Task Progress Update Report

Name: LIM SHI KAI (Sky)

Update Date: 24-10-2024

---

1. Overview of Assigned Tasks

Task 1: Adjust calibration for the X and Y axes

- **Objective:** Adjust the calibration of the X and Y axes by referring to Phang's code without directly copying it.
- **Assigned On:** 07-10-2024
- **Status:** Done

Task 2: Implement CLAHE (Contrast Limited Adaptive Histogram Equalization)

- **Objective:** Apply CLAHE to the output for image enhancement.
- **Assigned On:** 07-10-2024
- **Status:** Done

Task 3: CLAHE with GPU Implementation

- **Objective:** Integrate CLAHE functionality using GPU acceleration.
- **Assigned On:** 17-10-2024
- **Status:** Done

Task 4: Apply Threshold to CLAHE for Dark Areas

- **Objective:** Apply CLAHE selectively to dark regions, leaving bright regions unchanged.
- **Assigned On:** 17-10-2024
- **Status:** In Progress

Task 5: OOP Refactoring

- **Objective:** Refactor the code using Object-Oriented Programming principles for better organization.
- **Assigned On:** 17-10-2024
- **Status:** Done

2. Progress as of 24-10-2024

Task 1: Calibration for the X and Y Axes

- **Current Status:** Completed
- **Details:**
  - The calibration for both X and Y axes has been successfully adjusted. The method adapts from Phang's code, and calibration was demonstrated effectively with test data.
  - Attached:

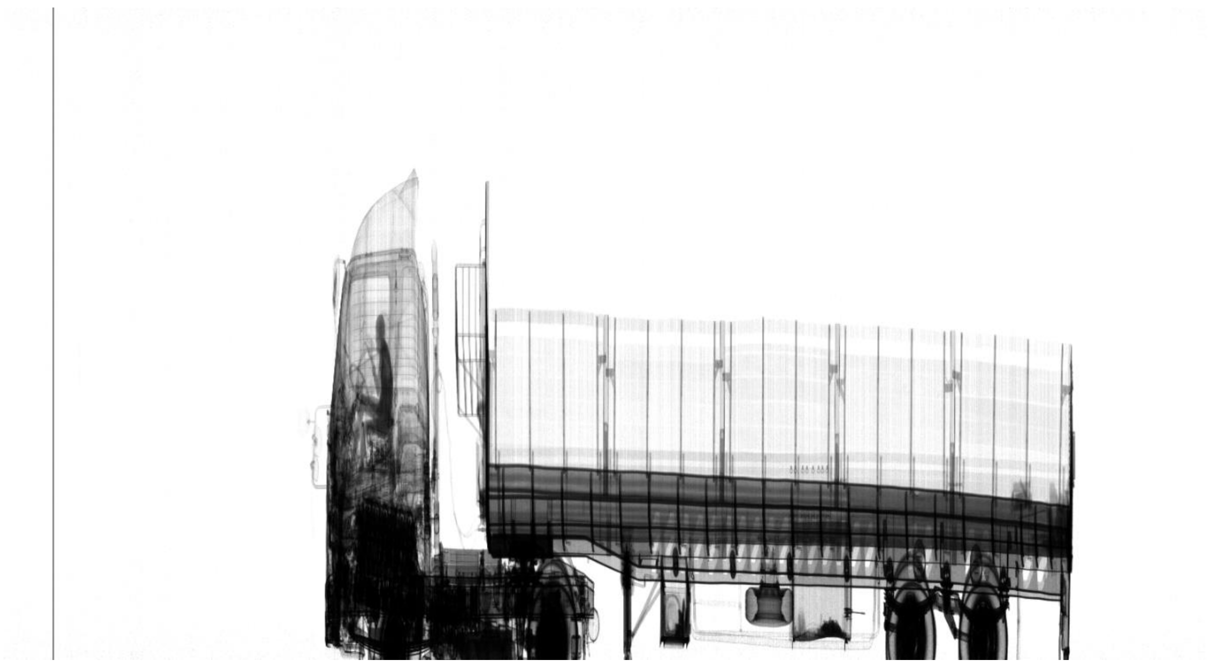
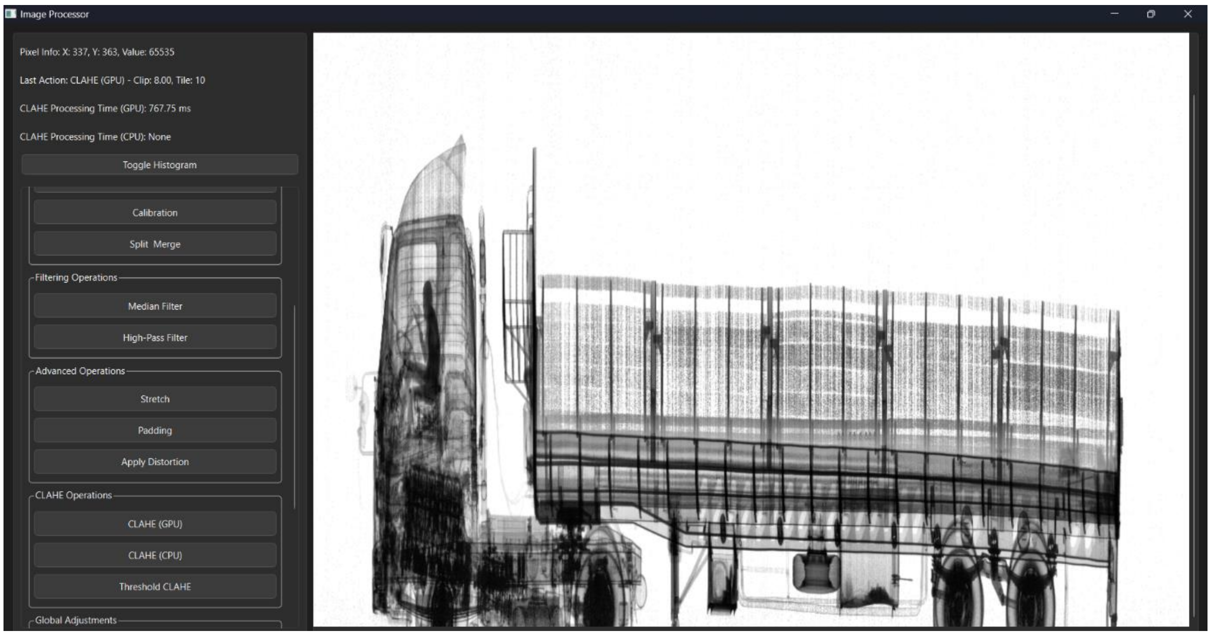


