Objectives

1. Review and finalize the functionality of the dark line detection and removal system, including the control panel interface and output information.
2. Submit the `darkline_pointer` header and source files as per the team lead's request.
3. Improve usability and flexibility in the UI for various functions, including line management, last action display, and parameter adjustments.
4. Add advanced image processing features such as edge enhancement and combined adjustment options for improved user experience.

Activities

1. **Code and Control Panel Review:**
    - Double-checked function implementations and the corresponding control panel components.
    - Verified the information output for accuracy and alignment with user actions.
2. **DarkLine Pointer Submission:**
    - Submitted `darkline_pointer` header and source files to the team lead, completing the requested task.
3. **Reset Mechanisms and Button Management:**
    - Added `resetDetectedLinesPointer()` to ensure pointer-detected lines are deleted when switching functions.
    - Implemented a reset button specifically for the detection method lines, allowing selective resets without affecting other methods.
    - Blocked the pointer method buttons when the vector-based method is active to prevent conflicts.
4. **Line Information Management:**
    - Updated the line information box to retain data for detected or removed lines even after reverting actions.
    - Ensured users must re-detect lines after a revert operation.
5. **UI Enhancements:**
    - Adjusted the size of the last action bar and parameter bar for dynamic resizing based on content.

        - Used `QSizePolicy::Expanding` for width and `QSizePolicy::Minimum` for height.
        - Enabled word wrap to handle long text.
        - Added `updateLastActionLabelSize()` to dynamically calculate label height based on text content and available width.
6. **Feature Additions:**
    - Added an **Edge Enhancement** function using the Sobel operator with adjustable parameters.
    - Created a combined adjustments button for gamma, sharpen, and contrast, with user-selectable options for global or regional application via the UI.
    - Unified line detection, removal, and reset buttons into a single interface, allowing users to choose the detection method while ensuring consistent removal and reset workflows.

Achievements

1. Submitted the finalized `darkline_pointer` files, meeting the team lead's requirements.
2. Improved the robustness of line detection and removal functionalities with better reset and conflict-avoidance mechanisms.
3. Enhanced UI usability with dynamic resizing of key components, improved label handling, and consolidated buttons for line management.
4. Added advanced image processing features, including edge enhancement and combined adjustment options for better functionality.

Problems & Solutions

1. **Problem:** Overlap and conflict between vector and pointer methods for line detection. **Solution:** Blocked pointer-based method buttons when vector-based methods are active, requiring a reset or line removal before switching methods.
2. **Problem:** Last action and parameter bar sizes were fixed, limiting flexibility for varied content.

Solution: Enabled dynamic resizing with `QSizePolicy` and implemented content-aware height adjustments using `updateLastActionLabelSize()`.

3. **Problem:** Line information was lost when reverting operations.
   Solution: Ensured the line information box retains data for previously detected or removed lines, while requiring re-detection post-revert.
4. **Problem:** Lack of user-friendly integration for adjustment functions and line management.
   Solution: Combined gamma, sharpen, and contrast adjustments into a single interface with options for global or regional adjustments. Unified line detection, removal, and reset into a streamlined UI workflow.