A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector

Rayson Laroca*, Evair Severo*, Luiz A. Zanlorensi*, Luiz S. Oliveira*,
Gabriel Resende Gonçalves†, William Robson Schwartz† and David Menotti*
*Department of Informatics, Federal University of Paraná (UFPR), Curitiba, PR, Brazil
†Department of Computer Science, Federal University of Minas Gerais (UFMG), Belo Horizonte, MG, Brazil

Email: {rblsantos, ebsevero, lazjunior, lesoliveira, menotti}@inf.ufpr.br   {gabrielrg, william}@dcc.ufmg.br

Abstract—**Automatic License Plate Recognition (ALPR) has been a frequent topic of research due to many practical applications. However, many of the current solutions are still not robust in real-world situations, commonly depending on many constraints. This paper presents a robust and efficient ALPR system based on the state-of-the-art YOLO object detector. The Convolutional Neural Networks (CNNs) are trained and fine-tuned for each ALPR stage so that they are robust under different conditions (e.g., variations in camera, lighting and background). Especially for character segmentation and recognition, we design a two-stage approach employing simple data augmentation tricks such as inverted License Plates (LPs) and flipped characters. The resulting ALPR approach achieved impressive results in two datasets. First, in the SSIG dataset, composed of 2,000 frames from 101 vehicle videos, our system achieved a recognition rate of 93.53% and 47 frames per second (FPS), performing better than both Sighthound and OpenALPR commercial systems (89.80% and 93.03%, respectively) and considerably outperforming previous results (81.80%). Second, targeting a more realistic scenario, we introduce a larger public dataset[1], called UFPR-ALPR dataset, designed to ALPR. This dataset contains 150 videos and 4,500 frames captured when both camera and vehicles are moving and also contains different types of vehicles (i.e., cars, motorcycles, buses and trucks). In the proposed dataset, the trial versions of commercial systems achieved recognition rates below 70%. On the other hand, our system performed better, with a recognition rate of 78.33% and 35 FPS.**

I. Introduction

Automatic License Plate Recognition (ALPR) has been a frequent topic of research [1]–[3] due to many practical applications, such as automatic toll collection, traffic law enforcement, private spaces access control and road traffic monitoring.

ALPR systems typically have three stages: License Plate (LP) detection, character segmentation and character recognition. The earlier stages require higher accuracy or almost perfection, since failing to detect the LP would probably lead to a failure in the next stages either. Many approaches search first for the vehicle and then its LP in order to reduce processing time and eliminate false positives.

Although ALPR has been frequently addressed in the literature, many studies and solutions are still not robust enough on real-world scenarios. These solutions commonly depend

on certain constraints, such as specific cameras or viewing angles, simple backgrounds, good lighting conditions, search in a fixed region, and certain types of vehicles (they would not detect LPs from vehicles such as motorcycles, trucks or buses).

Many computer vision tasks have recently achieved a great increase in performance mainly due to the availability of large-scale annotated datasets (i.e., ImageNet [4]) and hardware (GPUs) capable of handling a large amount of data. In this scenario, Deep Learning (DL) techniques arise. However, despite the remarkable progress of DL approaches in ALPR [5]–[7], there is still a great demand for ALPR datasets with vehicles and LPs annotations. The amount of training data is determinant for the performance of DL techniques. Higher amounts of data allow the use of more robust network architectures with more parameters and layers. Hence, we propose a larger benchmark dataset, called UFPR-ALPR, focused on usual and different real-world scenarios.

To the best of our knowledge, the SSIG SegPlate Database (SSIG) [8] is the largest public dataset of Brazilian LPs. This dataset contains less than 800 training examples and has several constraints such as: it uses a static camera mounted always in the same position, all images have very similar and relatively simple backgrounds, there are no motorcycles and only a few cases in which the LPs are not well aligned.

When recording the UFPR-ALPR dataset, we sought to eliminate many of the constraints found in ALPR applications by using three different non-static cameras to capture 4,500 images from different types of vehicles (cars, motorcycles, buses, trucks, among others) with complex backgrounds and under different lighting conditions. The vehicles are in different positions and distances to the camera. Furthermore, in some cases, the vehicle is not fully visible on the image. To the best of our knowledge, there are no public datasets for ALPR with annotations of cars, motorcycles, LPs and characters. Therefore, we can point out two main challenges in our dataset. First, usually, car and motorcycle LPs have different aspect ratios, not allowing ALPR approaches to use this constraint to filter false positives. Also car and motorcycle LPs have different layouts and positions.

As great advances in object detection were achieved through YOLO-inspired models [9], [10], we decided to fine-tune it for ALPR. YOLOv2 [11] is a state-of-the-art real-time object

---

[1]The UFPR-ALPR dataset is publicly available to the research community at https://web.inf.ufpr.br/vri/databases/ufpr-alpr/ subject to privacy restrictions.

detector that uses a model with 19 convolutional layers and 5 max-pooling layers. On the other hand, Fast-YOLO [12] is a model focused on a speed/accuracy trade-off that uses fewer convolutional layers (9 instead of 19) and fewer filters in those layers. Therefore, Fast-YOLO is much faster but less accurate than YOLOv2.

In this work, we propose a new robust real-time ALPR system based on the YOLO object detection Convolutional Neural Networks (CNNs). Since we are processing video frames, we also employ temporal redundancy such that we process each frame independently and then combine the results to create a more robust prediction for each vehicle.

The proposed system outperforms previous works and two commercial systems in the SSIG dataset and also in the proposed UFPR-ALPR. The main contributions of this paper can be summarized as follows:

- A new real-time end-to-end ALPR system using the state-of-the-art YOLO object detection CNNs[2];
- A robust two-stage approach for character segmentation and recognition mainly due to simple data augmentation tricks for training data such as inverted LPs and flipped characters;
- A public dataset for ALPR with 4,500 fully annotated images (over 30,000 LP characters) focused on **usual and different** real-world scenarios, showing that our proposed ALPR system yields outstanding results in both scenarios;
- A comparative evaluation among the proposed approach, previous works in the literature and two commercial systems in the UFPR-ALPR dataset.

