OpenALPR Design

matthill edited this page 4 days ago · 1 revision

OpenALPR Design

OpenALPR operates as a pipeline. The input is an image, various processing occurs in stages, and the output is the possible plate numbers in the image.

The pipeline stages occur in the following order:

Pipeline Phase	C++ class	Description
Detection	regiondetector.cpp	Finds potential license plate regions
Binarization	binarizewolf.cpp	Converts the plate region image into black and white
Char Analysis	characteranalysis.cpp	Finds character-sized "blobs" in the plate region
Plate Edges	platelines.cpp and platecorners.cpp	Finds the edges/shape of the license plate
Deskew	licenseplatecandidate.cpp	Transforms the perspective to a straight-on view based on the ideal license plate size.
Character Segmentation	charactersegmenter.cpp	Isolates and cleans up the characters so that they can be processed individually
OCR	ocr.cpp	Analyzes each character image and provides multiple possible letters/confidences
Post Processing	postprocess.cpp	Creates a top n list of plate possibilities based on OCR confidences. Also performs a Regex match against region templates if requested.

Detection

The detection phase happens one time for each input image. It uses the LBP algorithm (generally used for face detection) to find possible license plate regions (x,y, width, height). Each of these regions is sent to the later pipeline phases for further processing.

The detection phase is usually the most processing-intensive phase. It can be GPU accelerated to improve performance.

Binarization

Pages 6
Android compilation
Compilation instructions (Ubuntu Linux)
Compilation instructions (Windows)
Home
OpenALPR Daemon (alprd)
OpenALPR Design

This phase (and all subsequent phases) occur multiple times -- once for each possible license plate region.

The binarization phase creates multiple binary images for each plate region. The reason multiple binary images are used is to give us the best possible chance of finding all the characters. A single binarized image may miss characters if the image is too dark or too light for example. Binarization uses the Wolf-Jolien method as well as the Sauovola method with various parameters. Each of the binary images are processed in subsequent phases.

Character Analysis

Character analysis attempts to find character-sized regions in the plate region. It does this by first finding all connected blobs in the license plate region. Then it looks for blobs that are roughly the width and height of a license plate character and have tops/bottoms that are in a straight line with other blobs of similar width/height.

This analysis is done multiple times in the region. It starts by looking for small characters, then gradually looks for larger characters.

If nothing is found in the region, then the region is thrown out and no further processing takes place. If it finds some potential characters, then the character region is saved and further processing takes place.

Plate Edges

The next phase is to find the edges of the license plate. Keep in mind that the detection phase is only responsible for identifying a possible region where a license plate may exist. It often is going to provide a region that is a little larger or smaller than the actual plate. The plate edges tries to find the precise top/bottom/left/right edges of the license plate.

The first step is to find all of the hough lines for the license plate region. platelines.cpp processes the plate image and computes a list of horizontal and vertical lines.

platecorners uses this list as well as the character height (computed in Character Analysis) to find the likeliest plate line edges. It uses a number of configurable weights to determine which edge makes the most sense. It will try using a default edge (based on the ideal width/height of the plate) to see if that makes a good match.

Deskew

Given the plate edges, the deskew stage remaps the plate region to a standard size and orientation. Ideally this will give us a correctly oriented plate image (no rotation or skew).

Character Segmentation

The character segmentation phase tries to isolate all the characters that make up the plate image. It uses a vertical histogram to find gaps in the plate characters. This phase also cleans up the character boxes by removing small, disconnected speckles and disqualifying character regions that are not tall enough. It also tries to remove "edge" regions so that the edge of the license plate doesn't inappropriately get classified as a '1' or an 'I'

OCR

The OCR phase analyzes each character independently. For each character image, it computes all possible characters and their confidences.

Post Processing

Given a list of all possible OCR characters and confidences, post processing determines the best possible plate letter combinations. It is organized as a top N list. Post processing disqualifies all characters below a particular threshold. It also has a "soft" thresholds -- characters that are below this threshold will still be added to the possible list, but they also add a possible blank character -- since it's possible that the low confidence character is not really part of the plate.

The post processing also handles region validation if requested. For example, if I tell OpenALPR that this is a "Missouri" plate, then it will try and match the results against a template that matches the Missouri format (e.g., [char][char][number]-[char][number][char]). So, for example, if the top 3 list was:

- CFOCIG
- CF0CIG
- CF0C1G

The third entry matches the template, but the other two do not. So, post processing will signal that the third entry is our best match.