📘 Image Processing Toolkit – Technical Report

1.Project Overview:

The aim of this project is to design and implement a **GUI-based application** that demonstrates fundamental **image processing operations** in a simple and interactive way. The toolkit enables users to upload an image, apply various transformations, filters, and enhancements, and visualize results instantly.

This eliminates the need for manually writing code, making image processing accessible to students, beginners, and researchers.

---

2. Introduction

Image processing plays a vital role in **computer vision** and **AI-based applications** such as medical imaging, object recognition, and image enhancement.

This project introduces a **Streamlit-based graphical toolkit** integrated with **OpenCV** and **NumPy**. Users can perform real-time operations like:

- Color conversion (RGB ↔ BGR, HSV, YCbCr, Grayscale)
- Geometric transformations (rotation, scaling, translation, affine, perspective)
- Filtering (Gaussian, Median, Mean)
- Morphological operations (erosion, dilation, opening, closing)
- Image enhancement (histogram equalization, contrast stretching, sharpening)
- Edge detection (Sobel, Canny, Laplacian)
- Image compression and saving results

The system displays **Original vs. Processed Images** side by side, making analysis easier.

3🔍 Technical Flow

1. **Upload / Default Image Loading**
    - Accepts JPG, PNG, BMP.
    - Falls back to default image if none uploaded.
2. **Processing Layer**
    - Core OpenCV functions: cv2.cvtColor, cv2.warpAffine, cv2.filter2D, etc.
    - NumPy used for matrix operations.
3. **Visualization Layer**
    - Streamlit renders images in RGB/Gray.
    - Side-by-side comparison or tab display.

4. **Output Layer**
    - Encodes processed image using cv2.imencode.
    - Converts buffer → base64 → downloadable link.

---

4. Methodology

The workflow is divided into the following steps:

1. **Image Upload / Default Image**
    - Users can upload an image using Streamlit's file uploader.
    - If no image is uploaded, a **default photo** (e.g., KTM demo image) is displayed.
2. **Operation Selection**
    - Sidebar menu provides categories like **Color Conversion, Transformations, Filtering, Enhancement, Edge Detection, Compression**.
3. **Processing**
    - Based on user input, OpenCV functions (e.g., cv2.cvtColor, cv2.GaussianBlur, cv2.Canny) are applied.
    - NumPy handles array manipulation for stretching and scaling.
4. **Display Results**
    - Original and processed images are displayed side-by-side.
    - Image details (resolution, channels, size, format) are also shown.
5. **Save & Download**
    - Processed image can be downloaded in **PNG, JPG, or BMP** format.

---

5. Results

📌 GUI Preview

The application provides an **intuitive interface** where users can interactively test image operations.

- Example: Edge Detection using **Canny filter**
- Image information
- Colour conversion
- Edge detection
- Compression
- Transformation…..

Operations Menu

1. Image Info & Channel

Feature Category

- (•) Image Info
- ( ) Color Conversion
- ( ) Transformations
- ( ) Filtering & Morphology
- ( ) Enhancement
- ( ) Edge Detection
- ( ) Compression

Open: Upload an image

Drag and drop file here

Limit 200MB per file • PNG, J...

Browse files

17534101354...
3.0MB

Deplc

Operations Menu

1. Image Info & Channel

Feature Category

- ( ) Image Info
- (•) Color Conversion
- ( ) Transformations
- ( ) Filtering & Morphology
- ( ) Enhancement
- ( ) Edge Detection
- ( ) Compression

Open: Upload an image

Drag and drop file here

Limit 200MB per file • PNG, J...

Browse files

17534101354...
3.0MB

Deploy

Operations Menu

1. Image Info & Channel

Feature Category

- ( ) Image Info
- ( ) Color Conversion
- (•) Transformations
- ( ) Filtering & Morphology
- ( ) Enhancement
- ( ) Edge Detection
- ( ) Compression

