Received February 1, 2020, accepted February 14, 2020, date of publication February 19, 2020, date of current version February 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2974973

Multinational License Plate Recognition Using Generalized Character Sequence Detection

CHRIS HENRY, SUNG YOON AHN, AND SANG-WOONG LEE
Pattern Recognition and Machine Learning Laboratory, Gachon University, Seongnam 13557, South Korea

Corresponding author: Sang-Woong Lee (slee@gachon.ac.kr)

This work was supported by the Gyeonggi-do Regional Research Center (GRRC) Program of Gyeonggi Province (Analysis of Behavior Based on Senior Life Log) under Grant GRRC-2017-B01.

ABSTRACT Automatic license plate recognition (ALPR) is generally considered a solved problem in the computer vision community. However, most of the current works on ALPR are designed to work on license plates (LP) from specific countries and use country-specific information which limits their practical applicability. Such ALPR systems require changes in the algorithm to work on other countries' LPs. Previous works on multinational LP recognition are tested on datasets from various countries that share the same LP layout. To address this issue, this study presents a deep ALPR system designed to be applicable to multinational LPs. The proposed approach consists of three main steps – LP detection, unified character recognition, and multinational LP layout detection. The system is mainly based on the you only look once (YOLO) networks. Particularly, tiny YOLOv3 was used for the first step whereas the second step uses YOLOv3-SPP – a version of YOLOv3 that consists of the spatial pyramid pooling (SPP) block. The localized LP is fed into YOLOv3-SPP for character recognition. The character recognition network returns the bounding boxes of the predicted characters and does not provide information about the sequence of the LP number. A LP number with an incorrect sequence is considered wrong. Thus, to extract the correct sequence, we propose a layout detection algorithm that can extract the correct sequence of LP numbers from multinational LPs. We collected our own Korean car plate (KarPlate) dataset and made it publicly available. The proposed system was evaluated on LP datasets from five countries which include South Korea, Taiwan, Greece, USA, and Croatia. In addition, a small dataset containing LPs from 17 countries was collected to evaluate the effectiveness of the multinational LP layout detection algorithm. The proposed ALPR system consumes about 42 ms per image on average for extracting LP number. Experimental results demonstrate the effectiveness of our ALPR system.

INDEX TERMS License plate detection, license plate recognition, multinational license plate recognition.

I. INTRODUCTION

Automatic license plate recognition (ALPR) has huge applicability in various applications such as stolen vehicle identification, parking lot management, electronic toll collection, traffic flow monitoring, etc. This topic has been extensively researched by researchers worldwide to improve performance of ALPR in real-world scenarios. Current ALPR algorithms achieve exemplary performance in controlled environments; however, performance is decreased when dealing with complex scenes. The current challenges in ALPR includes multinational ALPR, dealing with uncontrolled conditions such as uneven illumination, weather (snow, fog, rain, etc.), image distortion, image blurring, occlusions, etc. Multinational ALPR is a challenging issue due to the differences in license plate (LP) layouts among different countries and the non-availability of public multinational LP datasets. Different LP layouts and lack of publicly available multi-country datasets are responsible for the meager amount of research work conducted on the problem of multinational ALPR. Few research works [6], [15] have proposed multinational ALPR systems that claim to work on LPs from different countries. However, these methods were validated on datasets from various countries that shared a common LP layout. Based on our analyses, most of the LPs worldwide can be broadly classified into single line or double line LP. The datasets used in previous works [6], [15] contain only single line LPs and may require additional steps to recognize double line LPs.

A typical ALPR pipeline commonly consists of the following three steps: license plate detection, character segmentation, and character recognition. License Plate detection is

The associate editor coordinating the review of this manuscript and approving it for publication was Massimo Cafaro.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

responsible for finding the location of the LP in a given image. Character segmentation is responsible for segmenting individual characters from the detected LP whereas the role of character recognition is to classify each of the segmented characters. The first two steps are crucial for correct ALPR since it directly affects the character recognition stage. Failure to localize the LP in the first stage leads to failure in the subsequent stages. In order to overcome this issue, some literature merges the character segmentation and recognition steps as object recognition step. Few recent publications proposed end-to-end deep learning structures to completely remove inter-dependency among the three stages. However, as mentioned in the preceding paragraph, these methods are either tailored to work on specific country's LPs or are tested on multi-country datasets that share a common single line LP layout.