Figure 2.1 Image after 100 lines mean for X and Y-axis

Task 2: CLAHE with GPU Implementation

- **Current Status:** Completed
- **Details:**
    - CLAHE has been successfully implemented with GPU acceleration using OpenCV and CUDA. The processing time is displayed alongside the output for performance analysis.
    - Attached:

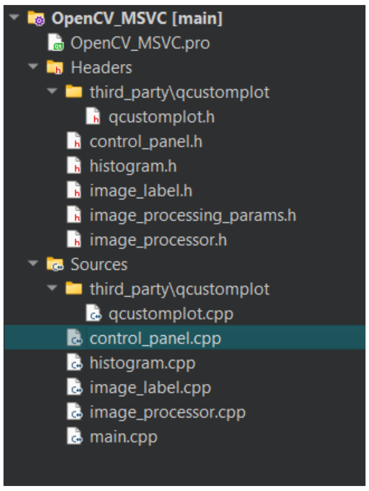


```
Pixel Info: X: 337, Y: 363, Value: 65535
Last Action: CLAHE (GPU) - Clip: 8.00, Tile: 10
CLAHE Processing Time (GPU): 767.75 ms
CLAHE Processing Time (CPU): None
```

Figure 2.2 Effect after CLAHE with GPU, processing time: 767.75ms

Task 3: OOP Refactoring

- **Current Status:** Completed
- **Details:**
    - Code refactored into an object-oriented structure for better organization and scalability. Key functions such as CLAHE and calibration are now encapsulated within distinct classes.
    - Attached:



```
OpenCV_MSVC [main]
    OpenCV_MSVC.pro
  Headers
    third_party\qcustomplot
      qcustomplot.h
    control_panel.h
    histogram.h
    image_label.h
    image_processing_params.h
    image_processor.h
  Sources
    third_party\qcustomplot
      qcustomplot.cpp
    control_panel.cpp
    histogram.cpp
    image_label.cpp
    image_processor.cpp
    main.cpp
```

Figure 2.3 Project Roof Directory

Task 4: CLAHE with Threshold

- **Current Status:** In Progress
- **Details:**
  - The initial approach used input parameters (threshold, clipLimit, tiles) to apply CLAHE but produced unsatisfactory results due to conceptual flaws.
  - The current approach during the progress update involves creating a mask to detect regions below a user-defined threshold, applying CLAHE to the entire image, and merging the mask, CLAHE-processed regions, and the original image. Regions below the threshold undergo CLAHE, while regions above remain unchanged.
  - Currently, this method runs in CPU mode and is under further testing and adjustment.
  - Attached: **Note: Threshold: 5000, ClipLimit: 2, Tiles: 80**