This paper is organized as follows. We briefly review related works in Section II. The UFPR-ALPR dataset is introduced in Section III. Section IV presents the proposed ALPR system using object detection CNNs. We report and discuss the results of our experiments in Section V. Conclusions and future works are given in Section VI.

II. RELATED WORK

In this section, we briefly review several recent works that use DL approaches in the context of ALPR. For relevant studies using conventional image processing techniques, please refer to [1], [2], [13]–[19]. More specifically, we first discuss works related to each ALPR stage, and then works that not fit into the other subsections. This section concludes with final remarks.

LP Detection: Many authors have addressed the LP detection stage with object detection CNNs. Montazzolli and Jung [20] used a single CNN arranged in a cascaded manner to detect both car frontal-views and its LPs, achieving high recall and precision rates. Hsu et al. [21] customized CNNs exclusively for LP detection and demonstrated that the modified versions perform better. Rafique et al. [22] applied Support Vector Machines (SVM) and Region-based CNN (RCNN) for LP

detection, noting that RCNNs are best suited for real-time systems.

Li and Chen [5] trained a CNN based on characters cropped from general text to perform a character-based LP detection, achieving higher recall and precision rates than previous approaches. Bulan et al. [3] first extracts a set of candidate LP regions using a weak Sparse Network of Winnows (SNoW) classifier and then filters them using a strong CNN, significantly improving the baseline method.

Character Segmentation: ALPR systems based on DL techniques usually address the character segmentation and recognition together. Montazzolli and Jung [20] proposed a CNN to segment and recognize the characters within a cropped LP. They correctly segmented more than 99% of the characters, outperforming the baseline by a large margin.

Bulan et al. [3] achieved very high accuracy in LP recognition jointly performing the character segmentation and recognition using Hidden Markov Models (HMMs) where the most likely LP was determined by applying the Viterbi algorithm.

Character Recognition: Menotti et al. [23] proposed the use of random CNNs to extract features for character recognition, achieving a significantly better performance than using image pixels or learning the filters weights with back-propagation. Li and Chen [5] proposed to perform the character recognition as a sequence labeling problem. A Recurrent Neural Network (RNN) with Connectionist Temporal Classification (CTC) was employed to label the sequential data, recognizing the whole LP without the character-level segmentation.

Although Svoboda et al. [24] have not performed the character recognition itself, they achieved high-quality LP deblurring reconstructions using a text deblurring CNN, which can be very useful in character recognition.

Miscellaneous: Masood et al. [7] presented an end-to-end ALPR system using a sequence of deep CNNs. As this is a commercial system, little information is given about the CNNs used. Li et al. [6] propose a unified CNN that can locate LPs and recognize them simultaneously in a single forward pass. In addition, the model size is highly decreased by sharing many of its convolutional features.

Final Remarks: Many papers only addressed part of the ALPR pipeline (e.g., LP detection) or performed their experiments on datasets that do not represent real-world scenarios, making it difficult to accurately evaluate the presented methods. In addition, most of the approaches are not capable of recognizing LPs in real time, making it impossible for them to be applied in some applications. In this sense, we employ the YOLO object detection CNNs in each stage to create a robust and efficient end-to-end ALPR system. In addition, we perform data augmentation for character recognition, since this stage is the bottleneck in some ALPR systems.

III. THE UFPR-ALPR DATASET

The dataset contains 4,500 images taken from inside a vehicle driving through regular traffic in an urban environment. These images were obtained from 150 videos with duration of 1 second and frame rate of 30 frames per second (FPS).

Fig. 1. Sample images of the UFPR-ALPR dataset. First three rows show the variety in backgrounds, lighting conditions, as well as vehicle/LP positions and types. The 4th row shows examples of vehicle and LP annotations. The LPs were blurred due to privacy constraints.

Thus, the dataset is divided into 150 vehicles, each with 30 images with only one visible LP in the foreground. It is noteworthy that no stabilization method was used. Fig. 1 shows the diversity of the dataset.

The images were acquired with three different cameras and are available in the Portable Network Graphics (PNG) format with size of $1,920 \times 1,080$ pixels. The cameras used were: *GoPro Hero4 Silver*, *Huawei P9 Lite* and *iPhone 7 Plus*. Images obtained with different cameras do not necessarily have the same quality, although they have the same resolution and frame rate. This is due to different camera specifications, such as autofocus, bit rate, focal length and optical image stabilization.

There are minor variations in the camera position due to repeated mountings of the camera and also to simulate a real condition, where the camera is not always placed in exactly the same position.

We collected 1,500 images with each camera, divided as follows: 900 of cars with gray LP, 300 of cars with red LP and 300 of motorcycles with gray LP. In Brazil, the LPs have size and color variations depending on the type of the vehicle and its category. Cars' LPs have a size of $40cm \times 13cm$, while motorcycles LPs have $20cm \times 17cm$. Private vehicles have gray LPs, while buses, taxis and other transportation vehicles have red LPs. There are other color variations for specific categories such as official or older cars. Fig. 2 shows some of the different types of LPs found in the dataset.
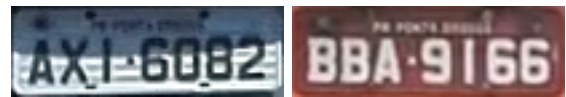
(a) Car LPs
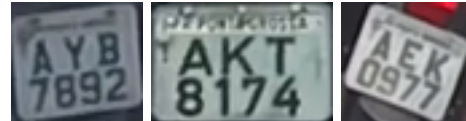
(b) Motorcycle LPs

Fig. 2. Examples of the different LP types found in the UFPR-ALPR dataset. In Brazil, cars' LPs have 3 letters and 4 digits in the same row and motorcycles' LPs have 3 letters in one row and 4 digits in another.

The dataset is split as follows: 40% for training, 40% for testing and 20% for validation, using the same protocol division proposed by Gonçalves et al. [8] in the SSIG dataset. The dataset distribution was made so that each split has the same number of images obtained with each camera, taking into account the type and position of the vehicle, the color and the characters of the vehicle's LP, the distance of the vehicle from the camera (based on the height of the LP in pixels) such that each split is as representative as possible.

