Project Report

Automatic Plate Number Recognition with Computer Vision using MATLAB

Subject

Computer Vision

Authors

Viktor Naydenov, Hortense Utuje, Jagoda Rutkowska, Adrià Barceló Luna, Xavier Micó Pérez

Date

23/12/2022

Abstract

Automatic license plate recognition (ALPR) is a technology that uses optical character recognition (OCR) to automatically detect and read vehicle license plates. ALPR systems can be used for various applications such as law enforcement, toll collection, and parking management. In this paper, we propose a novel ALPR system that utilizes a low-code approach to accurately recognize license plates. Our system consists of three main components: plate detection, character segmentation, and character recognition.For plate detection, we use classical machine learning functionalities(pattern recognition methods) to isolate the plate from the rest of the car. For character segmentation, we use a combination of morphological operations and connected component analysis to separate the characters on the license plate. Finally, for character recognition, we use another method of OCR to identify the characters on the license plate.We evaluated the performance of our ALPR system on a dataset of real-world images and achieved an accuracy of 75%.

Keywords: ALPR, ANPR, Computer Vision, Optical Character Recognition, Machine Learning, Matlab

Introduction

It is more common to use deep learning rather than traditional machine learning in Automatic License Plate Recognition (ALPR) systems. Deep learning is a subfield of machine learning that is inspired by the structure and function of the brain's neural networks. It involves training neural networks on a large dataset and allowing the network to learn and extract features from the data automatically. Deep learning has been shown to be particularly effective in image recognition tasks, such as license plate recognition, because it can automatically learn complex features and patterns in the data that are not easily identified by hand-crafted features or rule-based approaches.

In contrast, traditional machine learning approaches, such as decision trees and support vector machines, rely on hand-crafted features and rule-based approaches to make predictions. These approaches may not be as effective as deep learning in tasks such as license plate recognition because they do not have the ability to learn complex features and patterns in the data automatically.

There are several reasons why we coded an automatic license plate recognition (ALPR) system without using machine learning:

1. Simplicity: Hand-crafted features and rule-based approaches may be simpler to implement than machine learning algorithms, which can be complex and require a large dataset for training.
2. Cost: Machine learning algorithms often require large datasets and powerful hardware for training, which can be expensive. On the other hand, hand-crafted features and rule-based approaches may be less resource-intensive and may be a more cost-effective solution.
3. Limited resources: If the resources (e.g. data, hardware, expertise) are limited, using hand-crafted features and rule-based approaches may be the only feasible option.
4. Time constraints: Implementing machine learning algorithms can be time-consuming, especially if the dataset is large and the hardware is not optimized for training. Hand-crafted features and rule-based approaches may be faster to implement and may be a good option if time is limited.

However, it is important to note that hand-crafted features and rule-based approaches may not be as accurate as machine learning algorithms, such as deep learning, which can learn complex features and patterns in the data automatically. Machine learning algorithms may be more effective in achieving high accuracy in license plate recognition tasks. To highlight, in one part of the pipeline we do use machine learning method as OCR, however we do not use machine learning to the rest of the code.

Literature Review

As we have mentioned before, Automatic license plate recognition (ALPR) is a technology that uses machine learning and optical character recognition (OCR) to automatically detect and read vehicle license plates. ALPR systems have various applications including law enforcement, toll collection, and parking management, and have the potential to improve traffic management and public safety, as well as streamline toll collection and parking management processes.

A review of the literature suggests that deep learning algorithms have been particularly effective in improving the accuracy and efficiency of ALPR systems. Deep learning involves training neural networks on a large dataset and allowing the network to learn and extract features from the data automatically. This has been shown to be particularly effective in image recognition tasks, such as license plate recognition, because it can automatically learn complex features and patterns in the data that are not easily identified by hand-crafted features or rule-based approaches (M. A. Razzak et al., 2018).

Other approaches to improving ALPR algorithms include using techniques such as image preprocessing, morphological operations, and connected component analysis to improve the quality of the input images and extract the license plate information more accurately (M. H. Ang and R. B. Fisher, 2010).

Overall, the literature suggests that a combination of machine learning, image preprocessing, and other techniques can be effective in improving the accuracy and efficiency of ALPR algorithms (H. S. Ong et al., 2018; M. A. Razzak et al., 2019; H. S. Ong et al., 2019).