This paper presents a highly accurate deep ALPR system that is applicable to license plates belonging to multiple countries. In this study, we propose a three-stage deep multinational ALPR approach that combines deep learning with an image processing-based multinational LP layout detection algorithm. LP detection is the first stage of the proposed ALPR system which is responsible for detecting the LP region in an image. This stage uses tiny YOLOv3 network architecture [22] for detecting the LP region and is referred to as the "attention network" since it provides the LP region to the next stage of our ALPR system. Tiny YOLOv3 was selected since LP detection is a relatively simpler object recognition task and does not require an extremely deep network. Unified character recognition is the second stage of our ALPR pipeline and uses YOLOv3-SPP which we call as the "recognition network". YOLOv3-SPP is a modified version of YOLOv3 [22] that includes a spatial pyramid pooling (SPP) [23] block. YOLOv3-SPP was chosen due to its ability to deal with multiscale and small objects. LP region from the attention network is fed as input to the recognition network. The recognition network recognizes all the characters printed on the LP. The recognition network provides no information about the order of the recognized characters. For extracting an ordered string from the recognized characters, we propose a multinational LP layout detection algorithm. The proposed layout detection algorithm is based on image processing techniques and can extract the correct sequence of the LP number from multinational LPs. It does so by effectively classifying among single line and double line LPs. The proposed ALPR system was tested on Korean (KarPlate dataset), Taiwanese (AOLP dataset [26]), American (Caltech Cars (Rear) 1999 [28]), Greek (Medialab LPR Database [29]), and Croatian (University of Zagreb [30]) license plate datasets. Our own KarPlate dataset was generated by using our semi-automatic dataset generation strategy. Our ALPR system outperformed previous works in terms of performance and speed which is evident in Section 5. In order to test our layout detection algorithm, we collected a small dataset consisting of LPs from 17 different countries and tested it on the layout detection algorithm. The results show the effectiveness of our layout detection algorithm and validates the applicability of our ALPR system to multinational LPs.

The main contributions of our study are summarized below:

1) We present a deep end-to-end ALPR system which is applicable to multinational LPs without the need of any additional steps. Given an image containing a LP, the proposed system can extract the LP number with the correct sequence. The proposed ALPR system does not require any specific pre-processing to be used on different datasets. Previous studies could also be finetuned on other countries' datasets, however, modifications in the algorithm would be required to work on different LP layouts.
2) We propose a simple and effective multinational LP layout detection algorithm that can classify among various LP layouts used in most of the LPs used worldwide. Given the correct bounding boxes, the layout detection algorithm can extract the correct sequence of LP number from various LP layouts belonging to different countries.
3) To the best of our knowledge, we present the first publicly available[1] Korean car plate dataset (KarPlate dataset) containing more than 4,000 full HD images of Korean cars.
4) Based on the experimental results on datasets from five different countries, the proposed approach performs better than previous works in terms of performance even though many previous works utilize artificial data to prevent overfitting and to increase the dataset size. We did not use any artificial data in our system.

The remainder of the paper is organized as follows. Section II reviews related studies along with their limitations. The details about our semi-automatic data annotation strategy is discussed in Section III. The working of our algorithm is explained in Section IV while experimental evaluation is presented in Section V. Conclusion is covered in the last section of this paper.

II. RELATED WORKS

ALPR approaches can be broadly divided into two main categories – traditional image processing methods and deep learning methods. This section reviews the recent works in the latter category since our approach also falls into this category. Specifically, we will review the relevant literature in license plate detection (LPD) and license plate recognition (LPR) subcategories. In addition, we also review commercially available ALPR software and the work done on multinational LP detection/recognition. Lastly, we discuss the limitations of previous works.

