Handling Technology Report

Enhancing License Plate Recognition Through Homography Transformation

Made by Aurore, Léo, Handling Technology 2023-2024

TABLE OF CONTENT

Abstract

Continuing from a prior project focused on license plate recognition, we're addressing a persistent challenge: accurately recognizing characters on plates viewed from non-frontal angles or orientations. We propose a method utilizing homography transformation to adjust the perspective of the license plate image, making it appear as if viewed from the front. By understanding and applying this transformation, our aim is to enhance character recognition accuracy. In this Handling Technology Report, we detail the development and implementation of our algorithm, which combines image processing techniques with the principles of homography transformation. We present an overview of the algorithm's functionality, including the detection of license plate corners, the application of homography transformation, and the extraction of text information using optical character recognition (OCR). Additionally, we discuss the modifications made to the original code to improve detection accuracy and evaluate the effectiveness of our approach through testing on real-world datasets. Through our research and experimentation, we aim to improve the quality of our license plate recognition algorithm without using machine learning.

Explanation on how to run the code on colab

Please review the following pdf to have information on how to upload the images to colab and how to run the full code. You will only need an internet connection and the picture folder to run the code. You can already see the results of the code without running it.

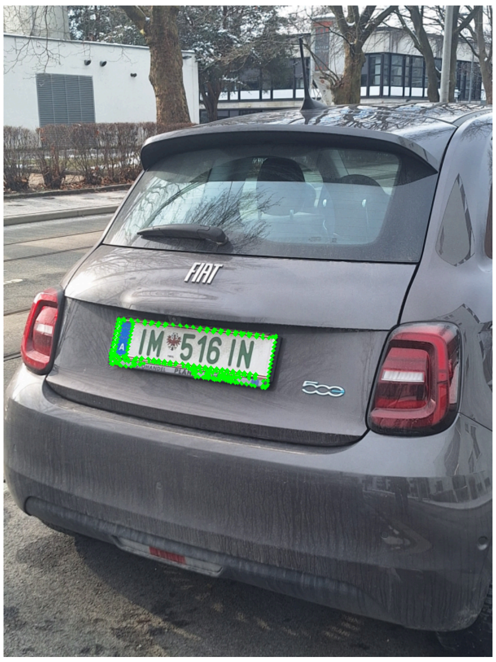
UsingGoogleColab.pdf

License plate recognition algorithm

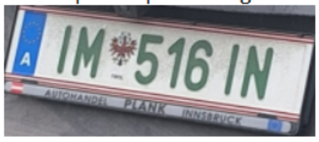
Quick overview of how it works

The algorithm developed for license plate detection and text recognition employs various image processing techniques without relying on machine learning. Utilizing Python, along with OpenCV and Pytesseract libraries, the process involves several key steps.

At first, the algorithm resizes the input image to enhance detection speed. Subsequently, it converts the image thanks to different filters like grayscale and bilateral filters to reduce noise. Contours are detected using OpenCV's findContours function, and the algorithm filters potential license plate contours based on size. Top 10 contours with the largest areas are retained for further processing. The algorithm then loops over these contours to identify potential license plates. Contours with four corners are selected and cropped for further analysis. A function calculates the percentage of white pixels in the cropped image to determine if it likely represents a license plate depending on an upper and lower threshold. If deemed a potential license plate, the cropped image undergoes character recognition using Pytesseract. The algorithm extracts the characters from the license plate and determines the region of origin based on the first two letters.



```
white pixel percentage: 74.24
```



```
TEXTE
You should add  initials to our austrian region repertory
```

Figure 1 - Results from the license plate recognition code

Testing accuracy revealed higher success rates for front-facing images compared to angled ones. Challenges such as image quality and angle significantly impacted algorithm performance, highlighting areas for further improvement.

Changes applied to the original code

In the initial version of the license plate detection code, we did not originally include the return of the four corners of the identified license plate. Instead, the output consisted solely of the cropped image around the license plate along with the text detected on it. This was the primary modification we made. Presently, the license plate detection algorithm has been updated to include the retrieval of the four corners of the detected license plate.

