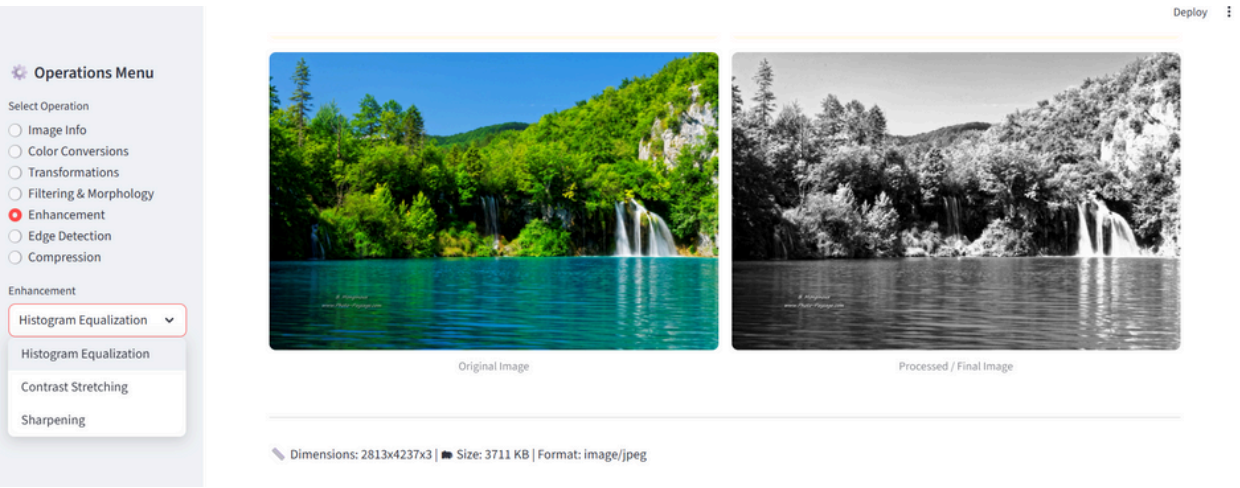

Deploy
Operations Menu
Select Operation
Image Info
Color Conversions
Transformations
Filtering & Morphology
Enhancement
Edge Detection
Compression
Category
Filtering
Morphology
Filter
Median Blur
Kernel
19
picture.jpeg
3.6MB
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
Original Image
Processed / Final Image
Dimensions: 2813x4237x3 | Size: 3711 KB | Format: image/jpeg

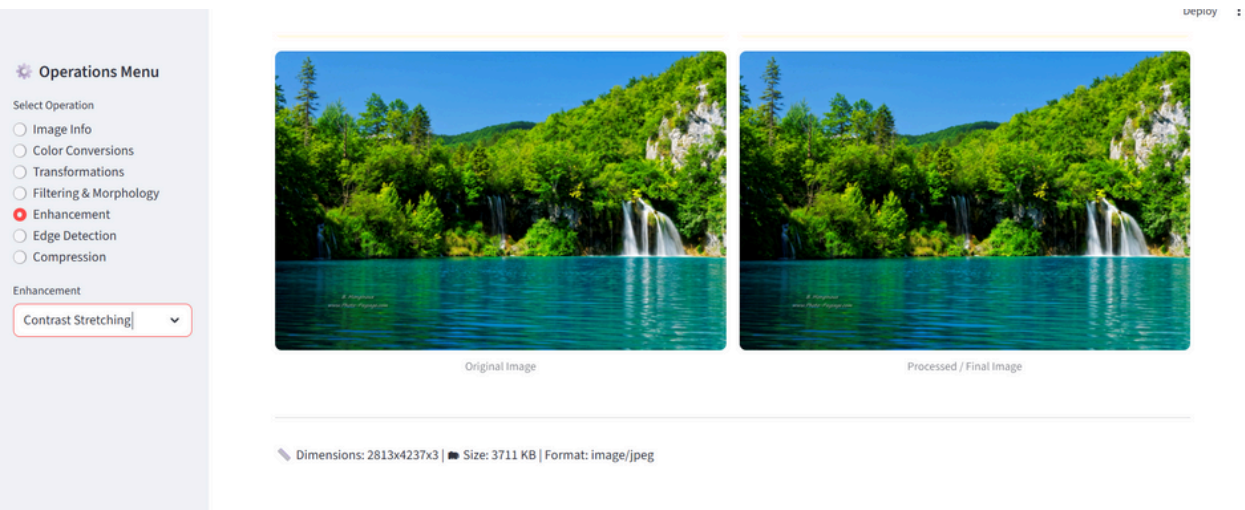

Deploy
Image Info
Color Conversions
Transformations
Filtering & Morphology
Enhancement
Edge Detection
Compression
Category
Filtering
Morphology
Filter
Sobel
dx
0
dy
1
Kernel
1
picture.jpeg
3.6MB
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
Original Image
Processed / Final Image
Dimensions: 2813x4237x3 | Size: 3711 KB | Format: image/jpeg