The heat maps of the distribution of the vehicles and LPs for the image frame in both SSIG and UFPR-ALPR datasets are shown in Fig. 3. As can be seen, the vehicles and LPs are much better distributed in our dataset.

(a) Vehicles (SSIG) (b) LPs (SSIG)

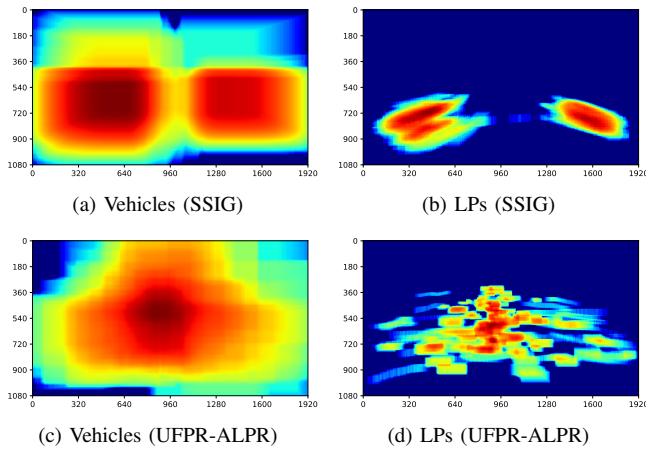(c) Vehicles (UFPR-ALPR) (d) LPs (UFPR-ALPR)

Fig. 3. Heat maps illustrating the distribution of vehicles and LPs in the SSIG and UFPR-ALPR datasets. The heat maps are log-normalized, meaning the distribution is even more concentrated than it appears.

In Brazil, each state uses particular starting letters for its LPs which results in a specific range. In Paraná (where the dataset was collected), LPs range from AAA-0001 to BEZ-9999. Therefore, the letters A and B have many more examples than the others, as shown in Fig. 4.
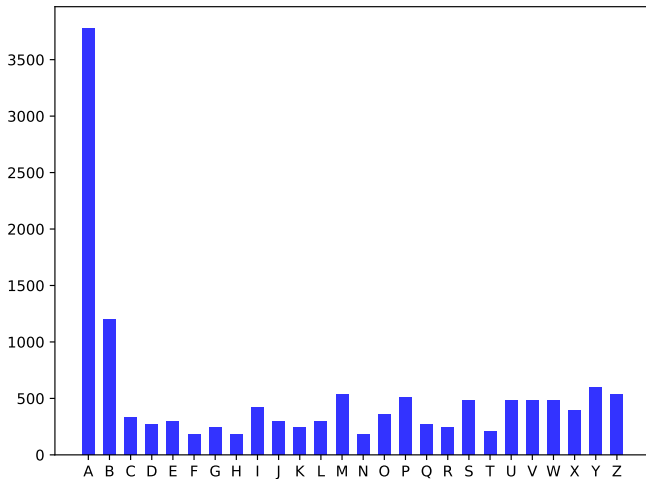

Fig. 4. Letters distribution in the UFPR-ALPR dataset.

Every image has the following annotations available in a text file: the camera in which the image was taken, the vehicle's position and information such as type (car or motorcycle), manufacturer, model and year; the identification and position of the LP, as well as the position of its characters. Fig. 1 shows the bounding boxes of different types of vehicle and LPs.

IV. PROPOSED ALPR APPROACH

This section describes the proposed approach and it is divided into four subsections, one for each of the ALPR stages (i.e., vehicle and LP detection, character segmentation and character recognition) and one for temporal redundancy. Fig. 5 illustrates the ALPR pipeline, explained throughout this section.

We use specific CNNs for each ALPR stage. Thus, we can tune the parameters separately in order to improve the performance of each task. The models used are Fast-YOLO, YOLOv2 and CR-NET [20], an architecture inspired by Fast-YOLO for character segmentation and recognition.

A. Vehicle and License Plate Detection

We train two CNNs in this stage: one for vehicle detection in the input image and other for LP detection in the detected vehicle. Recent works [20], [25] also performed the vehicle detection first.

We evaluated both Fast-YOLO and YOLOv2 models at this stage to be able to handle simpler (i.e., SSIG) and more realistic (i.e., UFPR-ALPR) data. For simpler scenarios, the Fast-YOLO should be able to detect the vehicles and their LPs correctly in much less time. However, for more realistic scenarios it might not be deep enough to perform these tasks.

In order to use both YOLO models[3], we need to change the number of filters in the last convolutional layer to match the number of classes. YOLO uses A anchor boxes to predict bounding boxes (we use $A = 5$) each with four coordinates (x, y, w, h), confidence and C class probabilities [11], so the number of filters is given by

$$filters = (C + 5) \times A. \tag{1}$$

In a dataset such as the SSIG dataset, we intend to detect only one class in both vehicle and LP detection (first the car and then its LP), so the number of filters in each task has been reduced to 30. On the other hand, the UFPR-ALPR dataset includes images from cars and motorcycles (two classes), so the number of filters in the vehicle detection task must be 35. In our tests, the results were better when using two classes (instead of just one class called 'vehicle'). The Fast-YOLO architecture used in both tasks is shown in Table I. The same changes were made in the YOLOv2 model (not shown due to lack of space).

TABLE I
FAST-YOLO NETWORK USED IN BOTH VEHICLE AND LP DETECTION. THERE ARE EITHER 30 OR 35 FILTERS IN THE LAST CONVOLUTIONAL LAYER TO DETECT ONE OR TWO CLASSES, RESPECTIVELY.