Another adjustment we implemented pertained to the decision of whether or not to print all the intermediate pictures generated during the detection algorithm. To accommodate this, we introduced a function parameter to manage this aspect.

Additionally, we observed that the algorithm still struggled with accurately determining whether the detected object was indeed a license plate. To address this, we introduced a maximum length constraint for the detected text. If the text exceeds a certain length, typically more than 10 characters, it is presumed not to be a license plate, thus enhancing the accuracy of detection.

The homography matrix

Quick explanation of how it works/What the homography matrix is

An homography matrix is a mathematical transformation that relates the geometric properties of points in one plane to the corresponding points in another plane. In computer vision and image processing, homography is commonly used for tasks such as image rectification and object recognition. In our case we use it to have a frontal view of the license plate to improve the character recognition.

The homography matrix is a 3x3 matrix denoted as H in our project. It represents a projective transformation between two planes (the real plane formed by the license plate on the picture and the frontal plane desired).

If you have a point in the source plane represented as P = [x, y, 1] (homogeneous coordinates), the transformed point in the destination plane P' = [x', y', 1] is obtained by multiplying it with the homography matrix : P' = H*P.

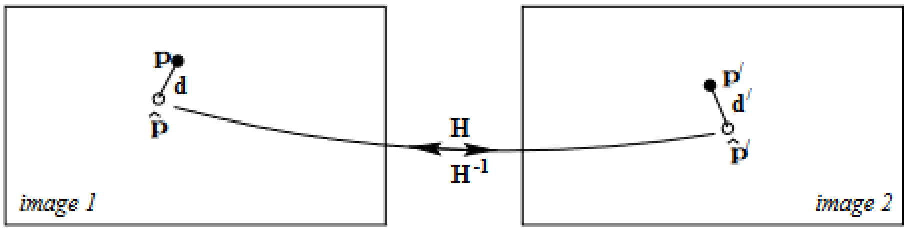


Figure 2 - Representation of the change induced by H

The elements of the homography matrix are determined by solving a system of linear equations using the known point correspondences. Typically, a minimum of four non-collinear points is needed to compute a unique homography matrix (corners of the license plate in our project).

Working with an already found homography matrix

To apprehend how the homography matrix worked on a picture, we first chose to experiment with applying a homography matrix using an existing OpenCV function to automatically determine the homography matrix and apply it to the license plate picture and see how it can modify the perspective of the image.

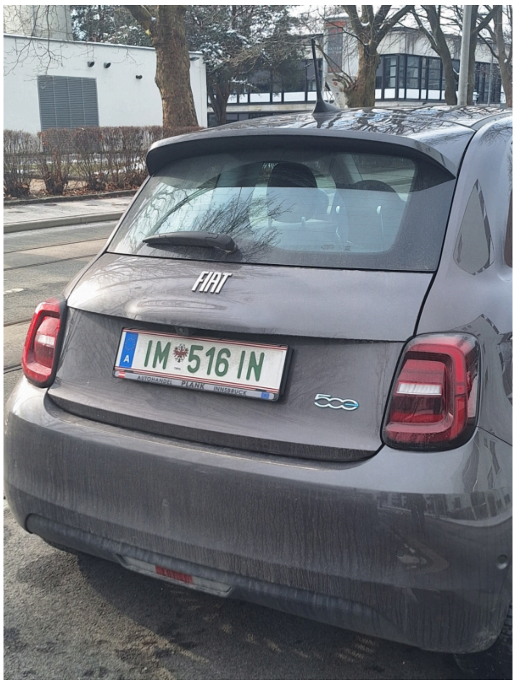


Figure 3 - The original image

We discovered that the change applied to the cropped license plate picture resulted in a black image, indicating potential issues with corner coordinates or objective contour definition. We hypothesized that the corner coordinates might be in the reference frame of the overall picture rather than the cropped license plate one, or that the objective corners were incorrectly defined. To address this, we explored alternative approaches.