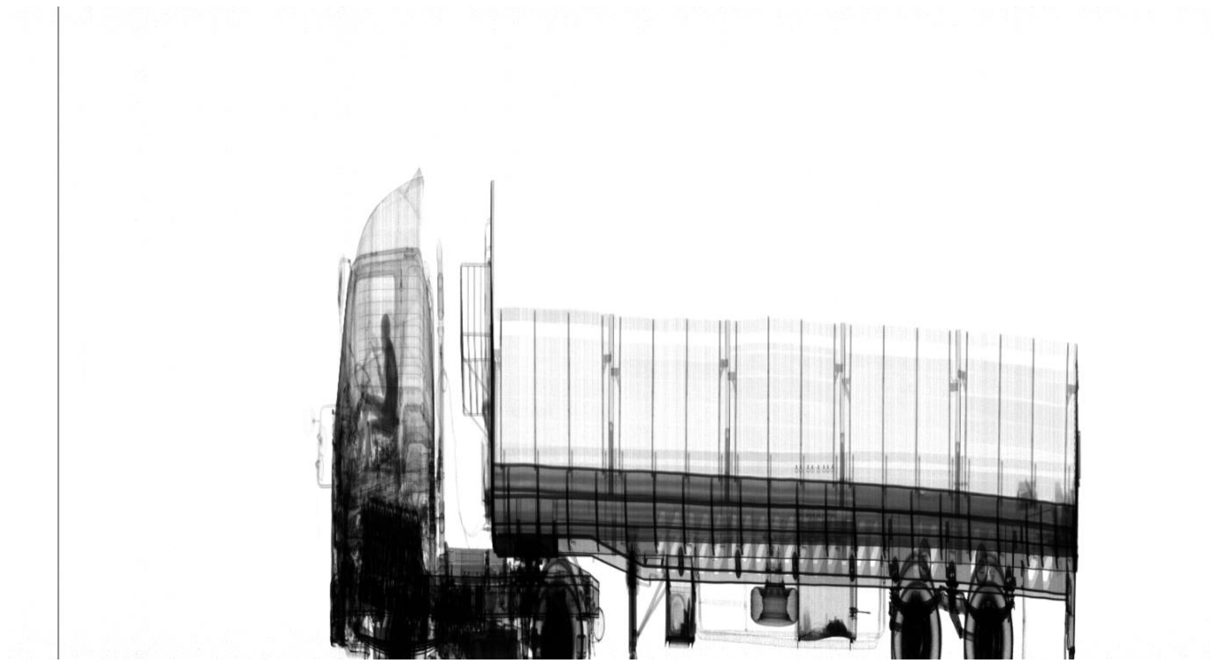


Figure 2.4 Original Image before Processing

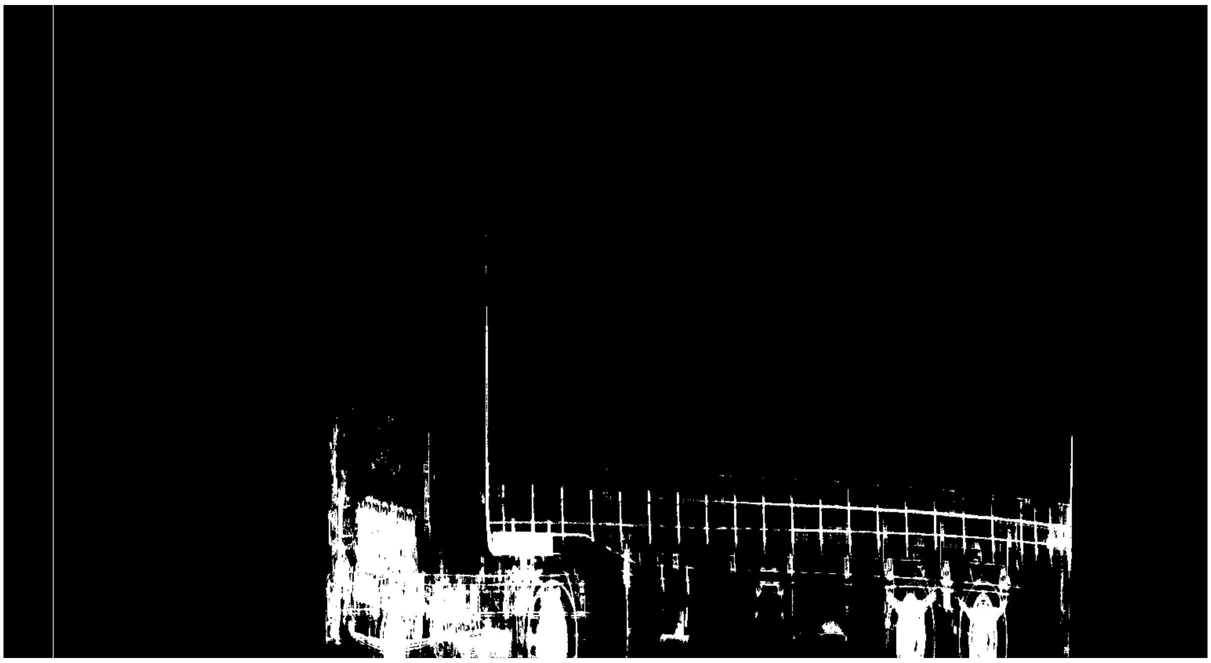


Figure 2.5 Dark Mask created based on the threshold input (Below the threshold will be in White)