#	Layer	Filters	Size	Input	Output
0	conv	16	$3 \times 3/1$	$416 \times 416 \times 3$	$416 \times 416 \times 16$
1	max		$2 \times 2/2$	$416 \times 416 \times 16$	$208 \times 208 \times 16$
2	conv	32	$3 \times 3/1$	$208 \times 208 \times 16$	$208 \times 208 \times 32$
3	max		$2 \times 2/2$	$208 \times 208 \times 32$	$104 \times 104 \times 32$
4	conv	64	$3 \times 3/1$	$104 \times 104 \times 32$	$104 \times 104 \times 64$
5	max		$2 \times 2/2$	$104 \times 104 \times 64$	$52 \times 52 \times 64$
6	conv	128	$3 \times 3/1$	$52 \times 52 \times 64$	$52 \times 52 \times 128$
7	max		$2 \times 2/2$	$52 \times 52 \times 128$	$26 \times 26 \times 128$
8	conv	256	$3 \times 3/1$	$26 \times 26 \times 128$	$26 \times 26 \times 256$
9	max		$2 \times 2/2$	$26 \times 26 \times 256$	$13 \times 13 \times 256$
10	conv	512	$3 \times 3/1$	$13 \times 13 \times 256$	$13 \times 13 \times 512$
11	max		$2 \times 2/1$	$13 \times 13 \times 512$	$13 \times 13 \times 512$
12	conv	1024	$3 \times 3/1$	$13 \times 13 \times 512$	$13 \times 13 \times 1024$
13	conv	1024	$3 \times 3/1$	$13 \times 13 \times 1024$	$13 \times 13 \times 1024$
14	conv	30/35	$1 \times 1/1$	$13 \times 13 \times 1024$	$13 \times 13 \times 30/35$
15	detection				

[3]For training YOLOv2 and Fast-YOLO we used convolutional weights pre-trained on ImageNet [4], available at https://pjreddie.com/darknet/yolo/
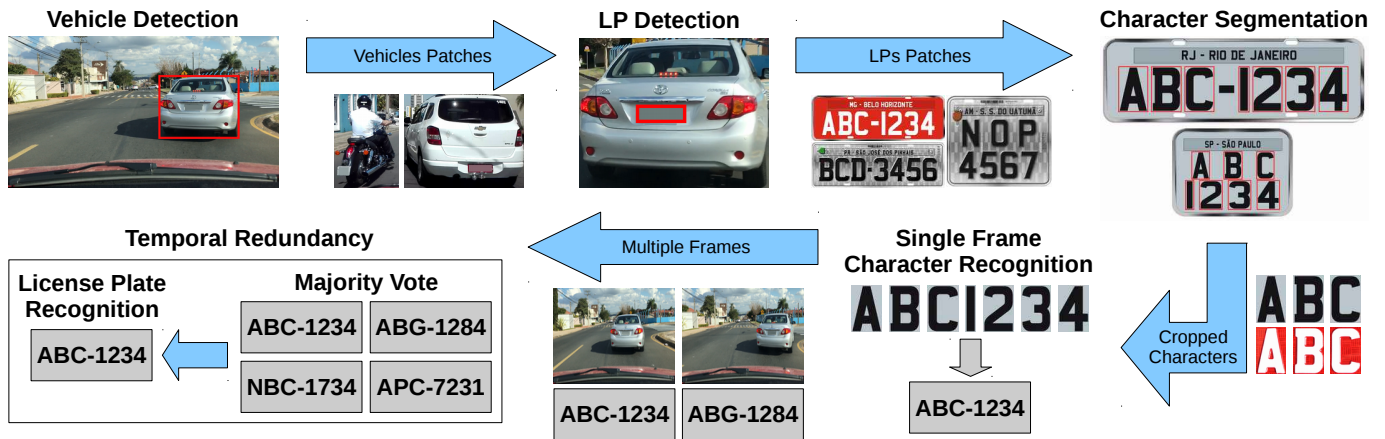
Fig. 5. A usual ALPR pipeline having temporal redundancy at the end.

While the entire frame and the vehicle coordinates are used as inputs to train the vehicle detection CNN, the vehicle patch (with a margin) and the coordinates of its LP are used to learn the LP detection network. The size of the margin is defined as follows. We evaluated, in the validation set, the required margin so that all LPs would be completely within the bounding boxes of the vehicles found by the vehicle detection CNN. This is done to avoid losing LPs in cases where the vehicle is not very well detected/segmented.

By default, YOLO only returns objects detected with a confidence of 0.25 or higher. In the validation set, we evaluated the best threshold in order to detect all vehicles having the lowest false positive rate. A negative recognition result is given in cases where no vehicle is found. For LP detection we use threshold equal 0, as there might be cases where the LP is detected with very low confidence (e.g., 0.1). We keep only the detection with the highest confidence in cases where more than one LP is detected, as each vehicle has only one LP.

B. Character Segmentation

Once the LP has been detected, we employ the CNN proposed by Montazzolli and Jung [20] (CR-NET) for character segmentation and recognition. However, instead of performing both stages at the same time through an architecture with 35 classes (0-9, A-Z, where the letter O is detected jointly with the digit 0), we chose to first use a network to segment the characters and then another two to recognize them. Knowing that all Brazilian LPs have the same format: three letters and four digits, we use 26 classes for letters and 10 classes for digits. As pointed out by Gonçalves et al. [25], this reduces the incorrect classification.

The character segmentation CNN (architecture described in Table II) is trained using the LP patch (with a margin) and the characters coordinates as inputs. As in the previous stage, this margin is defined based on the validation set to ensure that all characters are completely within its predicted LP.

The CNN input size (240×80) was chosen based on the LP's ratio of Brazilian cars (3×1), however, the motorcycles

TABLE II
CHARACTER SEGMENTATION CNN, PROPOSED IN [20]. WE CHANGED THE NUMBER OF FILTERS IN THE LAST CONVOLUTIONAL LAYER TO 30, AS WE WANT TO FIRST SEGMENT THE CHARACTER (ONE CLASS).

