SV30703 DIGITAL IMAGE PROCESSING

ASSIGNMENT 2

My Own ADOBE PHOTOSHOP

LECTURER: PROF. DR. ABDULLAH BIN BADE

No.	Name	Matrix
1	MOHAMAD ADLI ZILIKHRAM BIN SAHARUDIN	BS22110469

Table of Contents

Introduction

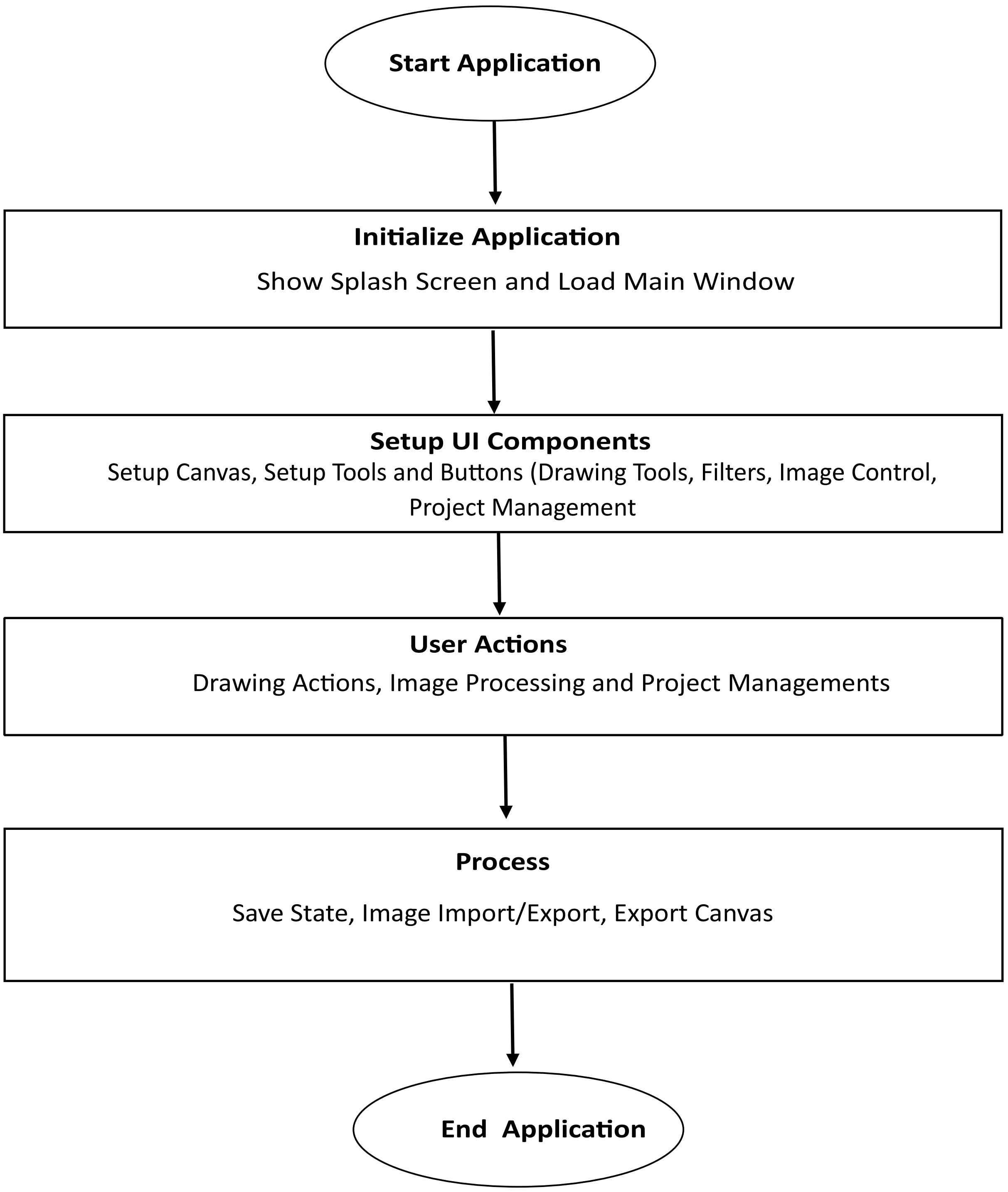
This Python script is a comprehensive GUI-based image editing and drawing application developed using various functions, including python-OpenCV, matplotlib, and PyQt5. It offers a wide range of functionalities for both artistic and technical image manipulation. The application allows users to create canvases of adjustable dimensions and provides various drawing tools, such as pencil, brush, eraser, and shape tool. With these functions, the user can do any freeform drawing and add shapes like rectangles, ellipses, and lines. A text tool is also included, offering customization options for fonts, colors, and opacity.

The application supports image processing features, including importing and exporting images, applying filters like Gaussian Blur, Median Filter, and Bilateral Filter, and performing dehazing operations. It also incorporates edge detection capabilities using popular algorithms such as Sobel, Laplacian, and Canny edge detection. Users can activate grid lines and rulers for precise drawing and alignment.

In addition to basic editing, the application includes project management functionalities that allow the saving and loading of projects, preserving all image and drawing states through pickle serialization. The interface is designed to be user-friendly, featuring interactive panels for different toolsets, filter applications, and even 3D image representations. Visual feedback is provided through splash screens and tool-specific buttons, making the application versatile for a wide range of image editing tasks.

2.0 Application Explanation

2.1 Flow Diagram


Start Application
Initialize Application
Show Splash Screen and Load Main Window
Setup UI Components
Setup Canvas, Setup Tools and Buttons (Drawing Tools, Filters, Image Control, Project Management
User Actions
Drawing Actions, Image Processing and Project Managements
Process
Save State, Image Import/Export, Export Canvas
End Application

System Architecture

1. Application Initialization

The application starts by initializing the main window and splash screen. The splash screen displays an animated GIF for a set duration before loading the main window with all UI components and tools.

