Objectives

1. Enhance the modularity and efficiency of the dark line detection by transferring functions to separate files.
2. Resolve issues with specific MSVC kit compatibility causing crashes, enabling stable operation across different development environments.
3. Implement multithread processing for the CLAHE function on both CPU and GPU versions to improve processing efficiency and maintain detailed progress tracking.
4. Conduct supervisor testing on code for accuracy in mathematical logic and algorithmic functionality.

Activities

1. **Enhancement of Dark Line Detection:**
    - Moved dark line functions into dark_line.h and dark_line.cpp files.
    - Adjusted label display for detected line location and weight, now only highlighting the line without additional labels.
    - Addressed minor issues with information labels that arose from the transfer.
2. **MSVC Kit Compatibility Issue Resolution:**
    - Identified crashes when running specific functions under the MSVC kit.
    - Tested alternative kits to stabilize program functionality, with plans for further investigation to ensure cross-compatibility.
3. **Multithread Processing for CLAHE:**
    - **GPU Version of Threshold CLAHE:**
        - Developed parallel stages for 8-bit and 16-bit conversion, with asynchronous GPU processing using CUDA streams.
        - Implemented detailed tracking for multiple processing stages, including:
            - Vector to Mat conversion
            - Dark mask creation
            - GPU CLAHE processing
            - Sharpening and final blending
        - Utilized CUDA streams for efficient progress tracking.
    - **CPU Version of Threshold CLAHE:**
        - Added multithreading for key steps, with detailed progress tracking:
            - Vector/Mat conversion
            - Dark mask creation

        - Gaussian smoothing
        - CLAHE processing
        - Sharpening
        - Progressive blending and final normalization
    - Optimized thread usage with hardware_concurrency(), ensuring balanced workload distribution, thread-safe logging, and synchronized thread execution.
    - Implemented error handling and resource management for safe execution.
4. **Testing and Code Review with Supervisor:**
    - Verified logic, mathematics, and algorithmic accuracy with supervisor to ensure code reliability and correctness.

Achievements

1. Successfully modularized dark line detection by creating dedicated header and source files, simplifying code structure.
2. Temporarily resolved MSVC kit compatibility issues by using alternative kits, enabling continued development and testing.
3. Implemented multithreaded processing for CLAHE on both CPU and GPU, with robust progress tracking and efficient resource management.
4. Completed supervisor code testing, verifying accuracy in mathematical and algorithmic computations.

Problems and Solutions

Problem 1: Label information was not correctly displayed after transferring dark line functions to new files.

Solution 1: Adjusted the label display to remove redundant information, displaying only highlighted lines and refining the info label handling.

Problem 2: Crashes occurred when using the MSVC kit, interrupting program stability.

Solution 2: Switched to alternative kits to avoid crashes temporarily, with plans for additional debugging on MSVC compatibility.

Problem 3: Multi-thread processing debug for Sharpen, Gamma, and Contrast needed to be removed temporarily due to function kit issues.

Solution 3: Disabled multi-thread debug for these functions until MSVC kit stability can be assured across all modules.

Problem 4: Progress tracking and efficient synchronization were challenging for the GPU and CPU versions of CLAHE processing.

Solution 4: Implemented CUDA streams for GPU, and hardware_concurrency() for CPU, with detailed progress logging to ensure effective parallel processing and workload balance across threads.