#	Layer	Filters	Size	Input	Output
1	conv	32	$3 \times 3/1$	$240 \times 80 \times 3$	$240 \times 80 \times 32$
2	max		$2 \times 2/2$	$240 \times 80 \times 32$	$120 \times 40 \times 32$
3	conv	64	$3 \times 3/1$	$120 \times 40 \times 32$	$120 \times 40 \times 64$
4	max		$2 \times 2/2$	$120 \times 40 \times 64$	$60 \times 20 \times 64$
5	conv	128	$3 \times 3/1$	$60 \times 20 \times 64$	$60 \times 20 \times 128$
6	conv	64	$1 \times 1/1$	$60 \times 20 \times 128$	$60 \times 20 \times 64$
7	conv	128	$3 \times 3/1$	$60 \times 20 \times 64$	$60 \times 20 \times 128$
8	max		$2 \times 2/2$	$60 \times 20 \times 128$	$30 \times 10 \times 128$
9	conv	256	$3 \times 3/1$	$30 \times 10 \times 128$	$30 \times 10 \times 256$
10	conv	128	$1 \times 1/1$	$30 \times 10 \times 256$	$30 \times 10 \times 128$
11	conv	256	$3 \times 3/1$	$30 \times 10 \times 128$	$30 \times 10 \times 256$
12	conv	512	$3 \times 3/1$	$30 \times 10 \times 256$	$30 \times 10 \times 512$
13	conv	256	$1 \times 1/1$	$30 \times 10 \times 512$	$30 \times 10 \times 256$
14	conv	512	$3 \times 3/1$	$30 \times 10 \times 256$	$30 \times 10 \times 512$
15	conv	30	$1 \times 1/1$	$30 \times 10 \times 512$	$30 \times 10 \times 30$
16	detection				

LPs are nearly square (1.17×1). That way, we enlarged horizontally all detected LPs (to 2.75×1) before performing the character segmentation.

We also create a negative image of each LP, thereby doubling the number of training samples. Since the color of the characters in the Brazilian LPs depends on the category of the vehicle (e.g., private or commercial), the negative images simulate characters from other categories.

In some cases, more than 7 characters might be detected. If there are no overlaps (Intersection over Union (IoU) ≥ 0.25), we discard the ones with the lowest confidence levels. Otherwise, we perform the union between the overlapping characters, turning them into a single character. As motorcycle LPs can be very tilted, we use a higher threshold (IoU ≥ 0.75) to consider the overlap between its characters.

C. Character Recognition

Since many characters might not be perfectly segmented, containing missing parts, and as each character is relatively

small, even one pixel difference between the ground truth and the prediction might impair the character's recognition. Therefore, we evaluate different padding values (1-3 pixels) in the segmented characters to achieve higher recognition rates. As Fig. 6 illustrates, the more padding pixels the more noise information is added (e.g., portions of other characters or the LP frame).

| 0p | 1p | 2p | 3p | 4p | 5p | 6p |

Fig. 6. Comparison of different values of padding.

As previously mentioned, we use two networks for character recognition. For training these networks, the characters and their labels are passed as input. For digit recognition, we removed the first four layers of the character segmentation CNN, since in our tests the results were similar, but with a lower computational cost. However, for letter recognition (more classes and fewer examples) we still use the entire architecture of the character segmentation CNN. The networks for digit and letter recognition have 75 and 155 filters in the last convolutional layer, respectively (see Eq. 1).

The use of two networks allows the tuning of network parameters (e.g., input/output size) for each task. The best network sizes found in our experiments are $42 \times 26 \rightarrow 21 \times 13$ and $270 \times 80 \rightarrow 33 \times 10$ for digits and letters, respectively.

Having knowledge of the specific LP country layout (e.g., the Brazilian layout), we know which characters are letters and which are digits by their position. We sort the segmented characters by their horizontal and vertical positions for cars and motorcycles, respectively. The first three characters correspond to the letters and the last four to the digits, even in cases where the LP is considerably tilted. It is worth noting that a country (e.g., USA) might have different LP layouts, so this approach would not be suitable in such cases.

In addition to performing the training with the characters available in the training set, we also perform data augmentation in two ways. First, we create negative images to simulate characters from other vehicle categories (as in the character segmentation stage) and then, we also check which characters can be flipped both horizontally and vertically to create new instances. Table III shows which characters can be flipped in each direction.

TABLE III
THE CHARACTERS THAT CAN BE FLIPPED IN EACH DIRECTION TO CREATE NEW INSTANCES. WE ALSO USE THE DIGITS 0 AND 1 AS TRAINING EXAMPLES FOR THE LETTERS O AND I, RESPECTIVELY.

Flip Direction	Characters
Vertical	0, 1, 3, 8, B, C, D, E, H, I, K, O, X
Horizontal	0, 1, 8, A, H, I, M, O, T, U, V, W, X, Y
Both	0, 1, 6(9), 8, 9(6), H, I, N, O, S, X, Z

As in the LP detection stage, we use confidence threshold = 0 and consider only the detection with the highest confidence. Hence, we ensure that a class is predicted for every segmented character.

D. Temporal Redundancy

After performing the LP recognition on single frames, we explore the temporal redundancy information through the union of all frames belonging to the same vehicle. Thus, the final recognition is composed of the most frequently predicted character at each LP position (majority vote).

Temporal information has already been explored previously in ALPR [25], [26]. In both studies, the use of majority voting has greatly increased recognition rates.

V. EXPERIMENTAL RESULTS

In this section, we conduct experiments to verify the effectiveness of the proposed ALPR system. All the experiments were performed on an NVIDIA Titan Xp GPU (3,840 CUDA cores and 12 GB of RAM) using the Darknet framework [27].

We consider as correct only the detections with IoU > 0.5. This value was chosen based on previous works [6], [18], [20]. In addition, the following parameters were used for training the networks: $80k$ iterations (max batches) and learning rate = $[10^{-3}, 10^{-4}, 10^{-5}]$ with steps at $25k$ and $35k$ iterations.

Experiments were conducted in two datasets: SSIG and UFPR-ALPR. We report the results obtained by the proposed system and compare with previous work and two commercial systems[4]: Sighthound [7] and OpenALPR[5] [28]. According to the authors, both are robust in the detection and recognition of Brazilian LPs.

It is important to emphasize that although the commercial systems were not tuned specifically for these datasets, they use much larger private datasets, which is a great advantage, especially in DL approaches.

In the OpenALPR system we choose which LP's style we want to detect (i.e., Brazilian) and we do not need to make any changes. On the other hand, Sighthound uses a single model for LPs from different countries. Therefore, we made some adjustments in its prediction so that it fits the Brazilian LPs format, such as swapping 0 by O and vice versa.

A. Evaluation on the SSIG Dataset

The SSIG dataset [8] is composed of 2,000 images of 101 vehicles with the following annotations: the position of the vehicle's LP, its identification (e.g., ABC-1234) and each character's position.