2. Main Window and UI Components

The main window (CanvasWindow) serves as the central component. It contains the following key elements:

- **Canvas:** The primary drawing area where images, shapes, and text can be added.
- **Overlay Layer:** A transparent QPixmap used to draw temporary elements like shapes and sketches.
- **Panels:**
  - **Left Panel:** Contains buttons for tools (draw, erase, crop, etc.).
  - **Top Panel:** Contains options for pen settings (color, size, opacity) and other quick-access features.
  - **Right Panel:** Houses different tool menus (image editor, filters, 3D visualization) managed via a QStackedLayout.

3. Drawing Tools

The application supports various drawing tools, each encapsulated in its own class:

- **DrawingTool:** Handles freehand drawing with customizable pen settings.
- **EraseTool:** Allows erasing with adjustable size and style.
- **ShapeTool:** Enables drawing shapes like rectangles, ellipses, lines, and triangles.
- **TextTool:** Provides functionality to add and customize text on the canvas.

4. Image Processing

The application includes several image processing functions:

- **Filters:**
    - Gaussian Blur
    - Median Filter
    - Bilateral Filter
    - Unsharp Masking
    - Laplacian Filter
    - Sobel Filter
- **Edge Detection:**
    - Canny Edge Detection
    - Prewitt Edge Detection
    - Sobel Edge Detection
- **Dehazing:** Removes haze from images.
- **Glitch Effects:** Adds artistic glitch effects.

5. Grid and Ruler

The GridTool class manages the display of grid lines and rulers for precision drawing. Users can toggle these features on or off.

6. Project Management

The application supports project saving and loading with the following capabilities:

- **Save Project:** Saves the current canvas, overlay, images, and text objects using pickle serialization.
- **Open Project:** Loads a previously saved project and restores all elements.

7. Undo/Redo Functionality

The application maintains a history stack to allow undoing and redoing of actions. The states are saved as snapshots of images, overlays, and text objects.

8. Canvas Interaction

Users can perform various interactions with the canvas:

- **Import Images:** Load images and add them to the canvas.
- **Drag and Scale:** Move and resize images.
- **Crop Mode:** Crop sections of the canvas using a rectangular selection.
- **Zoom In/Out:** Adjust the zoom level for detailed editing.

9. 3D Visualization

The application supports generating 3D representations of images using matplotlib. This feature is accessible via the right panel's 3D representation section.

10. Event Handling

- The paintEvent method handles all drawing and rendering logic, including:
- Drawing the canvas, overlay, and imported images.
- Rendering grid lines, rulers, and text objects.
- Updating the mini canvas (thumbnails).

11. Menu and Navigation

The QMenuBar provides options for file operations (import, export, save project) and view settings (toggle grid lines, rulers, thumbnails). A QStackedLayout allows switching between different tool categories (image editor, filters, 3D representation).

12. Additional Features

- **Thumbnails:** Display mini versions of the current canvas for quick previews.
- **Flip Operations:** Flip images horizontally or vertically.
- **Zoom Controls:** Zoom in, zoom out, and reset zoom to the default view.

2.3 System Algorithm

1. Application Initialization

- **Load Splash Screen**
    - Display a splash screen with an animated GIF for a specified duration.
    - Close the splash screen and launch the main application window.
- **Initialize Main Window**
    - Set up the main application window (CanvasWindow) with dimensions and a canvas (QPixmap) for drawing.
    - Load icons, set window properties, and create UI components such as buttons, sliders, and panels.

2. User Interaction and Tool Selection

- **Tool Selection Logic**
    - User selects a tool by clicking on buttons for:
        - **Drawing** (Pencil, Brush, etc.)
        - **Erasing**
        - **Shapes** (Rectangle, Ellipse, Line, etc.)
        - **Text**
        - **Image Import/Export**
        - **Filters**
    - Set flags (e.g., drawing_mode, eraser_mode, shape_mode) to determine the active tool.

3. Drawing Tools

- **Start Drawing** (start_drawing)
    - Capture the mouse position when the user clicks on the canvas.
    - Initialize drawing by storing the starting position.
- **Continue Drawing** (continue_drawing)
    - Track mouse movements while holding down the button.
    - Draw lines dynamically on the overlay_pixmap using QPainter.
- **End Drawing (end_drawing)**
    - Finalize the drawing by committing the overlay to the main canvas (pixmap).
    - Clear the overlay and update the display.

4. Eraser Tool

- **Start Erasing** (start_erasing)
    - Capture the starting position for erasing.
- **Continue Erasing** (continue_erasing)
    - Erase by drawing over the canvas with a white pen (simulating erasure).
- **End Erasing** (end_erasing)
    - Commit the changes to the main canvas and clear the overlay.

5. Shape Tool

- **Start Drawing Shape** (start_drawing)
    - Capture the starting position for the shape.
- **Continue Drawing Shape** (continue_drawing)
    - Preview the shape dynamically on the overlay while dragging the mouse.
- **End Drawing Shape** (end_drawing)
    - Draw the final shape on the main canvas and clear the overlay.

6. Text Tool

- **Start Typing** (start_typing)
    - Capture the position where the text will be added.
    - Initialize the text input.
- **Handle Key Press** (handle_key_press)
    - Update the text dynamically based on user input.
    - Support backspace and enter keys for editing and finalizing the text.
- **Finalize Text** (finalize_text)
    - Commit the text to the main canvas and store it in the text_objects list.

7. Image Import and Export

- **Import Image** (import_image)
    - Open a file dialog to select an image.
    - Load the image using cv2.imread and convert it to QPixmap.
    - Add the image to the canvas and center it.
- **Export Image** (export_as_png)
    - Merge the main canvas, overlay, and images.
    - Save the result as a PNG file using a file dialog.

