Automatic Number-Plate Recognition
Image Processing Group Project Summary
2022/23 Fall Semester

Barnabás Börcsök BARNABAS.BORCSOK@EDU.BME.HU
Gergő Haragos GERI.HARAGOS@GMAIL.COM
Zoltán Simon ZOLTAN.SIMON@EDU.BME.HU
Bence Sándor Szabó SZABO.BENCE.SANDOR@GMAIL.COM

Budapest University of Technology and Economics

1. Introduction

Proofread + Actualize introduction section

There are many use cases in traffic control, where the Licence Plate (LP) of vehicles must be read. This task can be automated by computers. A wall-mounted or even hand-held camera can take pictures of cars. The image then can be processed by various algorithms to detect the LP, segment the characters and finally recognise these characters. In recent years with the increasing amount of traffic, the need for well-performing Automatic License Plate Recognition (ALPR) systems has increased substantially.

We propose our own ALPR implementation. We write about the state-of-the-art literature of the field. Our solution will be based on some of these publications. Our goal when selecting from the many available methods was to select the methods with the most promising test results.

We provide a project outline describing our planned tasks. The development process can be split up into a few large portions. This is visualised on a Gantt chart, where you can review each subtask.

After successful implementation, we will test our work on predefined training data as well as further real-life data. We also list some evaluation considerations.

Our implementation at `https://github.com/bobarna/bme-image-processing`, with detailed, reproducable steps.

Add document outline here.

2. Previous Solutions

2.1 Automatic Number-Plate Recognition

In the past years many research projects have been focused on **alpr!** (**alpr!**). We have surveyed some of the most recent and most promising works in this field. A recent survey by Shashirangana et al. (2021) gives an overview of different methods and practices used in **alpr!**.

Texture-based methods use characters present on the LP as the basis for **alpr!**. Significant color difference between the board and its characters creates a high-frequency color transition. If the image is grayscale, there is an easy to distinguish change of colors between the characters and the background of the board. This creates a unique pixel intensity distribution in the region of the plate. The plate region should have a high edge density. This is used in edge-based systems. In ke Xu et al. (2005) the authors used scan-line technique for **alpr!**.

Introduced by Redmon et al. (2016) as a novel object detection method, **yolo!** (**yolo!**) serves as the basis for the **alpr!** introduced by Laroca et al. (2019). The naming of **yolo!** comes from the fact that

it performs the object detection for the full image in a single pass. The employed Neural Network (NN) divides the image into regions and predicts bounding boxes and probabilities for each region. Building on this method the authors achieved a license plate recognition rate of 96.9%. The method was tested on multiple different datasets with outstanding results. Besides the novel approach, the authors of this work also released a public dataset of $38,351$ manually labeled bounding boxes on $6,239$ images.

A benchmark for **alpr!** is introduced by Gonçalves et al. (2016). This benchmark is composed of a dataset helping the **lpcs!** (**lpcs!**) step. High success rate of this step is crucial for end-to-end success of **alpr!**. Besides the dataset, the authors also propose a new evaluation measure of the location of the bounding box within the ground-truth annotation. To further optimise the **lpcs!** step, they suggest a more straightforward approach to perform it efficiently.

2.2 Object Detection

Write object detection previous solutions section

2.3 Optical Character Recognition

Write OCR previous solutions section (Zoli's excellent notebook is a good starting point)

3. Method

Rewrite this section based on what we actually did.

Separate into subsections: Object Detection, OCR, etc.

Aligning with the discussion of previous works in section 2, we will use a method based on **yolo!**(Redmon et al. (2016)) for the task of object detection, in a similar vein as Laroca et al. (2019).

We propose a two-phase method for **alpr!**. First, we will train a NN to recognize license plates on an image. Then, we feed the recognized LP to a second NN, trained for the task of Optical Character Recognition (OCR) of LPs.

3.1 Implementation

We plan to implement both NN components with PyTorch (Paszke et al. (2019)). For cutting out the LP from the image, we will use the OpenCV library (Bradski (2000)).

For data, we will rely on existing benchmark datasets for the most part (e.g. Gonçalves et al. (2016)). We focus on Hungarian LPs, but will probably have to resort to the use of internation datasets.

4. Evaluation

Write this section, actualize it to what we actually did.

Write about Precision/Recall, etc.

We will consider different cases depending on the difficulty of the given image, which will be determined based on how well our solution performs under the given (visual) setting. Based on this, we will categorize these cases by perceived difficulty, and show examples.

These are some of the settings we expect to make ALPR more difficult:

Create these comparisons, plot them in a figure.

- Direct sunlight

- Partial occlusion of the license plate
- Miscellaneous weather effects such as rain or fog
- Small size of the licence plate (i.e. the object is further away)

We will examine which of these settings introduce the most perceived noise, thus decreasing the effectiveness of our solution.

Other than these, – as is the case with all learning-based methods, – there is an inherent uncertainty on the generalization capabilities of our solution. After training and testing our solution on a predefined dataset, the real trial will be generalizing to photos we never encountered, as these images will be out-of-distribution for sure. Thus, we plan to finish our project with evaluating our solution *in the wild*, measuring its robustness on real-life data.

Add image results.

References

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. URL `https://github.com/opencv/opencv`.

Gabriel Resende Gonçalves, Sirlene Pio Gomes da Silva, David Menotti, and William Robson Schwartz. A benchmark for license plate character segmentation. *CoRR*, abs/1607.02937, 2016. URL `http://arxiv.org/abs/1607.02937`.

Hong ke Xu, Fu hua Yu, Jia hua Jiao, and Huan sheng Song. A new approach of the vehicle license plate location. In *Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05)*, pages 1055–1057, 2005. doi: 10.1109/PDCAT.2005.24.

Rayson Laroca, Luiz A. Zanlorensi, Gabriel Resende Gonçalves, Eduardo Todt, William Robson Schwartz, and David Menotti. An efficient and layout-independent automatic license plate recognition system based on the YOLO detector. *CoRR*, abs/1909.01754, 2019. URL `http://arxiv.org/abs/1909.01754`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

Jithmi Shashirangana, Heshan Padmasiri, Dulani Meedeniya, and Charith Perera. Automated license plate recognition: A survey on methods and techniques. *IEEE Access*, 9:11203–11225, 2021. doi: 10.1109/ACCESS.2020.3047929.