The high-resolution images (1,920 × 1,080 pixels) were acquired with a static digital camera and are available in the PNG format. A sample frame of the dataset is shown in Fig. 7.

The SSIG dataset uses the following evaluation protocol: 40% of the dataset for training, 20% for validation and 40% for testing. According to the authors, this protocol was adopted because many character segmentation approaches do not require model estimation and a larger test set allows the reported results to be more statistically significant.

<hr>

[4]OpenALPR and Sighthound systems have Cloud APIs available at https://www.openalpr.com/cloud-api.html and https://www.sighthound.com/products/cloud, respectively. The results presented here were obtained in January 2018.

[5]Although it has an open-source version, the commercial version uses different algorithms for OCR trained with larger datasets to improve accuracy.

Fig. 7. A sample frame of the SSIG dataset. It should be noted that there are vehicles in the background that do not have annotations. The LPs were blurred due to privacy constraints.

We report only the results obtained with the Fast-YOLO model in the vehicle and LP detection subsections, since it achieved impressive recall and precision rates in both tasks.

1) Vehicle Detection: Since the SSIG dataset does not have vehicle annotations, we manually label the vehicle's bounding box on each image of the dataset. Another possible approach would be to train a vehicle detector using the large-scale CompCars dataset [29], but that way many vehicles (including those in the background) would also be detected.

In order to perform vehicle detection, we first evaluated different confidence thresholds. We started with a confidence of 0.5, however, some vehicles were not detected. All 407 vehicles in the validation set were successfully detected when the threshold was reduced to 0.25. Based on that, we decided to use half of this value (i.e., 0.125) in the test set to increase the chance that all vehicles are detected. With this threshold, we achieved a recall of 100% and precision above 99% (only 7 false positives).

2) License Plate Detection: Every vehicle in the validation set was well segmented with its LP completely within the predicted bounding box. Therefore, we use the vehicle patches without any margin to train the LP detection network. As expected, all LPs were correctly detected in both validation and test sets (recall and precision = 100%).

3) Character Segmentation: A margin of 5% (of the bounding box size) is required so each detected LP contains all its characters fully. Therefore, we double this value (i.e., 10%) in the test set and in the training of the character segmentation CNN.

We evaluated, in the validation set, the following confidence thresholds: 0.5, 0.25 and 0.1, but the recall achieved was 99.89%, regardless. Therefore, we chose to use a lower threshold (i.e., 0.1) in the test set to miss as few characters as possible. That way, we attained 99.75% (5,614/5,628) recall.

4) Character Recognition: The padding values that yielded the best recognition rates in the validation set were 2 pixels for letters and 1 pixel for digits. In addition, data augmentation with flipped characters only improved letter recognition, hampering digit recognition. We believe that a greater padding and data augmentation improve letter recognition because each class have far fewer training examples, compared to digits.

We first analyzed the results without temporal redundancy information. The proposed system achieved recognition rate of 85.45%, recognizing all three letters and all four digits in 86.32% and 98.63% of the time, respectively.

The results are greatly improved when taking advantage of temporal redundancy information. The final recognition rate is 93.53%, since the digits are correctly recognized in all vehicles and the letters in 93.53% of them. This result is given based on the number of frames correctly recognized, thereby vehicles with more frames have greater weight in the final result.

The recognition rates accomplished by the proposed system were considerably better than those obtained in previous works (81.8% → 93.53%), as shown in Table IV. As expected, the commercial systems also achieved great recognition rates, but only the proposed system was able to recognize correctly at least 6 of the 7 characters in all LPs. This is particularly important since the LP's identification can be combined with the vehicle's manufacturer/model [30] or its appearance [25] to further enhance the recognition.

TABLE IV
RECOGNITION RATES OBTAINED BY THE PROPOSED ALPR SYSTEM, PREVIOUS WORKS AND COMMERCIAL SYSTEMS IN THE SSIG DATASET.

ALPR	≥ 6 characters	All correct (vehicles)
Montazzolli and Jung [20]	90.55%	63.18%
Sighthound [7]	89.05%	73.13%
Proposed	99.38%	85.45%
OpenALPR [28]	92.66%	87.44%
Gonçalves et al. [25] (with redundancy)	–	81.80% (32/40)
Sighthound (with redundancy)	99.13%	89.80% (35/40)
OpenALPR (with redundancy)	95.77%	93.03% (37/40)
Proposed (with redundancy)	**100.00%**	**93.53% (37/40)**

According to our experiments, the great improvement in our system lies on separating the letter and digits recognition on two networks, so each one is tuned specifically for its task. Moreover, data augmentation was essential for letter recognition, since some classes (e.g., C, V) have less than 20 training examples.

In Table V, we report the recall/accuracy rate achieved in each ALPR stage separately, as well as the time required for the proposed system to perform each stage. The reported time is the average time spent processing all inputs in each stage, assuming that the network weights are already loaded.

TABLE V
RESULTS OBTAINED AND THE COMPUTATIONAL TIME REQUIRED IN EACH ALPR STAGE IN THE SSIG DATASET. RECALL STANDS FOR DETECTION AND SEGMENTATION, AND ACCURACY STANDS FOR RECOGNITION.

ALPR Stage	Recall/Accuracy	Time (ms)	FPS
Vehicle Detection	100.00%	4.0746	245
License Plate Detection	100.00%	4.0654	246
Character Segmentation	99.75%	1.6555	604
Character Recognition	97.83%	1.6452 × 7	87
ALPR (all correct)	85.45%	21.3119	47
ALPR (with redundancy)	93.53%		

Since the same model is used for vehicle and LP detection, the time required for both stages is very similar. The same is true for character segmentation and recognition, but the latter is performed 7 times (one time for each character). The average processing time for each frame was 21.31 seconds, an average of 47 FPS.

Our system had no difficulty recognizing red LPs, even with less training examples. According to our experiments, this is due to the negative images used in the training of the character segmentation and recognition CNNs. Due to the agreement terms of the SSIG dataset, we can not show qualitative results. Only a few LPs (all from the training set) can be shown for illustrations of publications.