8. Filters and Image Processing

- **Apply Filters**
    - **Gaussian Blur** (apply_gaussian_blur)
        - Apply a Gaussian blur with a user-specified kernel size.
    - **Median Filter** (apply_median_filter)
        - Apply a median filter for noise reduction.
    - **Bilateral Filter** (apply_bilateral_filter)
        - Apply a bilateral filter to preserve edges while reducing noise.
    - **Unsharp Masking** (apply_unsharp_mask)
        - Sharpen the image using unsharp masking.
    - **Laplacian Filter** (apply_laplacian_filter)
        - Perform edge detection using the Laplacian filter.
    - **Sobel Filter** (apply_sobel_filter)
        - Apply the Sobel operator for edge detection.
- **Dehazing** (apply_dehaze)
    - Remove haze from an image using dark channel prior and guided filtering.

9. Project Management

- **Save Project** (save_project)
    - Serialize the current state of the project (canvas, images, text) using pickle.
    - Save images and overlays to a temporary folder.
- **Open Project** (open_project)
    - Load a saved project file and restore the canvas, images, and text objects.

10. Undo/Redo Functionality

- **Save State** (save_state)
    - Save the current state of the canvas, images, and text to an undo_stack.
- **Undo Action** (undo_action)
    - Restore the last saved state from undo_stack and add the current state to redo_stack.
- **Redo Action** (redo_action)
    - Restore the next state from redo_stack and add the current state to undo_stack.

11. Grid and Ruler

- **Toggle Grid** (toggle_grid)
    - Show or hide grid lines on the canvas.
- **Toggle Ruler** (toggle_ruler)
    - Show or hide rulers along the top and left sides of the canvas.

12. 3D Visualization

- **Show 3D Representation (**show_3d_representation)
    - Generate and display a 3D representation of the current image using matplotlib.

3.0 Strength and Uniqueness

3.1 Features Aligning with Requirements

1. Thumbnail Creation for Loaded/Modified Images

Strength: The code provides the ability to display a mini canvas that updates with the current state of the image.

Uniqueness: The MiniCanvasWindow class allows users to see a live thumbnail preview, which is useful for tracking changes or maintaining an overview of the current image state.

2. Multi-Window Support with Gridlines and Rulers

Strength: The application supports multiple drawing windows and features gridlines and rulers.

Uniqueness: The GridTool class allows toggling gridlines and rulers on the canvas, aiding in precision image manipulation. This can be combined with the multi-window capability to enhance productivity.

3. Real-Time Histogram Panel

Strength: The code displays a real-time histogram of the loaded images and updates dynamically when changes are made.

Uniqueness: The ability to differentiate and toggle between RGB channels and a combined channel view provides a flexible and detailed analysis of image histograms. This is implemented via checkboxes and a FigureCanvas for visualization.

4. Image Filtering with Adjustable Properties

Strength: The application supports six filtering methods:

Gaussian Blur

Median Filter

Bilateral Filter

Unsharp Masking

Laplacian Filter

Sobel Filter

Uniqueness: Each filter has associated sliders to adjust properties like kernel size, strength, and other parameters. This allows users to fine-tune filters in real time, offering a highly interactive experience.

5. Bit-Plane Slicing

Strength: The code can perform bit-plane slicing and display output using both plt.show() and cv2.imshow().

Uniqueness: Eight buttons for each bit-plane (0 to 7) allow users to explore the individual bit-plane components of an image, providing flexibility in analysis and visualization.

6. Edge Detection Techniques

Strength: The application implements three edge detection techniques:

Canny Edge Detection

Prewitt Edge Detection

Sobel Edge Detection

Uniqueness: Each technique is adjustable using sliders, allowing users to modify properties such as thresholds and kernel sizes for precise edge detection results.

7. Image Contouring

Strength: The code supports image contouring with adjustable threshold values.

Uniqueness: The real-time sliders enable users to dynamically refine the contour detection process, making it easy to identify objects and boundaries in images.

8. Interactive Image Thresholding

Strength: Thresholding can be applied using an interactive slider for precise control.

Uniqueness: The seamless integration of sliders makes it easy to experiment with different threshold values and immediately see the results.

9. Power Law Transformation

Strength: The application includes Power Law (Gamma) Transformation with a slider to adjust the gamma value interactively.

Uniqueness: This allows users to enhance image brightness and contrast dynamically, providing flexibility in improving image quality.

10. Piecewise Linear Transformation

Strength: Supports Piecewise Linear Transformation with an interactive slider for adjusting transformation points.

Uniqueness: This feature helps in performing customized contrast stretching, offering more control over image enhancement.

11. Erosion and Dilation

Strength: The code supports image erosion and dilation with adjustable kernel sizes.

Uniqueness: Sliders for controlling kernel size provide flexibility in morphological operations, useful for tasks like noise removal and shape analysis.

12. Histogram Equalization

Strength: Implements histogram equalization and CLAHE (Adaptive Histogram Equalization).

Uniqueness: The ability to apply both standard and adaptive histogram equalization ensures better contrast enhancement for a variety of images.

13. Project Management and Undo/Redo

Strength: The code includes robust project management features:

Save and load projects with pickle serialization.

Undo/Redo functionality to revert or reapply changes.

Uniqueness: The combination of state-saving and project loading ensures that users can work on long-term projects without losing progress.

14. Comprehensive UI and Interactivity

Strength: The interface includes a variety of tools and buttons for:

Drawing, erasing, cropping, and manipulating images.

Applying filters and transformations with real-time feedback.

Uniqueness: The use of stacked layouts and dynamic UI updates (e.g., sliders, buttons, and real-time displays) makes the application versatile and user-friendly.

3.1 Special Features

Dehazing Functionality: Added an advanced dehaze feature for image enhancement.

Glitch Art Effect: A creative effect for glitch art generation.

Undo/Redo Support: State-saving functionality for undo/redo operations.

Export and Save Project: Ability to save and load projects, including all elements like images, text, and overlays.

Custom Shapes: Supports additional shapes like triangles and dashed lines in the ShapeTool.