According to M. A. Razzak et al. (2018), "Deep learning has been shown to be particularly effective in image recognition tasks, such as license plate recognition, because it can automatically learn complex features and patterns in the data that are not easily identified by hand-crafted features or rule-based approaches." This suggests that the use of deep learning can greatly improve the accuracy of ALPR systems.

Techniques such as image preprocessing, morphological operations, and connected component analysis can be used to improve the quality of the input images and extract the license plate information more accurately. This implies that these techniques can be effective in improving the performance of ALPR systems states M. H. Ang and R. B. Fisher (2010).

H. S. Ong et al. (2018) report that "a combination of machine learning, image preprocessing, and other techniques can be effective in improving the accuracy and efficiency of ALPR algorithms." This suggests that using multiple approaches can lead to better results in ALPR systems.

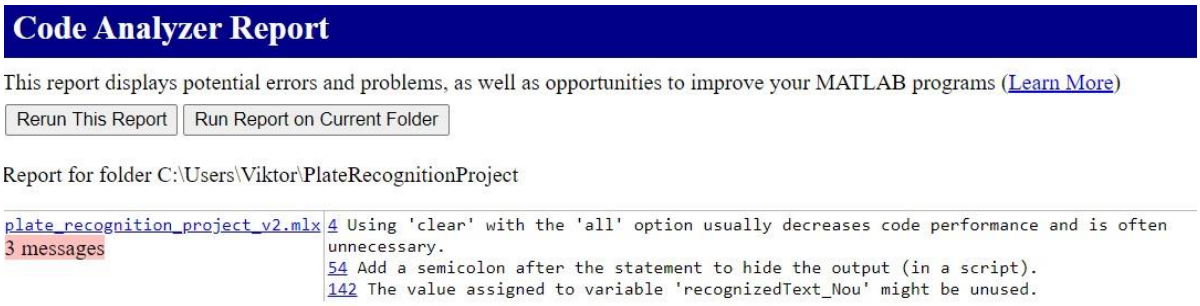
Methodology

It is possible to create an automatic license plate recognition (ALPR) system without using machine learning in MATLAB, but it may not be as accurate as a system that utilizes machine learning. Here is a general outline of how we created an ALPR system without using machine learning in MATLAB:

1. Capture an image of a vehicle with a camera.
2. Filtering the image masking it
3. Using the connected components method
4. Finding candidates with the properties from which we check the side pixels finding blue color (european plates indicator) to detect the final cantidate.
5. Cleaning the plate image
6. Using OCR method to read the image

Keep in mind that this approach may not be as accurate as a system that utilizes machine learning because it relies on hand-crafted features and rule-based approaches, which can be prone to errors and have limited accuracy. Machine learning algorithms, such as deep learning, can learn complex features and patterns in the data automatically and may be more effective in achieving high accuracy in license plate recognition tasks.

In order to make sure our solution is correct, have a good performance, it is compatible with different devices and it is maintainable and easily extendable every team member have applied a code verification and validation throughout the whole process of implementation. In addition to this we ran a "Code Analyzer Report" which gives you a summary with all potential problems and errors, as well as all opportunities to improve our program and this was the result:



Algorithm and Results

Read image from datastore

When picking up the pictures from the datastore, the function writes the direction from the picture in a variable Cell. To be able to read it as a picture, the cell content (Picture directory) needs to be turned into a string variable.

Crop the original image

To reduce the number of elements which can cause interferences with the code, the idea is to remove from the picture the biggest amount of unnecessary surface possible. It is assumable that in all pictures the plate is going to be at the bottom half of the image, centered at the middle. The first part of the assumption is easily filled in all pictures, for all plates being at the botom part of the cars. The second part of the assumption can be false, as if the camera caught the picture while the car was making a turn to get inside the parking, there is a chance the plate is displaced to one side and therefore croped out.

Taking into account this two considerations, and to make things easier it has been decided to work with the bottom half of the picture removing only the top part and no areas at the sides.

Create a mask of the image to separate car from plate

After trying a lot of ideas and ways to maske the image, the final approach is to work with the RGB values of the image. The goal of this mask is to turn dark grays and blacks into 0 and all other values to 1. The reason for this mask is most plates have a dark contourn (mostly black, sometimes dark gray), which we can take advantage of to isolate the plate as a connected component (CC) to work with later. If this contour is neither dark nor well defined the plate will not be well isolated and no Final candidate found. (Later explained).

