Sheet Counter

4th August 2024

Ahmad Meda
Final Year Student at Heriot-Watt University
BSc Computer Science

Table of Contents

Sheet Counter 1

Introduction 3

Task 3

Objectives 3

Implementation 3

Frameworks & Libraries 3

Problems Faced 4

Future Scope 4

Introduction

Task

In today's fast-paced world, the process of counting sheet stacks is often performed manually. This method is time-consuming and prone to errors, leading to inefficiencies and potential inaccuracies in inventory management and production processes.

The objective of this project is to develop an application that automates the counting of sheet stacks in a manufacturing plant. By leveraging computer vision techniques, the goal is to create a solution that accurately counts the number of sheets from images of stacks, thereby reducing manual effort and minimizing errors.

The application should be capable of:

1. Implementing image preprocessing techniques to enhance accuracy (e.g., edge detection, contrast adjustment).
2. Using traditional image processing methods for sheet counting such as OpenCV, etc.
3. Providing a user-friendly interface where users can upload images and get the sheet count as output.
4. Ensuring the solution is efficient and accurate with minimal latency

Objectives

To address the problem statement, the following objectives were set for the project:

- **Image Preprocessing:** Convert the uploaded image to grayscale and apply Gaussian blur to reduce noise. Then, use edge detection (Canny) to highlight the edges of the sheets.
- **Sheet Counting**: Identify contours in the preprocessed image and filter these contours to count the horizontal lines representing the sheets.
- **Fast Response time**: Display the results with minimum latency.
- **Web Application**: Create a web interface where users can upload images and receive the sheet count as output. The application uses Flask to handle file uploads and image processing.

Implementation

The sheet counting application processes images by converting them to grayscale and applying a Gaussian blur to reduce noise. The Canny edge detection algorithm is then used to highlight edges. Contours are identified, and horizontal lines representing the sheets are counted. Developed using Flask, the application provides a user-friendly web interface for uploading images and receiving the sheet count. HTML, CSS, JavaScript, and Tailwind CSS are used for design and responsiveness. Flask handles file uploads and image processing efficiently.

Frameworks & Libraries

- Flask: - Used as the web framework to create routes and serve the HTML template. - Handles HTTP requests and responses.
- **OpenCV**: Used for image processing, including grayscale conversion, Gaussian blur, edge detection, and contour finding.
- **HTML/CSS**: Used for designing the user interface.

- **JavaScript:** Used for creating dynamic and responsive elements in the web interface.
- **Tailwind CSS**: A utility-first CSS framework used for styling the web interface.

Problems Faced

Challenge: Identifying horizontal lines accurately.
- ***Solution***: Applied Gaussian blur to reduce noise and then used Canny edge detection to highlight edges. By carefully filtering contours, only the horizontal lines are counted.

Challenge: Handling various image qualities and formats.
- ***Solution***: Implemented image preprocessing techniques to standardize the input images, improving the robustness of the edge detection and contour finding processes.

Challenge: Ensuring the web application is user-friendly.
- ***Solution***: Designed a simple and intuitive user interface using HTML, CSS, and JavaScript. Used Tailwind CSS for consistent and responsive styling.

Future Scope

- **Improved Accuracy**: Implement advanced image processing techniques or deep learning models to improve the accuracy of sheet counting, especially for images with complex backgrounds or varying lighting conditions.
- **Batch Processing**: Enable batch processing of multiple images at once to save time for users who need to count sheets in several images.
- **Mobile Application**: Develop a mobile application for on-the-go usage, allowing users to take photos of sheet stacks and get immediate results.
- **Real-Time Video Processing:** Implement real-time processing of video streams to count sheets as they are stacked or moved in the manufacturing plant.