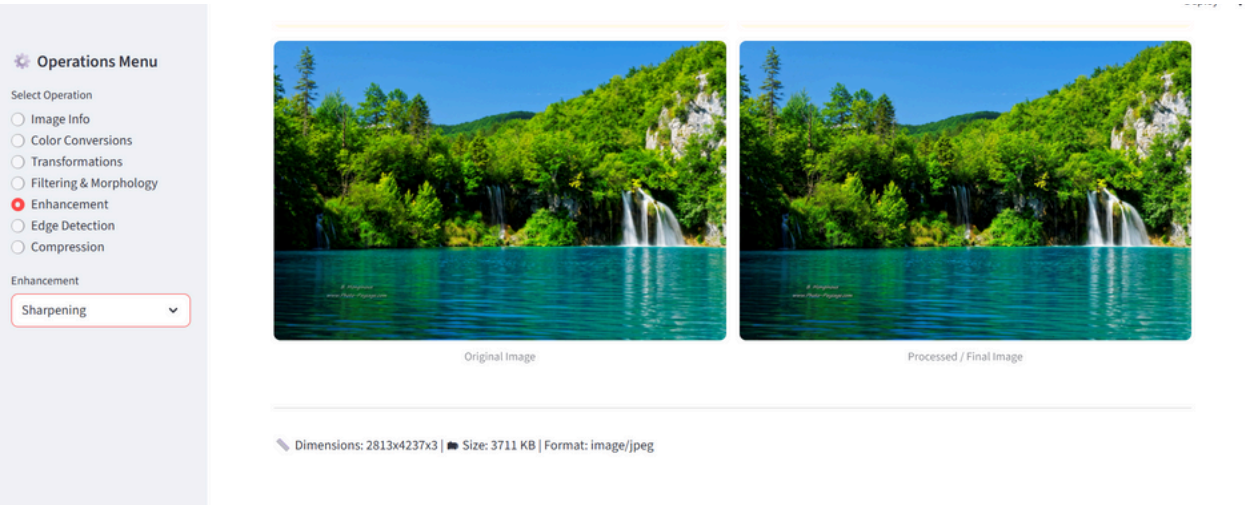

Deploy
Operations Menu
Select Operation
Image Info
Color Conversions
Transformations
Filtering & Morphology
Enhancement
Edge Detection
Compression
Category
Filtering
Morphology
Filter
Laplacian
Kernel
3
picture.jpeg
3.6MB
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
The use_column_width parameter has been deprecated and will be removed in a future release. Please utilize the use_container_width parameter instead.
Original Image
Processed / Final Image
Dimensions: 2813x4237x3 | Size: 3711 KB | Format: image/jpeg

5.Enhancement :