Open: Upload an image

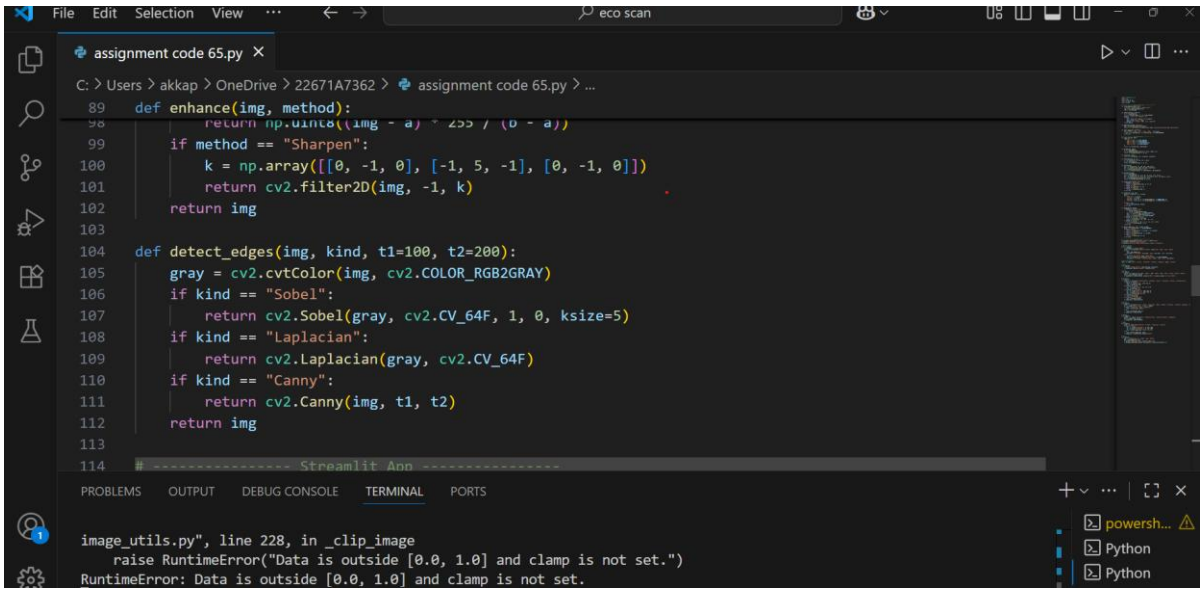
Drag and drop file here

Limit 200MB per file • PNG, J...

Browse files

17534101354...
3.0MB


Operations Menu
1. Image Info & Channel
Feature Category
Image Info
Color Conversion
Transformations
Filtering & Morphology
Enhancement
Edge Detection
Compression
Open: Upload an image
Drag and drop file here
Limit 200MB per file • PNG, J...
Browse files
17534101354...
3.0MB
Operations Menu
1. Image Info & Channel
Feature Category
Image Info
Color Conversion
Transformations
Filtering & Morphology
Enhancement
Edge Detection
Compression
Open: Upload an image
Drag and drop file here
Limit 200MB per file • PNG, J...
Browse files
17534101354...
3.0MB


File Edit Selection View
eco scan
assignment code 65.py
C: > Users > akkap > OneDrive > 22671A7362 > assignment code 65.py > ...
def enhance(img, method):
if method == "Sharpen":
k = np.array([[0, -1, 0], [-1, 5, -1], [0, -1, 0]])
return cv2.filter2D(img, -1, k)
return img
def detect_edges(img, kind, t1=100, t2=200):
gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
if kind == "Sobel":
return cv2.Sobel(gray, cv2.CV_64F, 1, 0, ksize=5)
if kind == "Laplacian":
return cv2.Laplacian(gray, cv2.CV_64F)
if kind == "Canny":
return cv2.Canny(img, t1, t2)
return img
---------------- Streamlit App ----------------
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
image_utils.py", line 228, in _clip_image
raise RuntimeError("Data is outside [0.0, 1.0] and clamp is not set.")
RuntimeError: Data is outside [0.0, 1.0] and clamp is not set.
powersh...
Python
Python

6. Conclusion

The **Image Processing & Analysis Toolkit** demonstrates how advanced image operations can be simplified into a **click-based GUI application**.

- ✅ Makes learning easier for students.
- ✅ Reduces programming effort.
- ✅ Useful for **educational demonstrations, research, and prototyping**.

By combining **OpenCV + Streamlit**, this project bridges the gap between **theory and hands-on practice** in image processing.

A. Sai Snehitha
22671A7365
AIML-B