B. Evaluation on the UFPR-ALPR Dataset

1) Vehicle Detection: We first evaluated the Fast-YOLO model, but the recognition rates achieved were not satisfactory. After evaluations with different confidence thresholds, the best recall rate achieved was 97.33%. This was expected since this dataset has greater variability in vehicle types and positions.

We chose to use the YOLOv2 model for vehicle detection, despite its higher computational cost. We evaluated several confidence thresholds, being 0.125 the best one, as in the SSIG dataset. The recall and precision rates achieved were 100% and 99%, respectively. Fig. 8 shows a motorcycle and a car detected with the YOLOv2 model.

Fig. 8. Examples of vehicle detection obtained with the YOLOv2 model.

2) License Plate Detection: We note that in more challenging images (usually of motorcycles), the vehicle's LP is not entirely within its predicted bounding box, requiring a small margin (5% in the validation set) so that the entire LP is completely within the predicted vehicle's bounding box. Therefore, we use a 10% margin in the test set and in the training of the LP detection CNN.

The detection rates obtained by both YOLO models were very similar (less than half a percent difference). Thus, we use the Fast-YOLO model for LP detection. The recall rate attained was 98.33% (1,770/1,800). We were not able to detect the LP in just one vehicle (in its 30 frames), because a false positive was predicted with greater confidence than the actual LP, as shown in Fig. 9.

We could use the character segmentation CNN to perform a post-processing in cases where more than one LP is detected, for example: evaluate on each detected LP if there are 7 characters or consider only the LP where the characters' confidence is greater. However, since the actual LP can be detected with very low confidence levels (i.e., ≤ 0.1), many

Fig. 9. A sample frame from the UFPR-ALPR dataset where the actual LP was not predicted with the highest confidence. The predicted position and ground truth are outlined in red and green, respectively. The LP was blurred due to privacy constraints.

false negatives would have to be analyzed, increasing the overall computational cost of the system.

3) Character Segmentation: In the validation set, a margin of 10% is required so each detected LP contains all its characters fully. We decided not to double the margin in the test set, as 20% would add a considerable amount of noise and background to the LPs patches.

The recall obtained was 97.59% when disregarding the LPs not detected in the previous stage and 95.97% when considering the entire test set. We accomplished better results in the SSIG dataset, but it is worth noting that our dataset has different LPs types and many of them are tilted. Fig. 10 depicts some LPs from different categories properly segmented, even when the LP is tilted or in presence of shadows.

Fig. 10. LPs from different categories properly segmented.

4) Character Recognition: The best results were obtained with 1 pixel of padding and data augmentation, for both letters and digits. The proposed system achieved a recognition rate of 64.89% when processing frames individually and 78.33% (47/60 vehicles) with temporal redundancy.

Despite the great results obtained in the previous dataset, both commercial systems did not achieve satisfactory results in the UFPR-ALPR dataset. Analyzing the results we noticed that a substantial part of the errors was in motorcycles images, highlighting this constraint in both systems. This suggests that those systems are not so well trained for motorcycles. OpenALPR performed better than Sighthound, attaining a recognition rate of 70% when exploring temporal redundancy information. Table VI shows all results obtained in the UFPR-ALPR dataset.

We report the recall/accuracy rate achieved in each ALPR stage separately in Table VII, as well as the time required for the proposed system to perform each stage. The vehicle detection stage is more time consuming in this dataset, as we use a larger CNN architecture (i.e., YOLOv2).

TABLE VI
RECOGNITION RATES OBTAINED BY THE PROPOSED ALPR SYSTEM AND COMMERCIAL SYSTEMS IN THE UFPR-ALPR DATASET.

ALPR	$\geq$ 6 characters	All correct (vehicles)
Sighthound [7]	62.50%	47.39%
OpenALPR [28]	54.72%	50.94%
Proposed	87.33%	64.89%
Sighthound (with redundancy)	76.67%	56.67% (34/60)
OpenALPR (with redundancy)	73.33%	70.00% (42/60)
Proposed (with redundancy)	**88.33%**	**78.33% (47/60)**

TABLE VII
RESULTS OBTAINED AND THE COMPUTATIONAL TIME REQUIRED IN EACH STAGE IN THE UFPR-ALPR DATASET. RECALL STANDS FOR DETECTION AND SEGMENTATION, AND ACCURACY STANDS FOR RECOGNITION.

ALPR Stage	Recall/Accuracy	Time (ms)	FPS
Vehicle Detection	100.00%	11.1578	90
License Plate Detection	98.33%	3.9292	255
Character Segmentation	95.97%	1.6548	604
Character Recognition	90.37%	1.6513×7	87
ALPR (all correct)	64.89%	28.3011	35
ALPR (with redundancy)	78.33%		

It is worth noting that despite using a deeper CNN model in vehicle detection (i.e., YOLOv2), our system is still able to process images at 35 FPS (against 47 FPS using Fast-YOLO). This is sufficient for real-time usage, as commercial cameras generally record videos at 30 FPS.

Fig. 11 illustrates some of the recognition results obtained by the proposed system in the UFPR-ALPR dataset. It is noteworthy that our system can generalize well and correctly recognize LPs under different lighting conditions.

AUC-1056   BCL-9595   ATU-4025

ASD-9743   AUS-0936   AXB-5487

AYL-2104   AHB-1989   AWX-9307

ALJ-9348   AKT-8174   MCA-9954

ABN-8528   AZU-3476   AWE-4633

Fig. 11. Qualitative results obtained by the proposed ALPR system in the UFPR-ALPR dataset. The first two rows show examples of correctly detected and incorrectly recognized LPs, while the following rows show samples of LPs (from different categories) successfully recognized.

VI. CONCLUSIONS

In this paper, we presented a robust real-time end-to-end ALPR system using the state-of-the-art YOLO object detection CNNs. We trained a network for each ALPR stage, except for the character recognition where letters and digits are recognized separately (with two distinct CNNs).

We also introduced a public dataset for ALPR that includes 4,500 fully annotated images (with over 30,000 LP characters) from 150 vehicles in real-world scenarios where both the vehicle and the camera (inside another vehicle) are moving. Compared to the largest Brazilian dataset (SSIG) for this task, our dataset has more than twice the images and contains a larger variety in different aspects.