Text Tool with Advanced Options:

- Customizable fonts, sizes, colors, and opacity.
- Real-time typing and dashed box visual aid.

Canvas Preview and Adjustable Size: Includes a preview when setting the canvas size dynamically.

Grid and Ruler Tools: Togglable gridlines and rulers for precise editing.

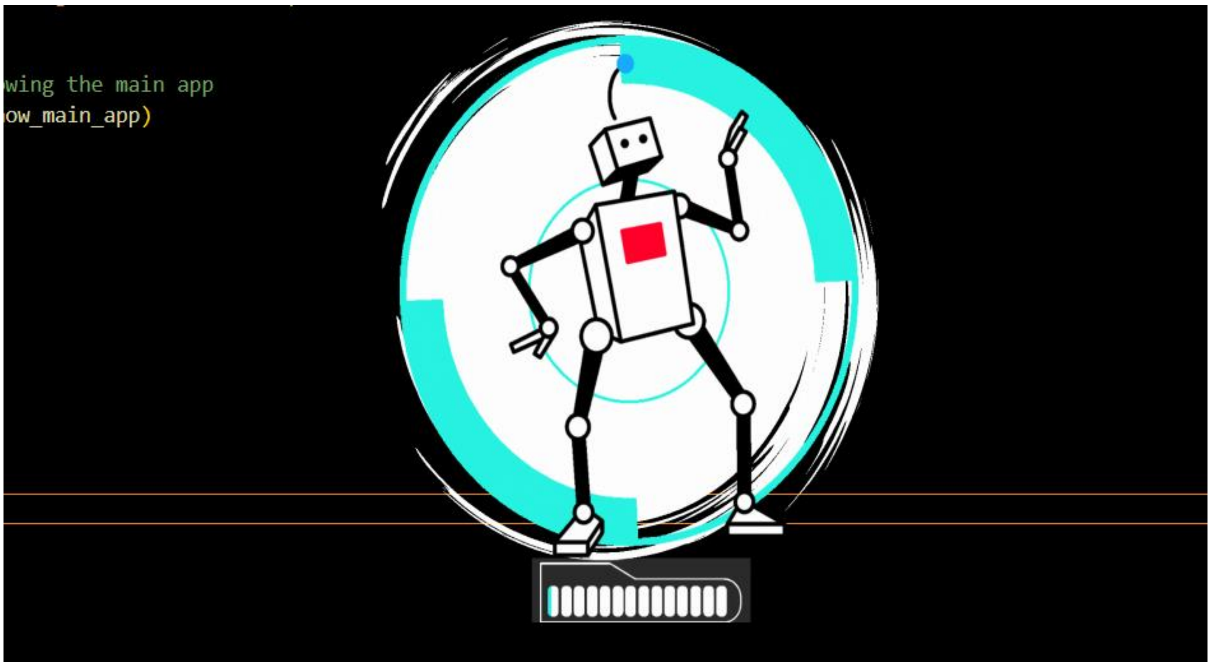
Zoom and Pan Tools: Implements zoom-in, zoom-out, and reset zoom functionality.

Image Flip: Horizontal and vertical flip tools.

3D Representation: Visualization of images in 3D (e.g., via histograms or other spatial representations).