- Histogram Equalization
- Contrast Stretching
- Sharpening

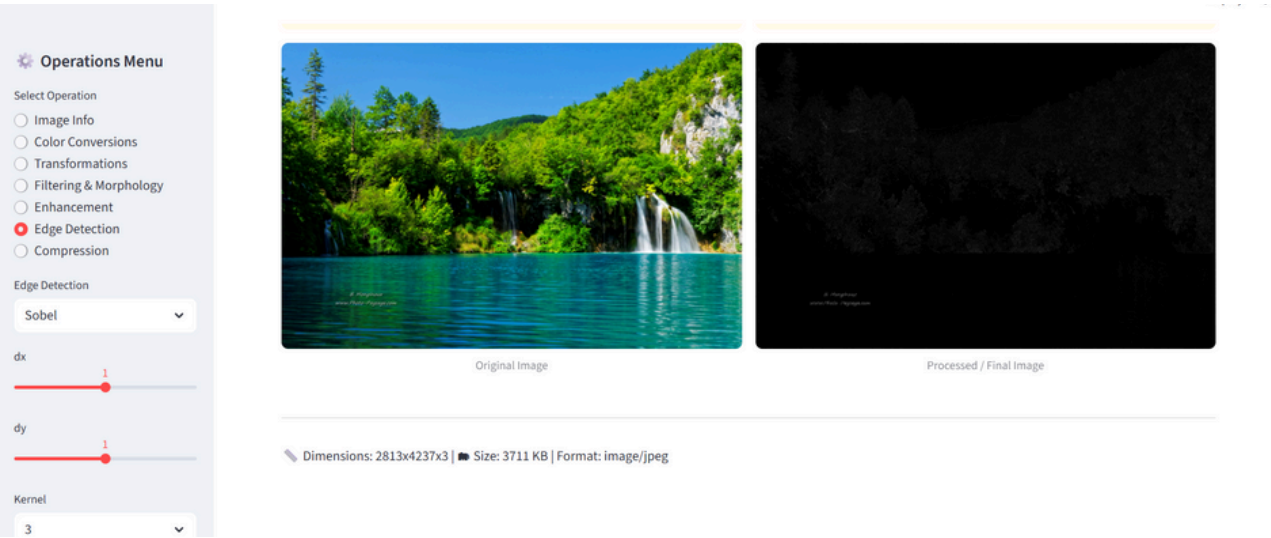


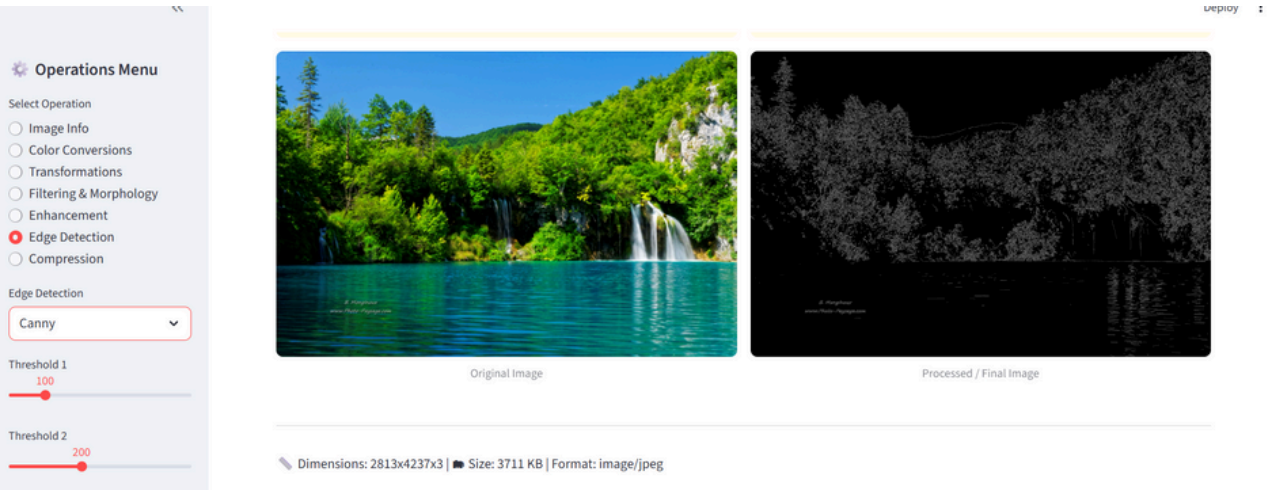


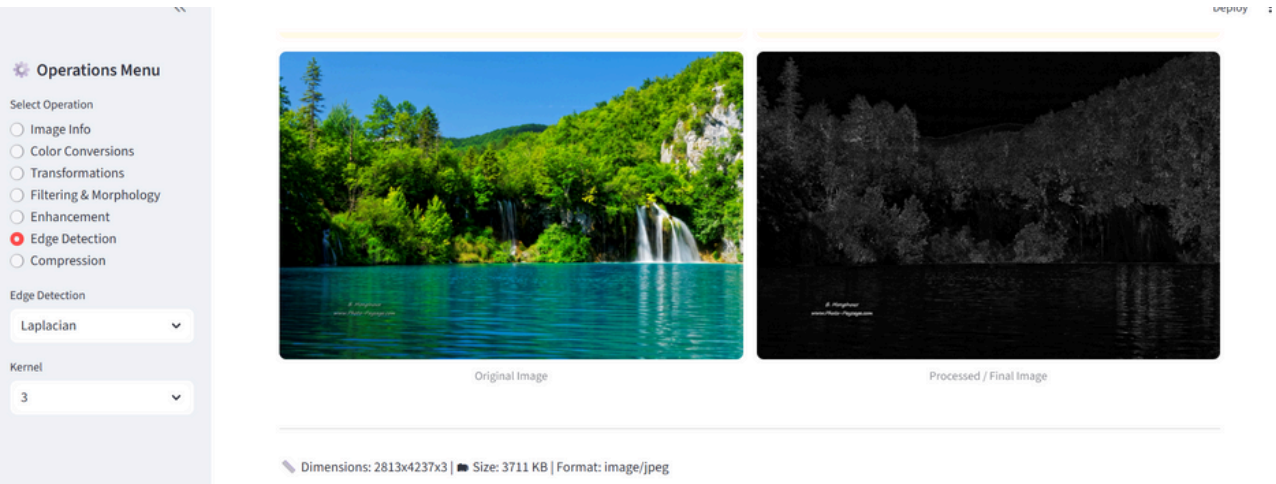


6.Edge Detection :

