Sheet Counter Application Documentation

1. Overall Approach

Objective

The primary objective of this application is to count the number of sheets in a stack from an uploaded image. The approach leverages computer vision techniques to identify and count horizontal lines that correspond to the sheets' edges.

Methodology

1. **Image Preprocessing**: Convert the image to grayscale and apply Gaussian blur to reduce noise.
2. **Edge Detection**: Use the Canny edge detection method to highlight the edges of the sheets.
3. **Line Detection**: Apply the Hough Line Transform to detect lines in the image, focusing on horizontal lines.
4. **Line Filtering and Counting**: Filter out non-horizontal lines and count the remaining lines, ensuring each line represents one sheet.

2. Frameworks/Libraries/Tools

- **OpenCV**: Used for image processing tasks including reading the image, converting it to grayscale, blurring, edge detection, and line detection.
- **NumPy**: Utilized for numerical operations and array manipulations.
- **Matplotlib**: Used for displaying images and plots during the development process.
- **Math**: Provides mathematical functions such as degree conversion and tangent calculations for slope analysis.
- **Streamlit**: A web application framework used to create a simple interface for users to upload images and view results.
- **PIL (Python Imaging Library)**: Used for image handling and conversion to ensure compatibility with OpenCV.

3. Challenges and Solutions

Challenge 1: Accurate Line Detection

- **Issue**: Detecting only the horizontal lines representing the sheets and avoiding noise or irrelevant lines.
- **Solution**: Applied a combination of Gaussian blur and Canny edge detection with carefully tuned parameters. Additionally, filtered lines by their slope to ensure they are nearly horizontal.

Challenge 2: Handling Different Image Qualities

- **Issue**: Variability in image quality could affect the accuracy of line detection.
- **Solution**: Used adaptive thresholding in the Canny edge detection and ensured the parameters of the Hough Line Transform were adaptable to different image conditions.

Challenge 3: User Interface for Image Upload and Result Display

- **Issue**: Providing a simple yet effective interface for users to upload images and get results.
- **Solution**: Implemented a Streamlit application that allows users to upload images and immediately see the count of detected sheets.

4. Future Scope

Improvements

- **Parameter Optimization**: Further tuning of the parameters used in Gaussian blur, Canny edge detection, and Hough Line Transform to improve accuracy across a wider range of images.
- **Advanced Line Filtering**: Implementing more sophisticated line filtering techniques using machine learning to better differentiate between sheet edges and noise.
- **Real-time Processing**: Extending the application to handle real-time video input, enabling the counting of sheets as they are being stacked.

Additional Features

- **Multi-language Support**: Expanding the application to support multiple languages for broader accessibility.
- **Batch Processing**: Allowing users to upload multiple images at once and receive a summary report of the sheet counts.
- **Integration with Cloud Services**: Storing and processing images on cloud platforms to enhance performance and scalability.