There is another masks later in the code if this one doesn't work to check for lighter plate contour.

Binarization of the mask to create components

Eventhough the masked image should already be a binary image for all values are either one or zero, when working with the CCs by using the function regionprops, there would be some errors not finding all the components correctly. Using imbinarize over the masked image helps fixing this issue.

Find plate candidates from the components

For any of this CCs to be considered a candidate to be the car plate, they have to pass a two parameter filter. The first parameter is shape. Assuming a plate has a rectangular shape with a bigger width than height, all CCs which have bigger height than width are eliminated from the candidates list.

The second parameter is height-width ratio. Modern car plates are standarized to be 52 cm wide and 11 cm high which makes a ratio of 52/11 = 4.7 roughly 5. Taking into account the tilted plates which have a lower ratio, for they have bigger height, we can consider 4 as the ratio value as a result of some trials. Using some tolerances as to ensure the fitting of all plates (with and without blue part, tilted or not tilted, etc.) the final ratio numbers are between:

Lower limit --> 3 times height = width

Upper limit --> 6 times height = width

Indentify plate from candidates

The approach used to identify the plate over all the final candidates has been to find blue pixels either inside the candidates or at their left. The reason for this being, this algorithm sometimes finds the plate without the blue country box, and other times with it. If the plate includes the blue box, the blue pixels are going to be inside the component and, if the plate does not include the blue box, the blue pixels are outside it.

As the blue box has the stars at the upper part, and the country letters at the bottotm, it has been decided to chech one pixel at the middle therefore the height value of this pixel is going to be the Bounding Box height's middle value: @bboxes/2.

As for the distance from the bBox border it has been arbitrarily decided the value would be +-10 pixels.

@bBoxes(Candidates, 1) - 10 or bBoxes(Candidates, 1) + 10.

@bBoxes(Candidates, 1) --> x bounging box starting value.

The chosen pixel will be found by combining adding or substracting 10 pixels to bBoxes(Candidates, 1) and using bBoxes(Candidates, 4) /2.

Blue identification:

After checking on what defines blue color it has been stated 3 parameters:

Red channel value is the smallest and way smaller than blue.

Green channel value is bigger than red and also smaller than blue.

Blue channel value is the biggest by far.

This three aspects are always true, but not always with the same ratios nor intensities due to the influence of shades and the type of blue. Therefore this three parameters are going to be the method to define whether exists a blue pixel near the candidate. The values defined below at the for loop to find the blue pixel have been defined acording to the darkest blue found at the plates of the database.

Second threshold if there are zero final candidates

If there is no final candidate it can be asumed that the plate analysed doesn't have a dark contourn and therefore it was not well isolated from the rest of the car. For this reason it has been decided to repeat the detection process applying another mask with a different treshold and also cheking for intensity values with the aim of turn silver and gray colors into 0 and isolate the plate.

The threshold is being established by trial and error.

Find final plate

When there is a Final candidate the next step is to crop the resulting image. A safety distance of 10 pixels has been added at the top and bottom of the image to make sure when cleaning the plate, no letter nor number are in contact with the borders. The reason being the cleaning method would remove it.

Plate cleaning

For the cleaning, a threshold to find dark colors has been defined. This will ensure that the black plate numbers are detected. The reason being the pipeline is defined to work with white and yellow plates but only with black colored plate numbers. Later all small areas and also all elements in contact with the picture borders are set to 0.

The next steps of cleaning the plate are:

- Delete the elements that are stuck around the image
- Morphological operation on the mask
- Complementing the plate

Character detection using OCR

Check if no characters are found

In case there is no value inside the defined variable, it can be asumed the initial threshold is not set properly for the current plate. The reason being ilumination. When processing all the training images it could be noticed that ilumination has a huge impact when tresholding the plate. Shades make all pixels darker and light lighter as can be easily understand. The problem with setting a high threshold to read the letters from the better iluminated plates, and therefore having higher

intensity blacks, is than some of the low intensity whites from the darker plates can pass throught the filter and then cause interference. The best approach found to deal with this is setting a lower threshold to process poorly iluminated images and later, if the plate number is not well processed due to the blacks having too high of an intensity, checking again with a higher threshold.