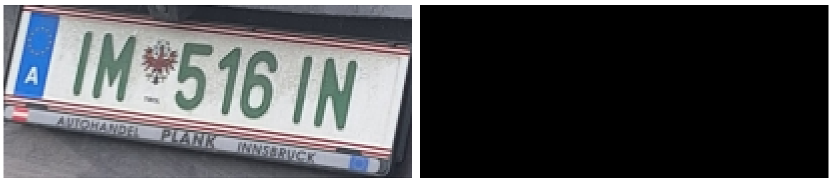


Figure 4 - Before and after picture of the cropped license plate image

By applying the change to the entire car picture instead of just the cropped license plate, we observed that the desired modification was achieved. However, the proportions were not accurate, prompting us to reconsider the objective contours since those determined the final change. We aimed to adjust the objective contours to ensure that the resulting image reflected the correct proportions for a license plate.

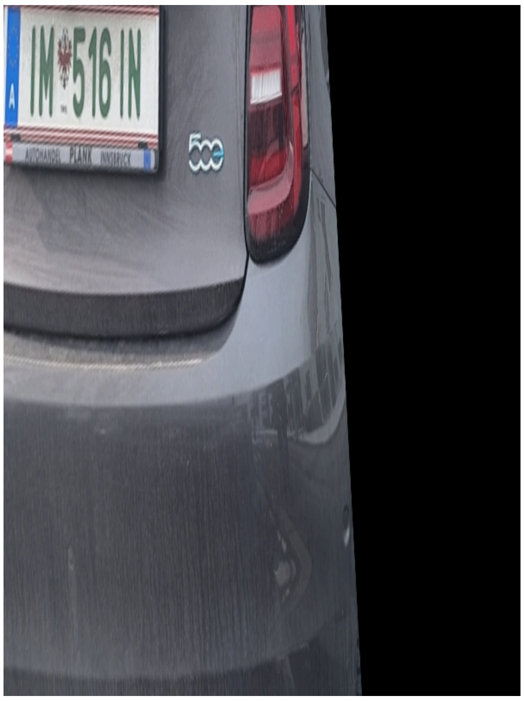
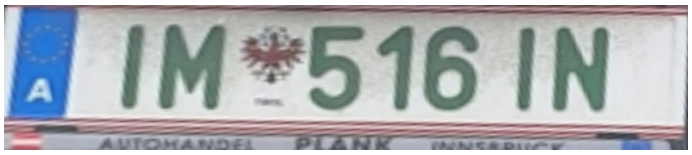
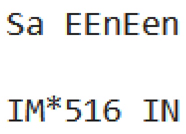


Figure 5 - Overall picture of the car after applying the homography matrix

Referencing standard license plate sizes, such as those in France and Austria, we refined the objective contours to better match the expected proportions. Although the proportions improved significantly, we recognized the need to crop the image specifically around the license plate for character recognition purposes. However, a potential limitation of this approach is its reliance on specific picture dimensions, which may necessitate further refinement if picture dimensions are altered and it can become a problem as the picture doesn't have the correct resolution for this resizing.





```
Sa EEnEen

IM*516 IN
```

Figure 6 - Overall picture after the wrapping from the homography and cropping

Figure 7 - Resulting text detection

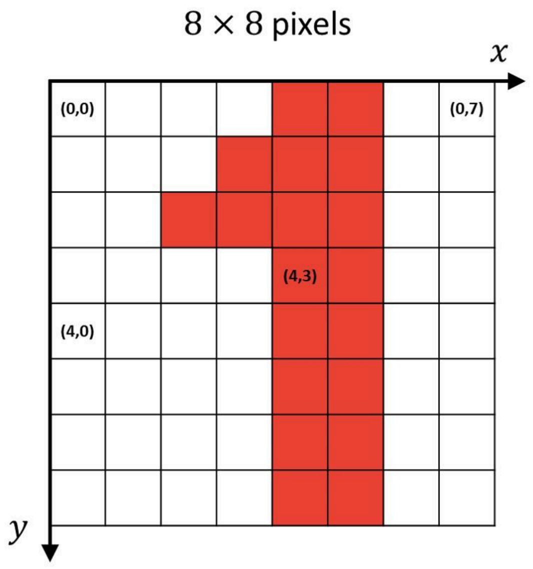
In summary, we successfully applied character recognition to the modified license plate image, even though the resulting text contains additional elements beyond the characters themselves. To use those operations later on for efficiency measurement, we developed a function that contains all the necessary steps, taking the picture name and corner coordinates as inputs.

