Objectives

1. Enhance the control panel UI styling, especially for zoom-related buttons, to improve visual clarity and feedback based on the zoom state.
2. Ensure all interactive buttons are disabled until a file is loaded and re-enable them based on specific conditions, such as fixed zoom mode.
3. Resolve issues with the "remove line" function to allow for single-step line removal, including updates to adjust SEARCH_RADIUS based on line width dynamically.
4. Transform dark line processing functions into a double 2D pointer method for improved memory efficiency, adapting related structures and data handling functions.
5. Update histogram functionality to activate automatically upon file load and integrate it with existing button controls.

Activities

- **Control Panel Styling**:
    - Updated zoom-related button colors with the following logic:
        - Button shows **"Deactivate Zoom"** and **"Fix Zoom"**: displays blue.
        - Button shows **"Deactivate Zoom"** and **"Unfix Zoom"**: displays red.
        - Button shows **"Activate Zoom"**: no color change.
    - Added logic to automatically block all buttons when no file is loaded, except for the "Browse" button.
- **File Load Signal and Button Enablement**:
    - Created a `fileLoaded` signal to indicate when a file is loaded successfully.
    - Modified file loading logic to emit `fileLoaded`, which connects to the `enableButtons` slot to enable buttons when a file is loaded.
    - Set up a vector to store all interactive buttons and added `enableButtons` to toggle button states.
    - Updated histogram toggle button control to activate upon file load, adding it to `m_allButtons` and linking it to the `fileLoaded` signal.
- **Remove Line Function Enhancement**:
    - Investigated why the line removal function did not remove lines in one step.
    - Tested with various `SEARCH_RADIUS` values and found that setting it to 100 resolved the issue.
    - Implemented `calculateSearchRadius` function to determine `SEARCH_RADIUS` dynamically based on line width:

        - Radius set to twice the actual line width, clamped between 10 and 200 pixels.
        - Updated `findReplacementValue` to accept line width for processing.
        - Enhanced function to use smaller `SEARCH_RADIUS` for thin lines and larger values for thicker lines, minimizing unintended pixel changes.
    - Modified control panel options to allow in-object line removal in multiple selections via mouse, keyboard, or select-all button.
- **Conversion to Double 2D Pointer**:
    - Converted dark line processing into a double 2D pointer structure:
        - Created `DarkLineImageData` struct to store `double**` data, rows, columns, and related attributes.
        - Replaced vector-based functions with pointer-based methods.
        - Updated threshold from `uint16_t` to `double` for greater precision.
        - Developed `DarkLinePtrArray` struct to replace vector usage for storing dark line information.
        - Adjusted method names and declarations to avoid conflicts with existing functions (e.g., `DarkLineProcessor` to `DarkLinePointerProcessor`).
        - Integrated the pointer method into the control panel under a dedicated section, labeled with "(2D Pointer)".
    - Encountered a crash when running the pointer-based method, with the issue still under investigation.
- **Zoom Control Enhancements**:
    - Updated control panel logic to block all buttons except specific zoom controls when zoom is active, with only warnings shown during fixed zoom mode.
    - Modified button re-enablement when zoom is fixed (`m_fixZoomButton` checked) to ensure all buttons except Zoom In, Zoom Out, and Reset Zoom are accessible.

Achievements

- Enhanced control panel styling for zoom functionality, providing clear color indicators for each zoom state.
- Implemented a robust system for enabling/disabling buttons based on file load status, improving user experience and avoiding accidental actions before a file is loaded.

- Successfully adapted the "remove line" function to handle lines of varying widths in one pass by dynamically adjusting `SEARCH_RADIUS`.
- Completed the transformation of dark line processing functions to use a double 2D pointer structure, laying the groundwork for more efficient memory usage.
- Activated the histogram toggle button upon file load, ensuring functionality aligns with other UI elements.

Problems & Solutions

- **Problem**: Line removal function did not remove lines in one step.
    - **Solution**: Increased `SEARCH_RADIUS` to 100 and developed a dynamic calculation method to adjust radius based on line width, ensuring single-step removal without affecting surrounding pixels.
- **Problem**: Program crash when running the new 2D pointer-based dark line processing.
    - **Solution**: Investigated pointer handling and memory allocation, ensuring structs and methods are aligned with the new pointer method; issue still under analysis.
- **Problem**: Histogram toggle button was not active after file load.
    - **Solution**: Added histogram toggle to `m_allButtons` and linked it to `fileLoaded`, ensuring the button state updates upon file loading.
- **Problem**: Button state inconsistencies when zoom is fixed.
    - **Solution**: Refined button control logic so all buttons except zoom-specific ones remain accessible in fixed zoom mode, while other controls remain disabled when zoom is active.