A. LICENSE PLATE DETECTION

A cascade framework consisting of two convolutional neural networks was used to detect the LP region in [1]. The first

[1] Available at "`http://pr.gachon.ac.kr/ALPR.html`".

convolutional neural network (CNN) in the cascade framework was responsible for extracting regions containing text from the image while the second CNN classified text regions as LP or as general text. Each image was divided into subregions which were fed independently into a CNN in [4]. The CNN outputs a score which indicates how likely a specific sub-region is to contain a LP. A cascaded approach like the one in R-CNN [27] was used in [5] for LP localization. A weak SNoW classifier generates candidate LP regions which were fed into a strong CNN classifier (AlexNet [16]) to be scrutinized. Images that failed to pass a confidence test were fed into another CNN (AlexNet [16]) which identified the reason of failure. Failure identification can help in identification of probable problem which helps in troubleshooting the ALPR system at the earliest convenience. A simple CNN was used in [5], [10] for LPD. Region Proposal Network and Box Regression layer was used in [5] and [10], respectively, to detect the LP location.

Many real-time LPD approaches [3], [7], [8], [11], [13] used YOLO networks or its modified versions due to its fast inference speed. Most deep neural networks struggle at detecting small objects. To tackle this issue, the study in [7] trained FAST-YOLO [18] to detect the frontal view of a car. The detected frontal view was cropped and fed into the same network to detect the LP. The LP appears bigger in the cropped frontal view image which makes it easier for the network to detect the LP. The study in [3] modified YOLO [18] and YOLO9000 [19] for application in LPD and achieved better accuracy than the original YOLO networks. A multidirectional LPD method [8] used two networks to detect rotated LPs. The first network, referred to as the attention network, detected the LP region while the second network MD-YOLO (modified version of YOLO [18]) detected the rotated bounding box of the LP. Two YOLO networks, one for vehicle detection and the other for LPD, were used in [13]. A CNN called WPOD-NET that regresses coefficients of an affine transformation for detecting and unwarping distorted LPs was proposed in [11]. The input to WPOD-NET is the image of a vehicle which was detected by YOLO [18].

B. LICENSE PLATE RECOGNITION

Many studies [1], [5], [6] have tried to unify the subtasks (character segmentation and character recognition) of LPR. The studies in [1], [6] considered LPR as a sequence labeling problem making the character segmentation step unnecessary. The study in [1] proposed the use of a recurrent neural network (RNN) with long short-term memory (LSTM) and Connectionist Temporal Classification (CTC) [20] for LPR. On the other hand, the study in [6] used Bidirectional RNNs (BRNNs) with CTC loss [20] for LPR. The study in [5] employed a sweeping OCR technique that swept an OCR classifier across the LP image and localized the characters by utilizing a probabilistic inference method based on hidden Markov models (HMMs). Viterbi decoding was used for determining the most likely code sequence using a language model. A slightly modified version of YOLO called CR-NET along with a heuristic method was used in [7] for Brazilian LPR. A few other studies [9], [11] also used CR-NET for LPR. A low-computation CNN inspired by SqueezeNet Fire Blocks [21] and Inception Blocks was used for real-time LPR in [10]. The work in [12] combined feature maps after ROI pooling and fed them into subsequent classifiers for LPR. A different approach which segmented characters from a LP using semantic segmentation followed by a counting refinement stage was adopted in [13]. A modified version of DeepLabv2 ResNet-101 model was used for semantic segmentation. The counting refinement stage extracted character regions and fed them into AlexNet [16] for character counting. A sliding-window single class detector via tiny YOLO classifiers was used in [17] for LPR. 36 tiny YOLO models were used to recognize characters in the AOLP dataset [26].

C. COMMERCIAL ALPR SOFTWARE

Commercially available ALPR software include Sighthound [2] and OpenALPR [14]. Sighthound outperformed OpenALPR and used a sequence of deep CNNs, however, exact details about the CNN architecture are unavailable since it is a commercial product. It is worth mentioning that [2], [14] were trained on their own large privately collected dataset for various countries.

D. MULTINATIONAL LICENSE PLATE DETECTION/RECOGNITION

To the best of our knowledge, there has been considerably meager research conducted on the topic of multinational license plate detection/recognition. The study in [15] proposed a system for multinational license plate detection in images with complex backgrounds. First, the rear vehicle lights were extracted by converting the image to YUV color space. Once the rear lights were detected, the LP area was detected by using a histogram-based approach on the edge energy map. The study by [6], although not termed as multinational, tested their approach on LP datasets from various countries to validate the generalizing capability of their method.