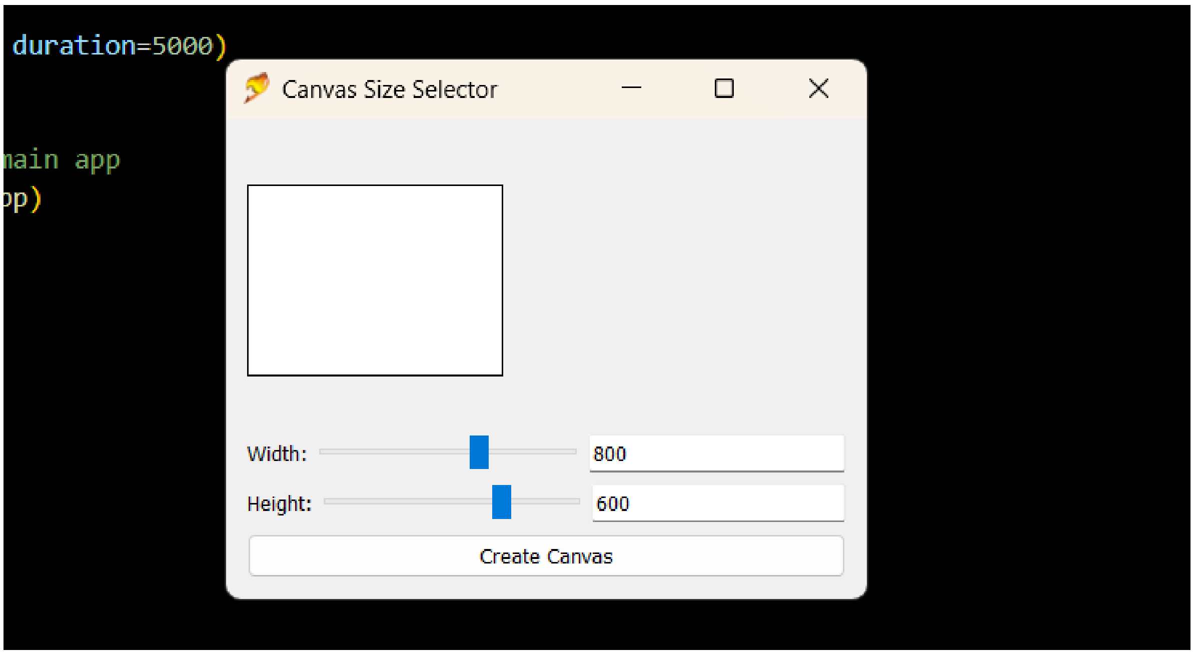
4.0 Sample output

Splash Screen



```
wing the main app
ow_main_app)
```

Canvas Initialization



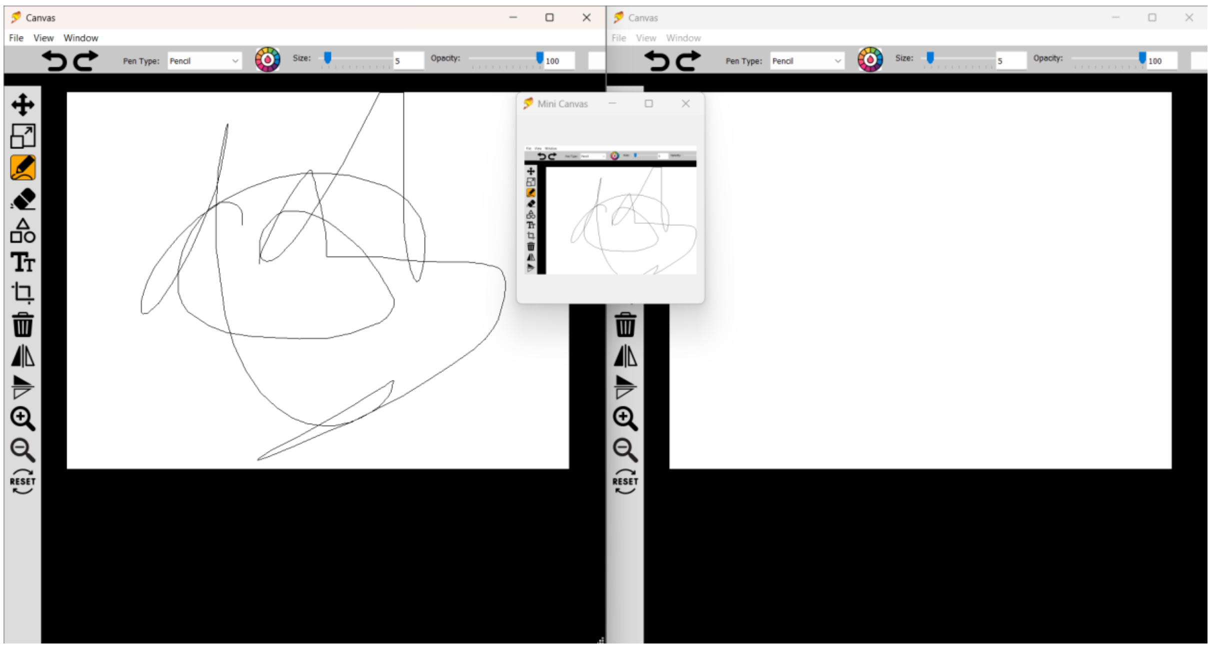
```
duration=5000)

main app
pp)
```

Canvas Size Selector

Width: 800

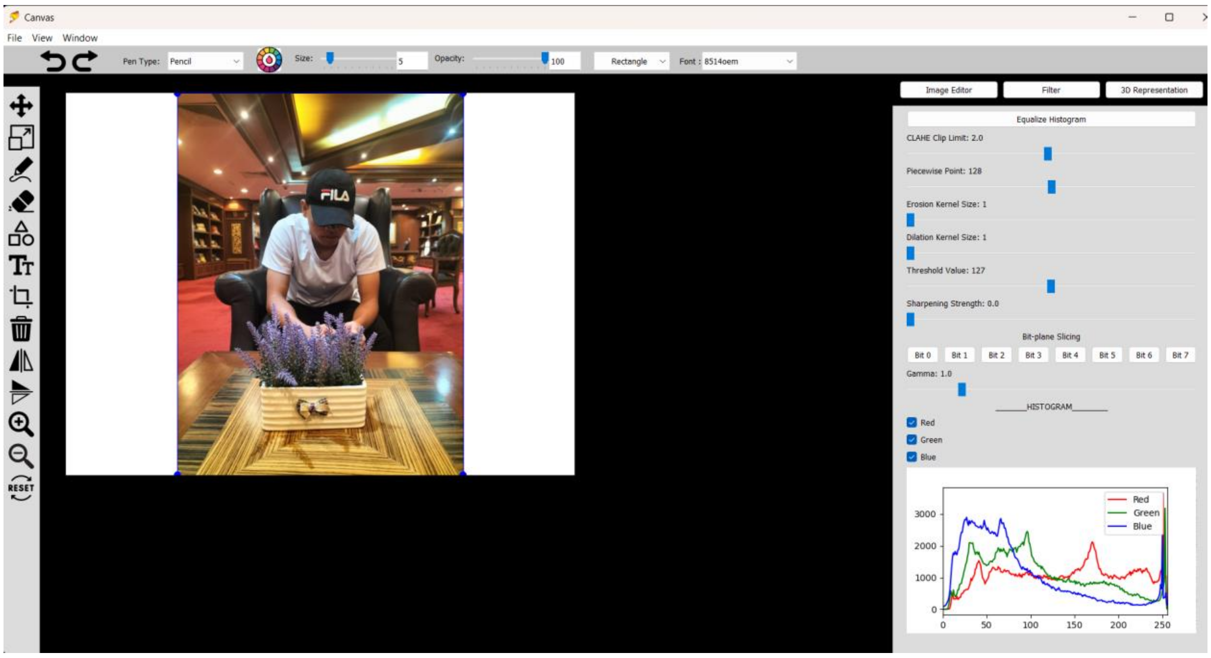
Height: 600

Multi-window and Thumbnail Creation



Pen Type: Pencil Size: 5 Opacity: 100