At present, the bottleneck of ALPR systems is the character segmentation and recognition stages. In this sense, we performed several approaches to increase recognition rates in both stages, such as data augmentation to simulate LPs from other vehicle's categories and to increase characters with few instances in the training set. Although simple, these strategies were essential to accomplish outstanding results.

Our system was able to achieve a full recognition rate of 93.53% (85.45% without temporal redundancy) in the SSIG dataset, considerably outperforming previous results (81.8% with temporal redundancy [25] and 63.18% without [20]) and presenting a performance slightly better than commercial systems (93.03%). In addition, the proposed system was the only to correctly recognize at least 6 characters in all LPs.

We also evaluated our proposed ALPR system and two commercial systems as baselines on the new dataset. The results demonstrated that the UFPR-ALPR dataset is very challenging since both commercial systems reached recognition rates below 70%. Our system performed better, with a recognition rate of 78.33%. However, this result is still not satisfactory for some real-world ALPR applications.

As future work, we intend to explore new CNN architectures to further optimize (in terms of speed) the vehicle and LP detection stages. We also intend to correct the alignment of inclined LPs and characters in order to improve the character segmentation and recognition. Additionally, we plan to explore the vehicle's manufacturer and model in the ALPR pipeline, as our new dataset provides such information. Although our system was conceived and evaluated on two country-specific datasets from Brazil, we believe that the proposed ALPR system is robust to locate vehicles, LPs and alphanumeric characters from any other country. In this direction, aiming for a fully robust system we just need to design a character recognition module that is independent of the LP layout.

REFERENCES

[1] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic license plate recognition (ALPR): A state-of-the-art review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 311–325, Feb 2013.

[2] C. Gou, K. Wang, Y. Yao, and Z. Li, "Vehicle license plate recognition based on extremal regions and restricted boltzmann machines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1096–1107, April 2016.

[3] O. Bulan, V. Kozitsky, P. Ramesh, and M. Shreve, "Segmentation- and annotation-free license plate recognition with deep localization and failure identification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2351–2363, Sept 2017.

[4] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.

[5] H. Li and C. Shen, "Reading car license plates using deep convolutional neural networks and LSTMs," *CoRR*, vol. abs/1601.05610, 2016. [Online]. Available: http://arxiv.org/abs/1601.05610

[6] H. Li, P. Wang, and C. Shen, "Towards end-to-end car license plates detection and recognition with deep neural networks," *CoRR*, vol. abs/1709.08828, 2017. [Online]. Available: http://arxiv.org/abs/1709.08828

[7] S. Z. Masood, G. Shu, A. Dehghan, and E. G. Ortiz, "License plate detection and recognition using deeply learned convolutional neural networks," *CoRR*, vol. abs/1703.07330, 2017. [Online]. Available: http://arxiv.org/abs/1703.07330

[8] G. R. Gonçalves, S. P. G. da Silva, D. Menotti, and W. R. Schwartz, "Benchmark for license plate character segmentation," *Journal of Electronic Imaging*, vol. 25, no. 5, pp. 053 034–053 034, 2016.

[9] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–4.

[10] B. Wu, F. Iandola, P. H. Jin, and K. Keutzer, "SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 446–454.

[11] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 6517–6525.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 779–788.

[13] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License plate recognition from still images and video sequences: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 377–391, Sept 2008.

[14] G. S. Hsu, J. C. Chen, and Y. Z. Chung, "Application-oriented license plate recognition," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 2, pp. 552–561, Feb 2013.

[15] A. H. Ashtari, M. J. Nordin, and M. Fathy, "An iranian license plate recognition system based on color features," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1690–1705, Aug 2014.

[16] M. S. Sarfraz, A. Shahzad, M. A. Elahi, M. Fraz, I. Zafar, and E. A. Edirisinghe, "Real-time automatic license plate recognition for CCTV forensic applications," *Journal of Real-Time Image Processing*, vol. 8, no. 3, pp. 285–295, Sep 2013.

[17] R. Panahi and I. Gholampour, "Accurate detection and recognition of dirty vehicle plate numbers for high-speed applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 4, pp. 767–779, April 2017.

[18] Y. Yuan, W. Zou, Y. Zhao, X. Wang, X. Hu, and N. Komodakis, "A robust and efficient approach to license plate detection," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1102–1114, March 2017.

[19] S. Azam and M. M. Islam, "Automatic license plate detection in hazardous condition," *Journal of Visual Communication and Image Representation*, vol. 36, pp. 172 – 186, 2016.

[20] S. Montazzolli and C. R. Jung, "Real-time brazilian license plate detection and recognition using deep convolutional neural networks," in *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images*, Oct 2017, pp. 55–62.

[21] G. S. Hsu, A. Ambikapathi, S. L. Chung, and C. P. Su, "Robust license plate detection in the wild," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Aug 2017, pp. 1–6.

[22] M. A. Rafique, W. Pedrycz, and M. Jeon, "Vehicle license plate detection using region-based convolutional neural networks," *Soft Computing*, Jun 2017.

[23] D. Menotti, G. Chiachia, A. X. Falcão, and V. J. O. Neto, "Vehicle license plate recognition with random convolutional networks," in *2014 27th SIBGRAPI Conference on Graphics, Patterns and Images*, Aug 2014, pp. 298–303.

[24] P. Svoboda, M. Hradiš, L. Maršík, and P. Zemčík, "CNN for license plate motion deblurring," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 3832–3836.

[25] G. R. Gonçalves, D. Menotti, and W. R. Schwartz, "License plate recognition based on temporal redundancy," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2577–2582.

[26] M. Donoser, C. Arth, and H. Bischof, "Detecting, tracking and recognizing license plates," in *Asian Conference on Computer Vision (ACCV)*, 2007, pp. 447–456.

[27] J. Redmon, "Darknet: Open source neural networks in C," http://pjreddie.com/darknet/, 2013–2018.

[28] OpenALPR Cloud API, http://www.openalpr.com/cloud-api.html.

[29] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3973–3981.

[30] L. Dlagnekov and S. J. Belongie, *Recognizing cars*. Department of Computer Science and Engineering, University of California, San Diego, 2005.