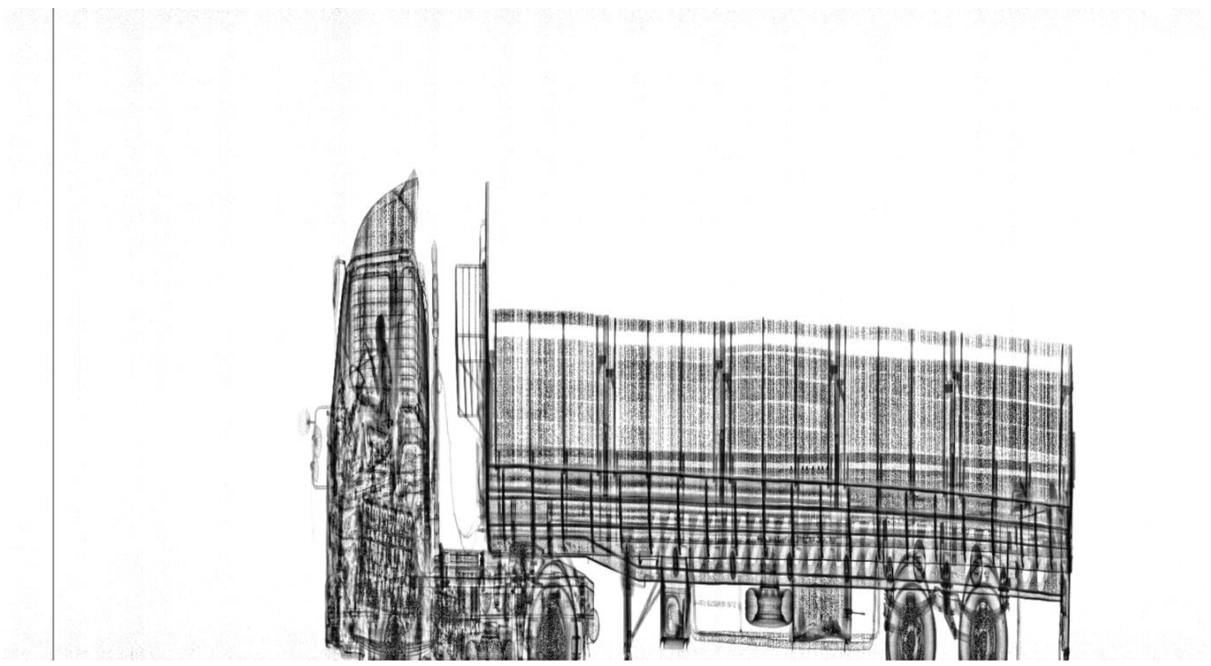


Figure 2.6 Overall CLAHE Result based on input

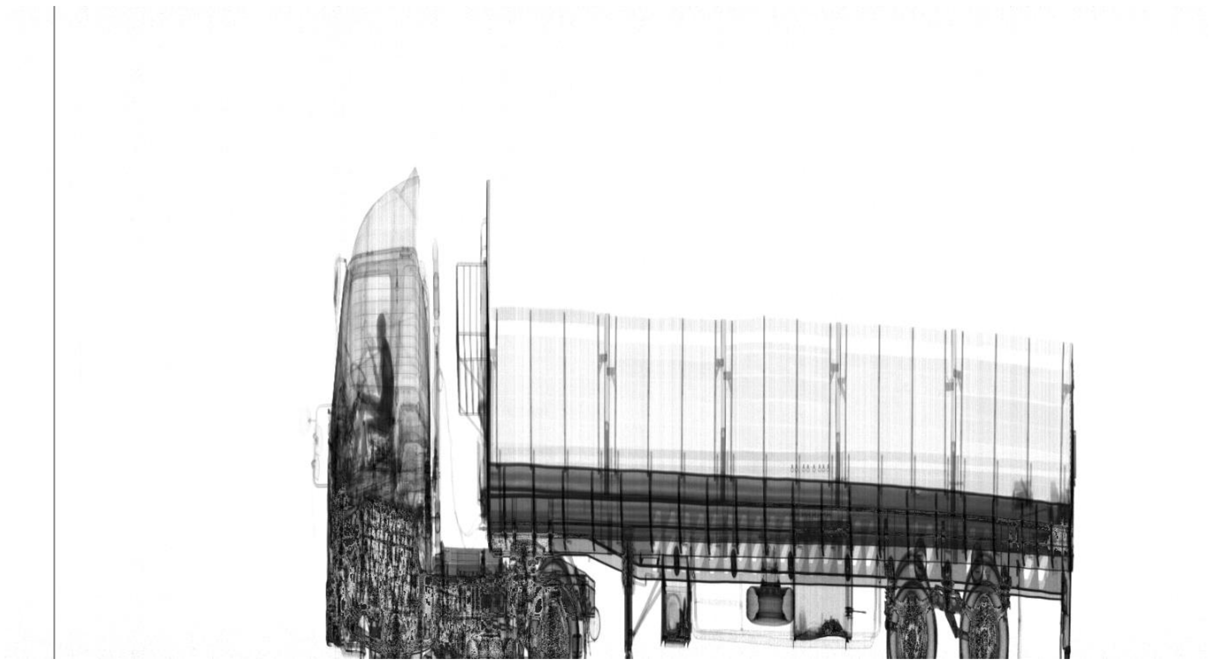


Figure 2.7 Final Result after merging with mask, original image and CLAHE output

3. New Tasks (Received on 24-10-2024)

Task 1: Show Thread Processing

- **Objective:** Display the thread processing for each row or column, including the thread ID.

Task 2: Add CLAHE in CPU Mode

- **Objective:** Add a function to apply CLAHE in CPU mode and display the processing time for comparison with the GPU implementation.