Real-Time Histogram with RGB channels



Pen Type: Pencil Size: 5 Opacity: 100 Rectangle Font : 8514oem

CLAHE Clip Limit: 2.0

Piecewise Point: 128

Erosion Kernel Size: 1

Dilation Kernel Size: 1

Threshold Value: 127

Sharpening Strength: 0.0

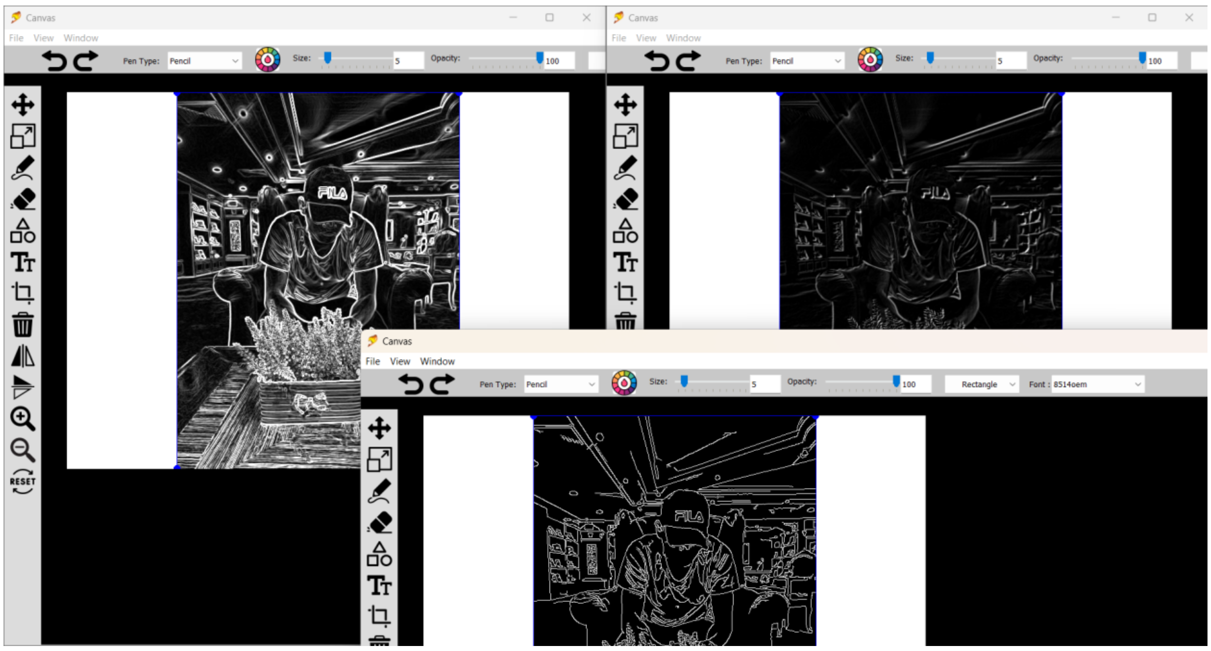
Gamma: 1.0

✓ Red

✓ Green

✓ Blue

Edge Detection (Canny, Sobel, Prewitt)