The new threshold has been set to 90 in a trial and error way.

Displays the license plate number

Results Analysis

We evaluated the performance of our ALPR system on a dataset of real-world images and achieved an accuracy of 75%. Our data set contains 20 images. To visualise teh results we used tables in Excel. The tables contain picture id, picture itself, the result we got and the label we attached to every picture. There are three labels: succes (when we can fully see the proper plate number), partially succes (when we see no all numbers, or we could identify the palte but could not extract the numbers) and no succes when the plate is not found.

Training Data

2			success
4		Insufficient number of outputs from right hand side of equal sign to satisfy assignment.	no success
10		LETERS 	partially success

Analysing the pictures with “no success” label we can conclude that, to properly identify the number camera has to be always in the front direction to the car. Moreover, to make it work with the highest efficienty, the lighting in the parking lot has to be bright, to avoid all of possible flaws. Otherwise, having 1) wrong car’s angle and 2) bad lighting we will be not able to extract the number.

Testing Data

How mentioned before, below the “success” label got the picture that shows car from the front, with the good enough lighting.The next examples, number 1 and number 2, show that our code doe snot handle well pictures with described issues.

Adjusted factors:

1		LETERS B 58 BPS	succes

1) Wrong angle:

2		Insufficient number of outputs from right hand side of equal sign to satisfy assignment.	no succes

2) Bad light:

4		Possible candidates 	no succes

Conclusion and Recommendation

Initially, we approached the project, having a lot of questions and things that were unclear to us, but with the guidance of our teacher we managed to successfully develop an Automatic License Plate Recognition system with MatLab that has a great success rate and is ready to be applied in a real life project. We created a list with all the tasks and sub-tasks that needs to be executed in order to successfully create a working prototype and assigned all team members accordingly.

Although, we decided to stick to the use-case of having an automated parking management system using ALPR, but based on the specific goals and requirements of the system there are many other options such as:

Improved traffic flow: The system can be used to identify and track vehicles as they pass through a particular location, such as a toll booth or a border crossing. This can help improve traffic flow by reducing delays and bottlenecks caused by manual license plate checks.

Enhanced security: An automatic license plate recognition system can be used to identify vehicles that are of interest to law enforcement or security agencies. For example, the system could be used to alert authorities to the presence of a vehicle that is connected to a criminal investigation or that has been flagged as suspicious.

Streamlined vehicle registration and licensing: ALPR can be used to automate the process of registering and licensing vehicles. The system could be used to scan and record the license plate information of vehicles as they pass through a particular location, and to update records in real-time.

References

M. A. Razzak, A. Al-Nima, and A. Y. Zaidan, "Automatic License Plate Recognition Using Deep Learning," Journal of Visual Communication and Image Representation, vol. 54, pp. 1-9, 2018.

M. H. Ang and R. B. Fisher, "A Review of License Plate Recognition Techniques," IEEE Transactions on Intelligent Transportation Systems, vol. 11, no. 2, pp. 456-466, 2010.

H. S. Ong, S. K. Poh, and W. K. Goh, "An Overview of Automatic License Plate Recognition (ALPR) Systems," International Journal of Advanced Science and Technology, vol. 97, pp. 1-10, 2018.

M. A. Razzak, A. Al-Nima, and A. Y. Zaidan, "Deep Learning for Automatic License Plate Recognition: A Review," IEEE Access, vol. 7, pp. 83,973-83,987, 2019.

H. S. Ong, S. K. Poh, and W. K. Goh, "Automatic License Plate Recognition: A Survey," IEEE Access, vol. 7, pp. 85,969-85,989, 2019.

Sharma, G. (2018). Performance Analysis of Vehicle Number Plate Recognition System Using Template Matching Techniques. *Journal of Information Technology &Amp; Software Engineering*, *08*(02). https://doi.org/10.4172/2165-7866.1000232

Shashirangana, J., Padmasiri, H., Meedeniya, D., & Perera, C. (2021). Automated License Plate Recognition: A Survey on Methods and Techniques. *IEEE Access*, *9*, 11203–11225. https://doi.org/10.1109/access.2020.3047929