- Sobel
- Canny
- Laplacian

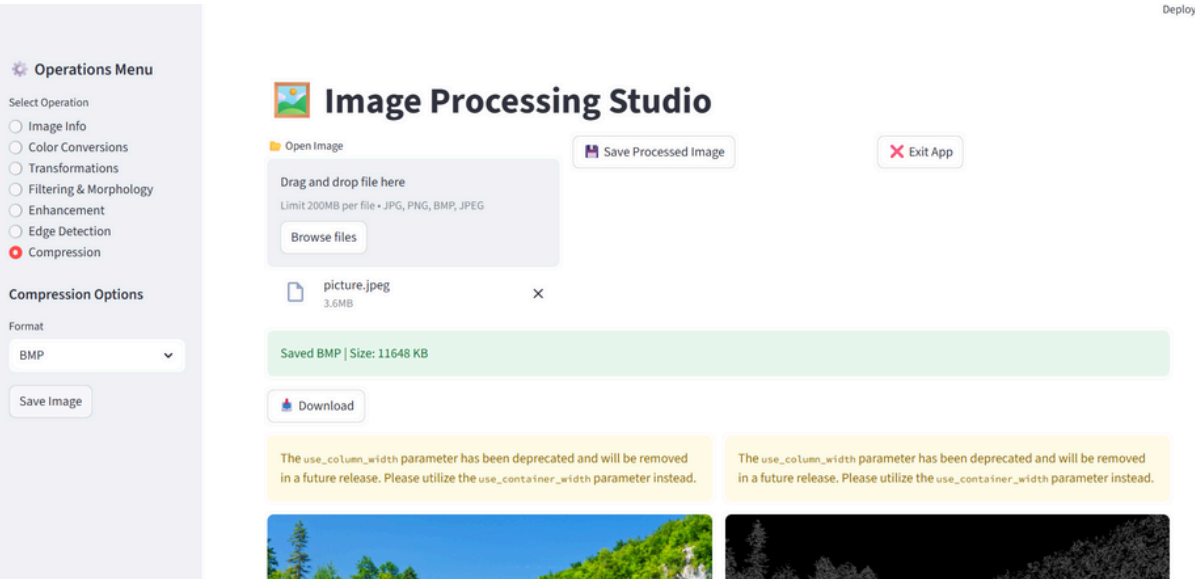


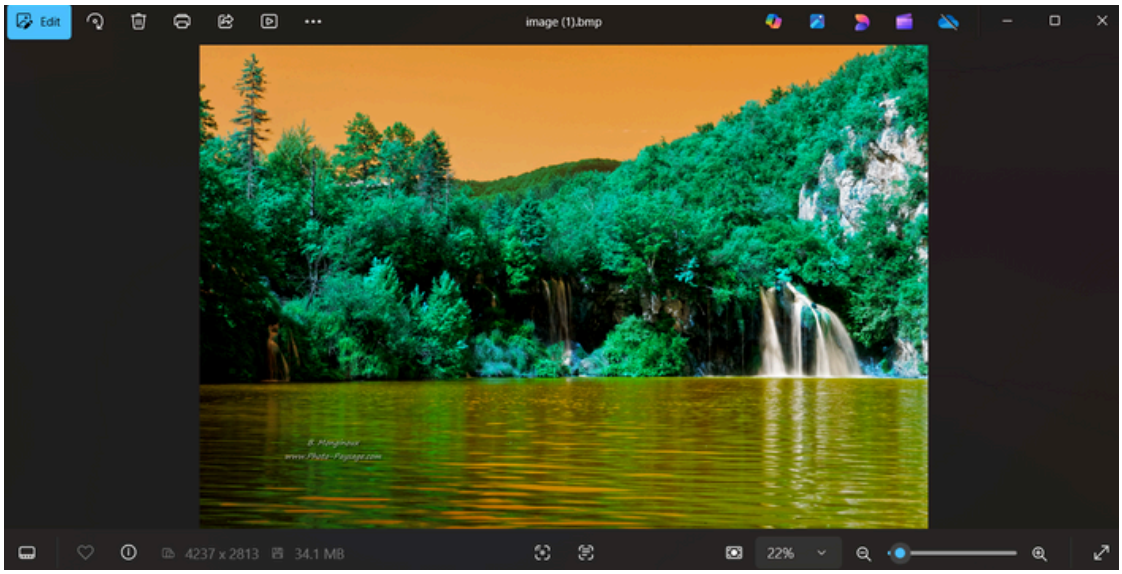




7.Compression :

- Save in JPG, PNG, BMP
- Compare file size & quality





Explanation of Algorithms Used :

This project implements several core algorithms in digital image processing and computer vision. The following explains the key methods used:

1. Color Conversion

- RGB to Grayscale: Converts a 3-channel color image into a single intensity channel by applying a weighted sum of R, G, and B values.
- RGB to HSV/LAB: Transforms color into alternative representations better suited for tasks like color filtering and segmentation.

2. Geometric Transformations

- Scaling (Resize): Achieved using interpolation methods such as Nearest Neighbor, Bilinear, and Bicubic, which estimate pixel values at new positions.
- Translation (Shift): Performed using an affine transformation matrix that repositions the image by specified x and y offsets.
- Affine Transform: Uses a 2×3 matrix that preserves collinearity and parallelism but allows rotation, translation, and scaling.
- Perspective Transform: Applies a 3×3 homography matrix to simulate the change of viewpoint, often used in rectification.

3. Filtering and Morphology

- Mean Filter: Smooths images by averaging pixel values in a neighborhood.
- Gaussian Filter: Reduces noise using a Gaussian kernel, preserving edges better than mean filtering.
- Median Filter: Replaces each pixel with the median of its neighbors, highly effective against salt-and-pepper noise.
- Sobel Operator: Computes image gradients to detect edges in horizontal or vertical directions.
- Laplacian Operator: Captures intensity changes in all directions using a second-order derivative.
- Dilation and Erosion: Morphological operations that expand or shrink foreground regions using structuring elements.
- Opening and Closing: Combinations of erosion and dilation used to remove noise or fill gaps in binary images.

4. Enhancement