E. LIMITATIONS

Most related works have one of the following limitations:

1) Most previous works on LP detection/recognition are tailored to work on license plates from specific countries. Generally, such approaches use country-specific information to constrain the issue of LP detection/recognition.
2) The previous works [6], [15] dealing with multinational license plates were tested on datasets from various countries, however, it must be noted that the datasets used for testing contained the same single-line license plate layout. In short, the datasets used lack diversity in LP layouts.
3) Certain previous works [1], [17] that test their approach on AOLP dataset require additional pre-processing

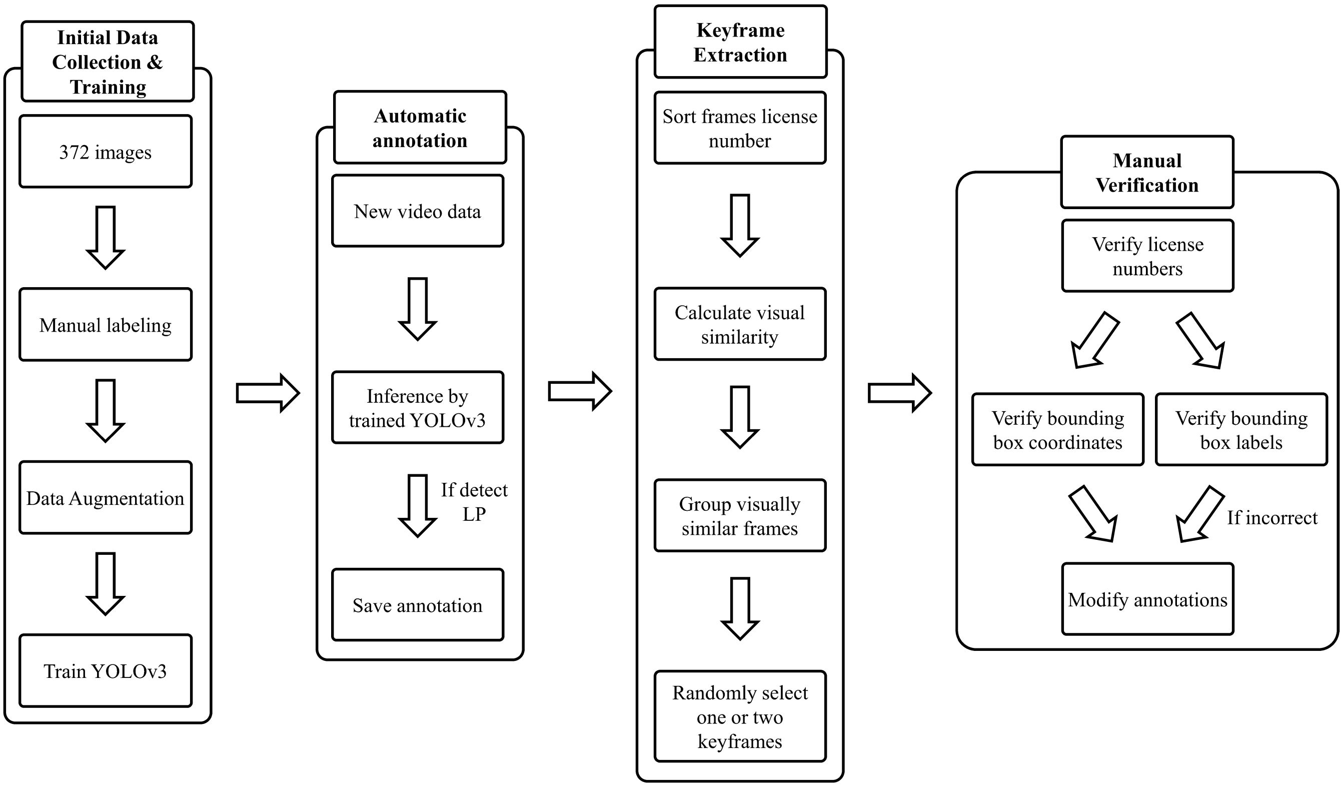
Initial Data Collection & Training
372 images
Manual labeling
Data Augmentation
Train YOLOv3
Automatic annotation
New video data
Inference by trained YOLOv3
If detect LP
Save annotation
Keyframe Extraction
Sort frames license number
Calculate visual similarity
Group visually similar frames
Randomly select one or two keyframes
Manual Verification
Verify license numbers
Verify bounding box coordinates
Verify bounding box labels
If incorrect
Modify annotations

FIGURE 1. Block diagram of semi-automatic data generation strategy.

using Hough transform when testing on subset RP to tackle the rotated LPs.