Task 3: Remove Vertical Black Lines

- **Objective:** Implement a function to remove vertical black lines in non-object regions. If the lines are near objects or vehicles, they should remain.
  - Attached:

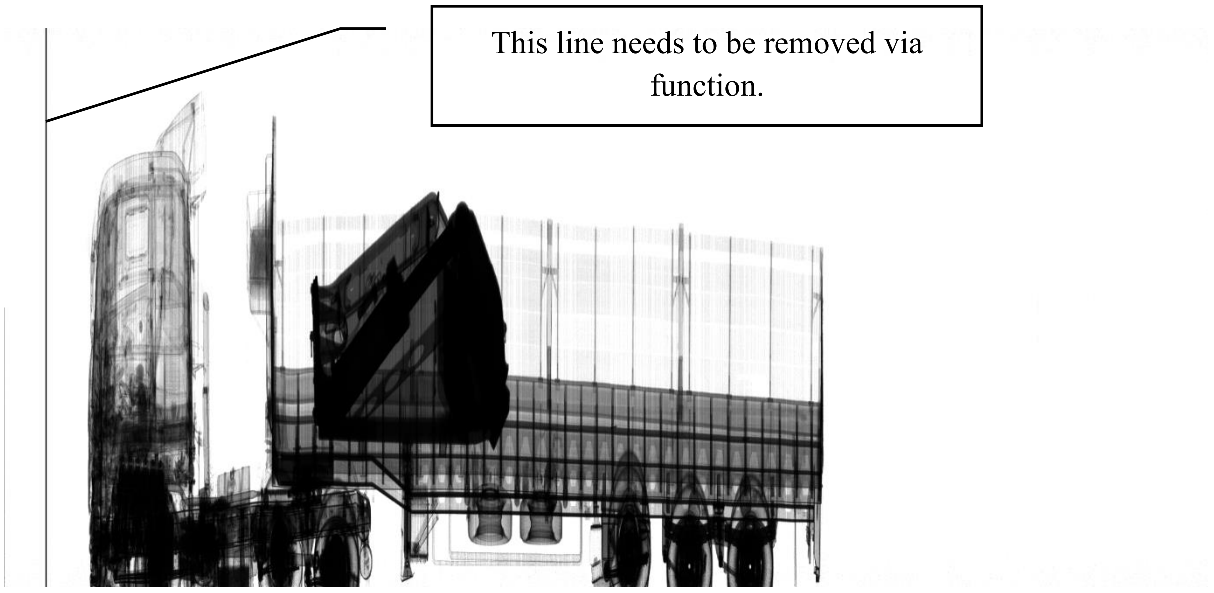


Figure 3.1 Sample Image for Task

Task 4: CLAHE with Threshold

- **Objective**: Apply CLAHE only to regions below a user-defined intensity threshold, leaving the rest of the image unchanged.
- Details: The new approach now involves storing the pixels below the user-defined threshold in a vector, processing them with CLAHE, and then returning them to the overall image. This ensures that CLAHE is applied only to the regions below the threshold while the rest of the image remains untouched. This method is currently being tested in CPU mode.

4. Roadblocks/Challenges

- **CLAHE with Threshold:**
  - The current method faces issues effectively merging the CLAHE regions and the original image. Ensuring the CLAHE effect is applied only in dark regions while preserving the rest of the image requires further refinement.
  - Mask generation works correctly, but the final visual result is not optimal.

5. Conclusion

- X and Y axis calibration completed, with effective adaptation from Phang's method.
- CLAHE implementation with GPU acceleration completed, showing improvements in processing time.
- Code refactored using Object-Oriented Programming principles, improving modularity and scalability.
- CLAHE with threshold concept revised:
    - The needed method is storing the pixels below the threshold in a vector, processing them with CLAHE, and returning them to the image.
    - The current method shows the Mask generation works correctly, but further refinement is needed to merge CLAHE-processed regions with the original image.
- Challenges remain with the CLAHE threshold approach, specifically achieving optimal visual results.
- The project progresses well, with the remaining tasks clearly defined and focused on improving the CLAHE threshold function.