- Histogram Equalization: Redistributes intensity values to enhance global contrast.
- Contrast Stretching: Linearly expands intensity ranges to improve visibility.
- Sharpening (Laplacian filter): Enhances edges by emphasizing high-frequency details.

5. Edge Detection

- Sobel: Detects edges by measuring gradient magnitude in x and y directions.
- Canny Edge Detector: A multi-stage algorithm including noise reduction, gradient calculation, non-maximum suppression, and hysteresis thresholding to detect strong and weak edges.
- Laplacian: Captures regions of rapid intensity change using second derivatives.

6. Compression

- JPG: Lossy compression using Discrete Cosine Transform (DCT) to reduce file size while sacrificing some quality.
- PNG: Lossless compression based on DEFLATE algorithm, preserves quality but with larger files.
- BMP: Uncompressed bitmap storage, very large in size but retains original quality.

Conclusion :

This project successfully demonstrated the design and implementation of an interactive Image Processing Toolkit using Streamlit and OpenCV. The toolkit covered key concepts in computer vision, including color space conversions, geometric transformations, filtering, morphological operations, image enhancement, edge detection, and compression. Through both theoretical study and practical implementation, the project bridged the gap between foundational image processing algorithms and real-world applications.

The step-by-step demonstrations and visual results highlight the importance of understanding how each operation affects image quality, size, and representation. This work not only reinforced theoretical concepts like sampling, quantization, and PSFs but also provided hands-on experience in applying them through an intuitive user interface.

Overall, the assignment enhanced understanding of core computer vision principles, improved practical coding skills, and provided a strong base for more advanced applications in machine learning and artificial intelligence.