III. SEMI-AUTOMATIC DATA GENERATION STRATEGY

Most Korean license plate datasets are not available publicly. Therefore, we create our own Korean car plate (KarPlate) dataset by using our own semi-automatic dataset generation strategy as illustrated in Fig. 1. The details about our dataset generation strategy are described below:

A. INITIAL DATA COLLECTION AND TRAINING

We collected image data from a car parking lot in South Korea. This initial dataset consists of 372 images – each image containing a distinct car. We manually annotate this data using an open-source image labeling software known as LabelImg [24]. After labeling our data, we generated two datasets – one consists of full car images with bounding box annotations for LP while the other dataset consists of images of cropped LPs with bounding box annotations for every character.

After dataset annotation, we augmented our dataset by using the augmentation strategy shown in Fig. 2. We designed this augmentation strategy in a way that enhances the robustness of our system across varying conditions. The system was trained by using the augmented images. Our networks were trained to an extent where it could be tested on unseen data with enough accuracy. The weights of the trained network were saved and used in the next step for automatic annotation of unseen data.

B. AUTOMATIC ANNOTATION

We collected raw video data for many days from a CCTV camera installed at a car wash facility in South Korea. From the raw video data, we extracted video clips which contain cars and merge all clips into one clip of about 32 hours duration.

The merged video clip was tested on our network which was trained in the previous step. If the system detected a LP in the frame, a script saves the frame along with the detected bounding box annotations and LP number. A total of 471,981 frames along with annotations and LP numbers were saved after running inference on video data.

However, these 471,981 images contained many redundant frames. These frames looked identical in appearance. It is important to extract only unique frames with distinct appearance.

C. KEYFRAME EXTRACTION

This step deals with the automatic selection of unique frames (keyframes) and deletion of the redundant ones. The primary goal of this step is to discriminate keyframes I_{key} from redundant frames I_r. Simply, the keyframes are found by calculating the similarity among the two images using correlation coefficient r. The overall approach is shown in Fig. 3.

Given a sequence I_{seq} of images containing n number of images, the first frame was set as the initial reference frame I_{ref}. The histogram of the reference frame H_{ref} was calculated and recursively compared with the histogram of

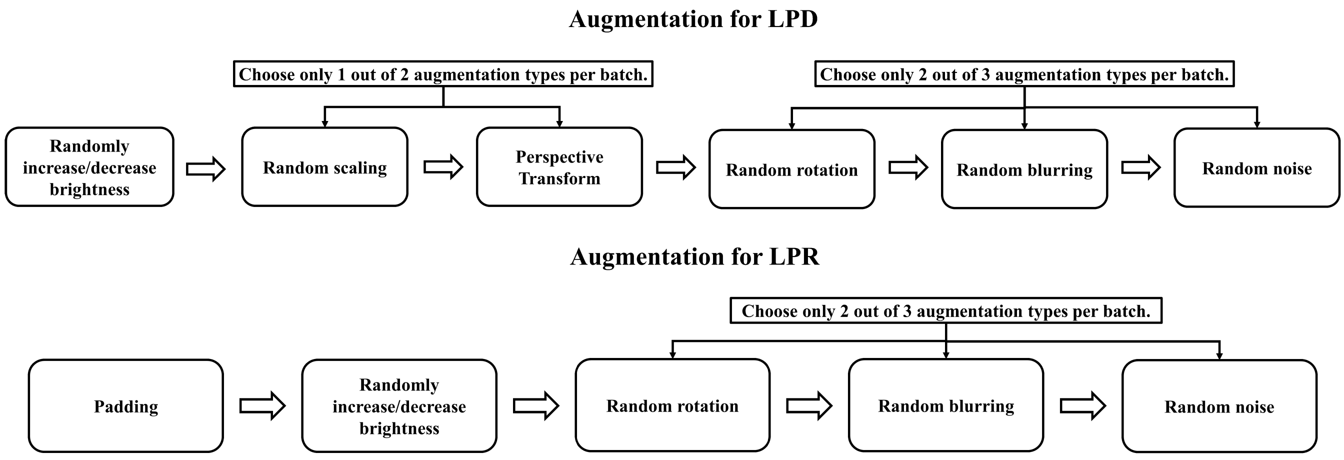
Augmentation for LPD
Choose only 1 out of 2 augmentation types per batch.
Choose only 2 out of 3 augmentation types per batch.
Randomly increase/decrease brightness
Random scaling
Perspective Transform
Random rotation
Random blurring
Random noise
Augmentation for LPR
Choose only 2 out of 3 augmentation types per batch.
Padding
Randomly increase/decrease brightness
Random rotation
Random blurring
Random noise