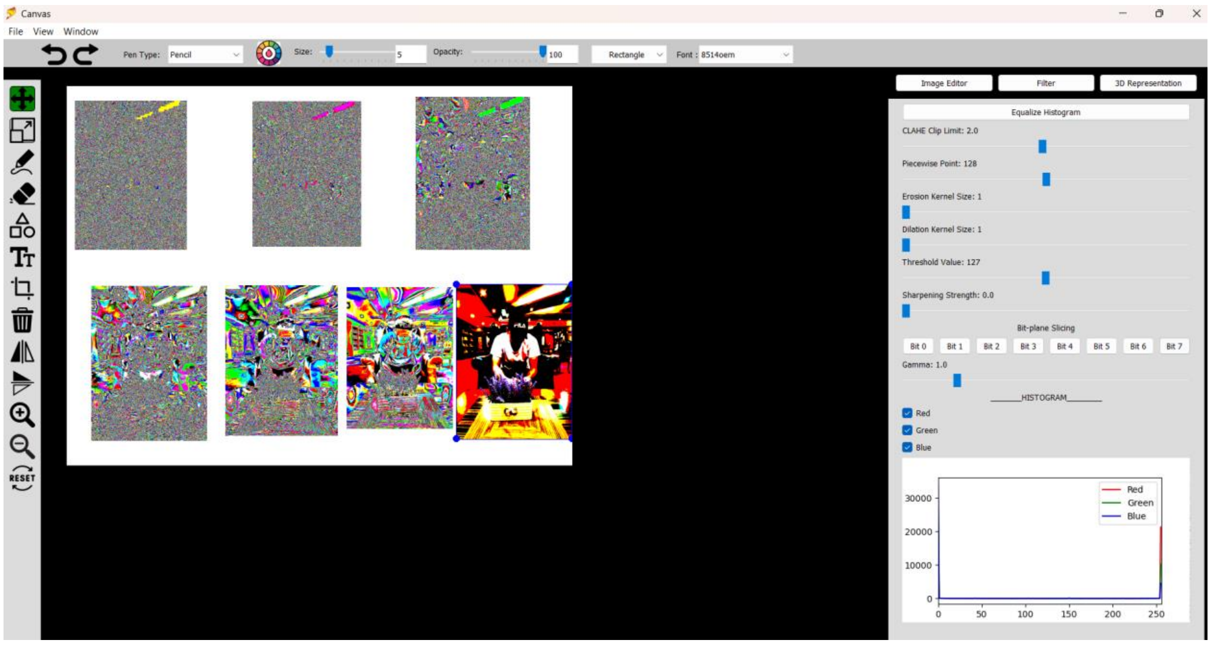


Pen Type: Pencil Size: 5 Opacity: 100

Pen Type: Pencil Size: 5 Opacity: 100

Pen Type: Pencil Size: 5 Opacity: 100 Rectangle Font : 8514oem

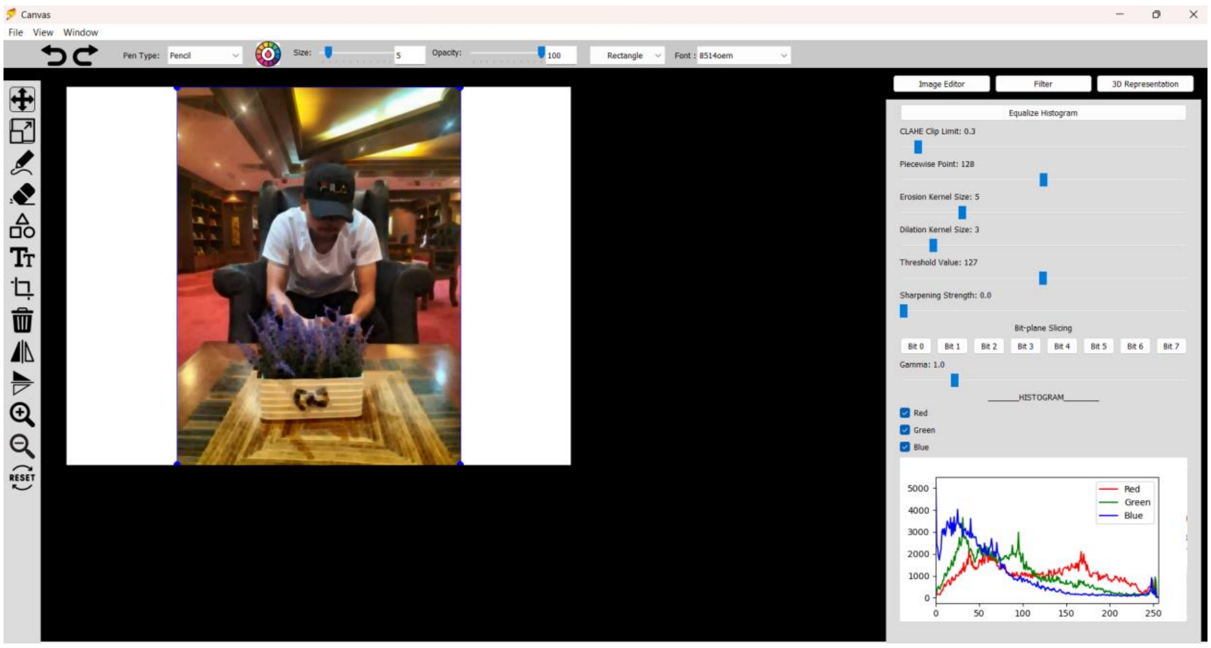
Bit-Plane Slicing



Pen Type: Pencil Size: 5 Opacity: 100 Rectangle Font : 8514oem

CLAHE Clip Limit: 2.0
Piecewise Point: 128
Erosion Kernel Size: 1
Dilation Kernel Size: 1
Threshold Value: 127
Sharpening Strength: 0.0
Gamma: 1.0
☑ Red
☑ Green
☑ Blue

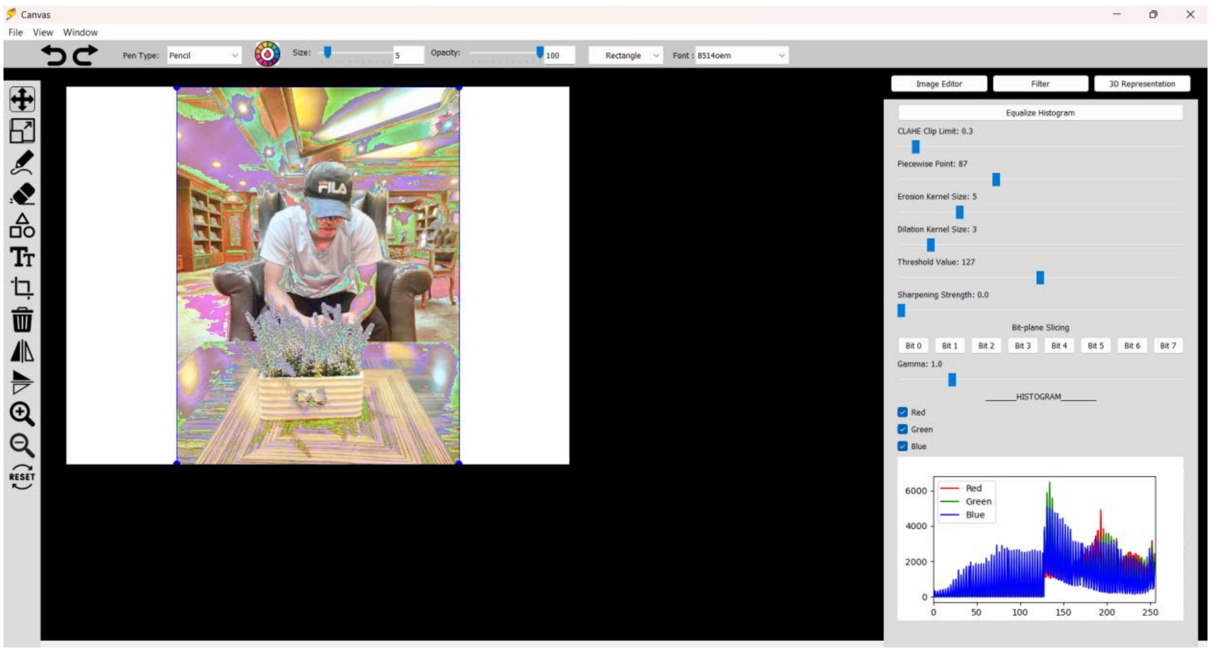
Filters with Sliders



Pen Type: Pencil
Size: 5
Opacity: 100
Rectangle
Font : 8514oem

CLAHE Clip Limit: 0.3
Piecewise Point: 128
Erosion Kernel Size: 5
Dilation Kernel Size: 3
Threshold Value: 127
Sharpening Strength: 0.0
Gamma: 1.0

