Sheet Stack Counter

Overall Approach

The primary goal of the Sheet Stack Counter application is to automate the counting of sheet stacks in a manufacturing plant using image processing techniques. The approach involves:

1. **Image Upload and Resizing**: Users upload an image, which is then resized to a standard dimension for consistent processing.
2. **Image Preprocessing**: The image is converted to grayscale, blurred to reduce noise, and edges are detected using the Canny edge detection algorithm.
3. **Line Detection**: The Hough Line Transform is used to detect lines in the edge-detected image. Users can choose to use custom parameters or let the application find the optimal parameters automatically.
4. **Counting Lines**: The application counts the detected lines, representing the number of sheets.
5. **Results Display**: The processed images and the count of detected lines and sheets are displayed to the user.

Frameworks/Libraries/Tools

- **Streamlit**: Used for building the web interface of the application, allowing users to upload images and view results.
- **OpenCV**: Used for image processing tasks such as resizing, grayscale conversion, Gaussian blur, edge detection, and line detection.
- **NumPy**: Utilized for numerical operations and handling image data as arrays.
- **Pillow (PIL)**: Used for image handling, particularly for converting NumPy arrays to image formats suitable for display with Streamlit.

Challenges and Solutions

Challenge 1: Parameter Selection for Line Detection

- **Solution**: Implemented a function (find_best_parameters) to automatically find the optimal parameters for line detection by iterating over different thresholds and maximum line gaps. This helps in maximizing the number of detected lines.

Challenge 2: Handling Noise and Irrelevant Lines

- **Solution**: Applied Gaussian blur to reduce noise in the image and set a minimum line length to filter out irrelevant or too-short lines that do not represent actual sheets.

Challenge 3: Ensuring User Flexibility

- **Solution**: Provided options for users to either use custom parameters or let the application find the optimal parameters. This gives flexibility to users based on their specific requirements and image characteristics.

Future Scope

1. Enhanced Image Preprocessing

- **Improvement**: Implement more advanced image preprocessing techniques such as adaptive thresholding, morphological operations, and edge smoothing to improve edge detection accuracy.

2. Deep Learning Integration

- **Improvement**: Integrate deep learning models for more accurate line detection and sheet counting, especially for more complex and varied images.

3. Batch Processing

- **Feature**: Allow users to upload multiple images at once and process them in batch mode, providing aggregate results and comparisons.

4. Real-Time Processing

- **Feature**: Develop capabilities for real-time image processing and counting, which can be integrated into manufacturing processes for continuous monitoring and counting.

5. Enhanced User Interface

- **Improvement**: Improve the user interface with more interactive elements, detailed instructions, and better visualization of results.