FIGURE 2. Proposed data augmentation strategy.

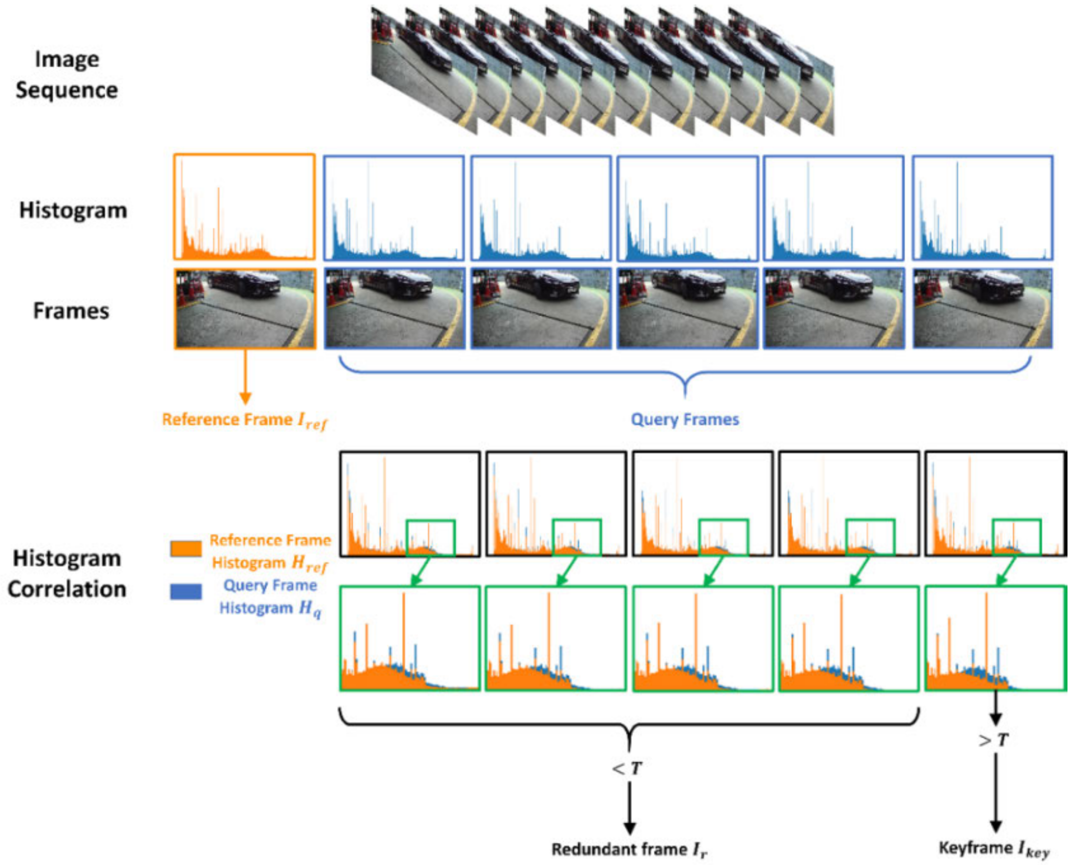

FIGURE 3. Keyframe extraction.