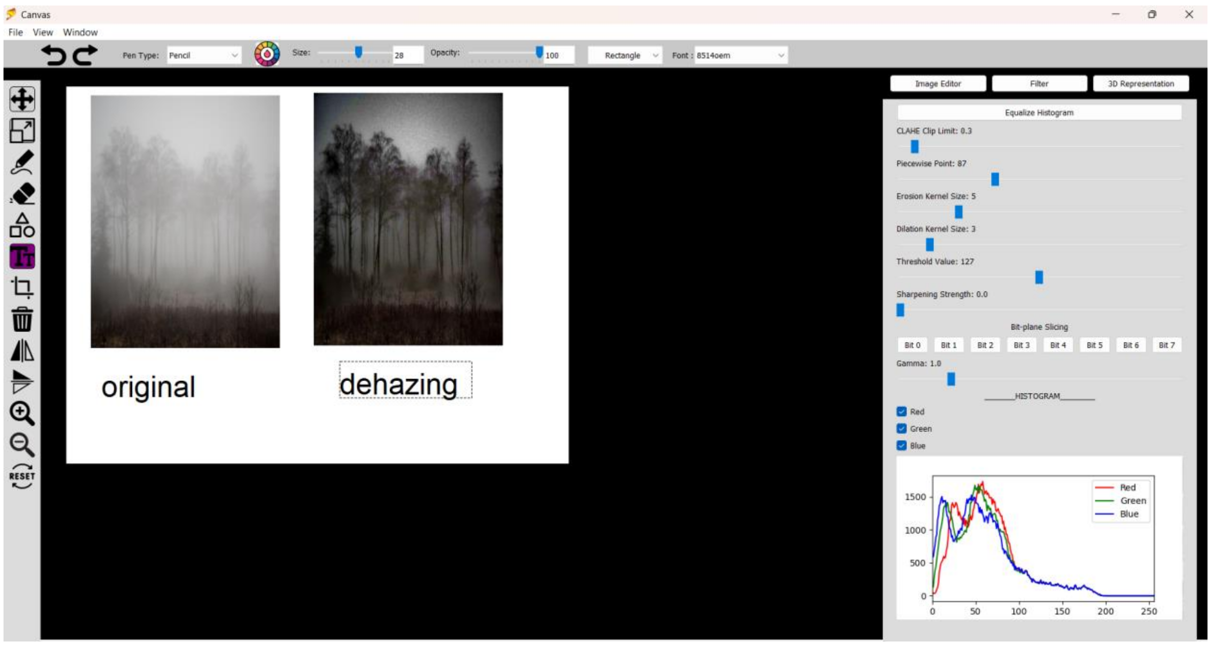
Image Transformation



Pen Type: Pencil
Size: 5
Opacity: 100
Rectangle
Font : 8514oem

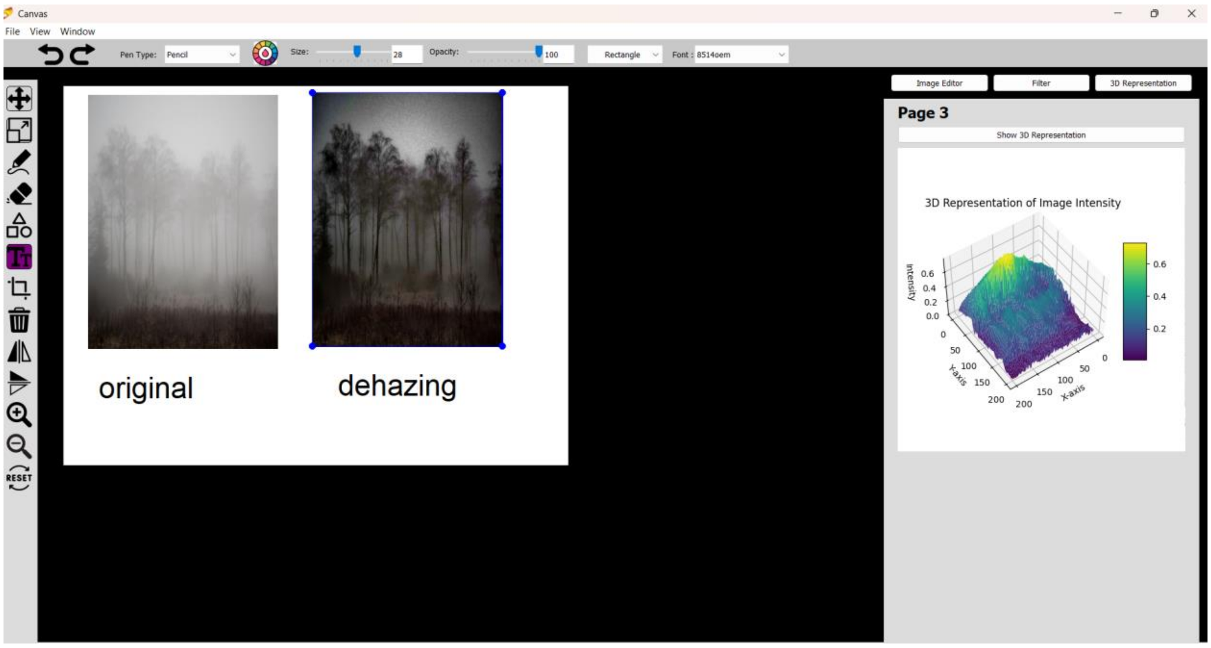
CLAHE Clip Limit: 0.3
Piecewise Point: 87
Erosion Kernel Size: 5
Dilation Kernel Size: 3
Threshold Value: 127
Sharpening Strength: 0.0
Gamma: 1.0

Dehazing



Pen Type: Pencil
Size: 28
Opacity: 100
Rectangle
Font : 8514oem
CLAHE Clip Limit: 0.3
Piecewise Point: 87
Erosion Kernel Size: 5
Dilation Kernel Size: 3
Threshold Value: 127
Sharpening Strength: 0.0
Gamma: 1.0
☑ Red
☑ Green
☑ Blue

3D Representation



Pen Type: Pencil
Size: 28
Opacity: 100
Rectangle
Font : 8514oem