Successful license plate detection rate with and without

To assess the impact of applying the homography matrix on the success rate of our license plate detection algorithm, we conducted a series of evaluations on our dataset.

Initially, without applying the homography matrix, we utilized our license plate detection algorithm on every picture in the dataset to determine the success rate. The results revealed a 0% successful reading of the license plate text and a 22% success rate in correctly detecting the position of the license plate.

Upon implementing the homography matrix to correct perspective issues, we encountered challenges with the wrapping, possibly due to variations in the order of the detected corners. To investigate this, we examined whether the order of the contours remained consistent across different pictures. Graphical representations indicated varying corner orders, making us standardize the order by determining the top-left, bottom-right, etc., points using norm1. The coordinate system of the points in OpenCv is not the usual one. The x axis coordinates are going from top to bottom and following the y axis going from left to right as you can see in the following figure. We chose that the point with the smaller norm1 would be the one closest to the origin so the top-left corner for the plate and with a similar logic we assigned the different corners.



After adjusting the corner orders, we re-evaluated the success rate of our algorithm. While the wrapping appeared to work in every case where the detected element was a license plate, the application of the character recognition algorithm had mixed results. Although the success rate of reading the license plate text improved from 0% to 3.5%, there were still issues where the characters could have been recognised but they were not, possibly because of the image quality issues post-resizing or non-optimal parameters in the character recognition process.

Finding the homography matrix ourselves

We then decided to find the homography matrix ourselves in order to see if we could get the same results as with the “findHomography” function used before.

The “find_homography_ourself_3D” function was created for this purpose. It takes a set of 4 corresponding points from the detected license plate contour and a predefined target contour (objectif_contour). Singular Value Decomposition (SVD) is utilized to derive the homography matrix H, enabling perspective correction for non-frontal license plate views.

A comparison is made between the self-made homography matrix, and the one obtained through OpenCV's cv2.findHomography function. The Frobenius norm of the matrix difference is calculated, providing a quantitative measure of the dissimilarity between the two matrices. The difference is on the order of 10e-8 which is very small and it thus ensures the accuracy of the homography matrix.

The “homography_ourself” function applies the custom homography matrix to correct the perspective of a given input image. The corrected image is then cropped to isolate the license plate region. OCR using Tesseract is employed to extract text information from the license plate. Finally we see that the new homography matrix also makes it possible to have a front view of the license plate and to recognize the characters, at least for the examples that were working well before.

Conclusion

In conclusion, while the application of the homography matrix showed some improvements in success rates, it's evident that more refinement is necessary to achieve optimal performance. This shows that the fundamental idea holds merit but requires fine-tuning on multiple fronts. Potential areas for enhancement include fine-tuning parameters within the character recognition algorithm to enhance accuracy, as well as addressing potential image quality concerns to lower any adverse effects on recognition performance.

Moreover, there is room for improvement in the way images are captured, especially if the algorithm is intended for use in scenarios like parking management systems. Adjusting the orientation and angle at which pictures are taken could significantly enhance the algorithm's effectiveness, particularly in capturing clear and detailed images of license plates. By capturing images closer to the license plate and with a reduced angle, we can potentially minimize distortions and improve recognition accuracy.

Looking ahead, the most promising area for advancement lies in the integration of machine learning techniques. Training the algorithm using machine learning algorithms could enable it to learn and adapt to diverse real-world scenarios, ultimately enhancing its robustness and performance across various conditions.

References

[1] Vision par ordinateur : Homographie et Calibration, Jean-Philippe Tardif et Sébastien Roy, 2006

[2] Digital Image Processing Projective Geometry, Kurt Niel, 2017

[3]https://www.opencv.org.cn/opencvdoc/2.3.2/html/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html#cv2.findHomography

[4]https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html?highlight=warpperspective#warpperspective