Vehicle License Plate Recognition System

Contents

Overview

This project aims to develop an automated system for detecting Romanian vehicle license plates from frontal images. The primary application of this system is in traffic monitoring, parking management, and law enforcement. The system should accurately identify and highlight the area where the license plate is located within an image.

Project Plan

Step 1: Image Preprocessing

- **Task:** Convert input images to grayscale.
- **Integration:** This step ensures that the image is easier to process for edge detection.
- **Deliverables:** Preprocessed grayscale images.
- **Potential Issues:** Images with poor lighting or reflections may degrade detection performance.

1

Step 2: Edge Detection

- **Task:** Use an edge detection algorithm to highlight significant contours.
- **Integration:** Edge detection helps segment objects in the image, including the license plate.
- **Deliverables:** Edge-detected images.
- **Potential Issues:** Too many edges are detected.

Step 3: Contour Detection and Filtering

- **Task:** Identify contours in the edge-detected image and filter them based on shape and size constraints to detect potential license plates.
- **Integration:** This step narrows down the search to candidate regions that resemble a license plate.
- **Deliverables:** Bounding rectangles around candidate regions.
- **Potential Issues:** Some objects in the image may have similar aspect ratios.

Step 4: License Plate Localization

- **Task:** Draw a bounding box around the identified license plate and display the result.
- **Integration:** Finalizes the detection and provides visual feedback.
- **Deliverables:** Marked images with license plate detections.
- **Potential Issues:** The algorithm may miss plates if they are occluded or distorted. Future improvements may involve machine learning-based object detection.

Implementation

The license plate recognition system is implemented using OpenCV in C++. The overall architecture follows a modular image processing pipeline, where each stage transforms the input to bring it closer to the desired goal: locating the license plate region in a vehicle image.

Module 1: Grayscale Conversion

- **Function:** bgr _2 grayscale(Mat img)
- **Description:** Converts the original BGR image to a grayscale image. This simplification reduces data complexity and removes color information that is not relevant to edge detection.
- **Design Rationale:** Grayscale images are computationally cheaper to process and suitable for edge-based operations.

Module 2: Image Smoothing (Box Blur)

- **Function:** box _blur(Mat img, int size)
- **Description:** Applies a box blur to reduce noise and soften the image, which helps reduce false edges in the next stage.
- **Design Rationale:** Noise can cause false positives in edge detection. A simple box blur helps by averaging pixel values in a neighborhood.

Module 3: Edge Detection

- **Function:** sobel edge(Mat img)
- **Description:** Applies the Sobel operator manually in both x and y directions to compute gradient magnitude and highlight image contours.
- **Design Rationale:** Sobel is a standard and interpretable method for detecting horizontal and vertical edges, which aligns well with the rectangular nature of license plates.

Module 4: Thresholding

- **Function:** threshold _binary(Mat img, uchar t)
- **Description:** Converts the edge-detected image into a binary image using a fixed threshold. Pixels above the threshold are set to white (255), and those below to black (0).
- **Design Rationale:** Binary images make it easier to identify regions of interest through connected components.

Module 5: Plate Detection

- **Function:** detect plate(Mat img)
- **Description:** Uses a breadth-first search (BFS) flood-fill algorithm to find connected white regions in the binary image. For each region, it calculates a bounding box and evaluates its area and aspect ratio.
- **Filtering Criteria:** The candidate region is selected based on:
  - Area (must exceed a threshold), – Aspect ratio (between 2.0 and 7.0),
  - Minimum size (width: 50, height: 15).
- **Design Rationale:** License plates typically have a consistent rectangular aspect ratio. These heuristics filter out non-relevant regions.

Testing

To evaluate the performance and robustness of the vehicle license plate detection system, we designed several test cases using different images representing a range of real-world conditions. The goal is to verify whether the system correctly identifies and localizes license plates within diverse images.

Test Methodology

1. **Manual Ground Truth Annotation**
   For each test image, a bounding box was manually drawn around the correct license plate location. This serves as the ground truth for comparison.

2. **Visual Output Comparison**
   The system output overlays two rectangles on the image:

   - **Green** for the ground truth (manual annotation)
   - **Red** for the predicted license plate bounding box

3. **IoU (Intersection over Union) Metric**
   We calculated the IoU between the ground truth and detected rectangles. This metric ranges from 0 (no overlap) to 1 (perfect match). IoU ≥ 0.5 is considered acceptable for successful detection.

Result: 0.94, Success.

Analysis: The plate was detected accurately. Minimal background noise and clear edge definition helped the Sobel operator perform optimally.

Result: 0.83, Success.