the following query frame H_q until the distance d between the two histograms reaches a threshold value T. Correlation coefficient was used as the distance measure. The criteria used to discriminate among frames is defined as follows:

$$I_{seq}^{n} = \begin{cases} I_{key} & \text{if } d > T \\ I_r & \text{if } d < T \end{cases} \tag{1}$$

$$\begin{aligned} r &= d(H_{ref}, H_q) \\ &= \frac{\sum_{i=0}^{255}(H_{ref}(i) - \overline{H_{ref}})(H_q(i) - \overline{H_q})}{\sqrt{\sum_{i=0}^{255}(H_{ref}(i) - \overline{H_{ref}})^2 \sum_{i=0}^{255}(H_q(i) - \overline{H_q})^2}} \end{aligned} \tag{2}$$

where, I_{seq}^{n} is the nth image in I_{seq}, $H_{ref}(i)$ and $H_q(i)$ denote the normalized frequency at pixel value i in the histograms

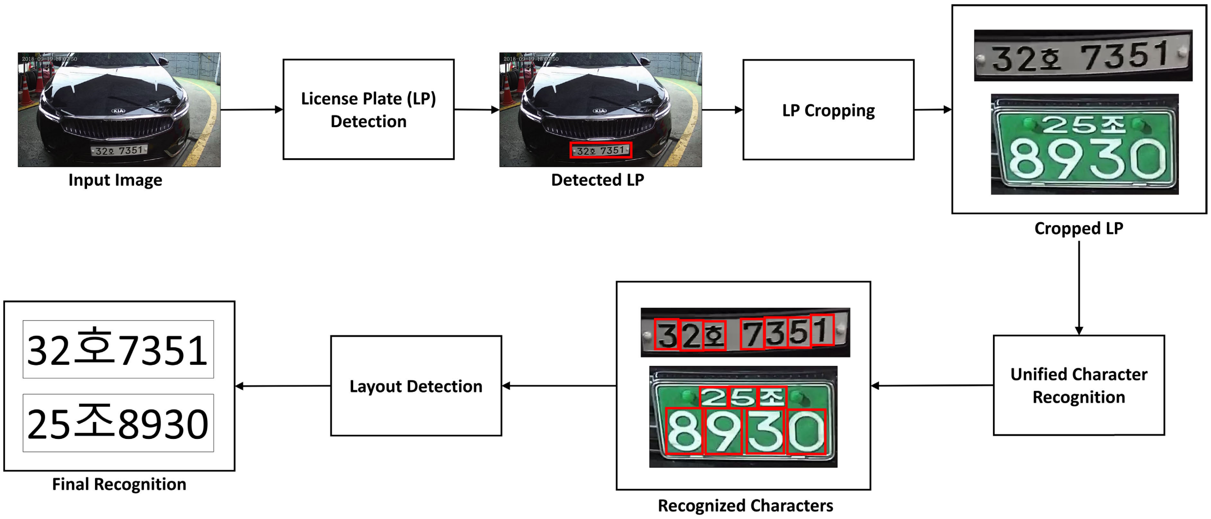

FIGURE 4. Block diagram of Proposed ALPR System.

H_q and H_{ref}, respectively, and $\overline{H_{ref}}$ and $\overline{H_q}$ represent the arithmetic mean of H_q and H_{ref}, respectively.

If d is below the threshold value, then that query frame is considered as redundant and the comparison is continued with the following query frame. On the other hand, if d exceeds the threshold then the query frame is considered as a keyframe. As soon a keyframe is found, the initial reference frame is switched with the found keyframe and the process continues until no frame remains in I_{seq}. A total of 3,893 keyframes were extracted from the 471,981 images by using keyframe extraction.

D. MANUAL VERIFICATION

In this step, we manually verify the annotations of the 3,893 extracted keyframes which were automatically annotated. This was to ensure that all the annotations were correct. Firstly, we verified the LP numbers of each image. This was followed by verification of bounding box coordinates and bounding box labels. In case of any incorrect annotation, the image's annotation file was corrected and updated.

IV. PROPOSED ALPR SYSTEM

In this section, we describe the working of our proposed ALPR approach. Our system is primarily based on YOLOv3 [22]. The overall block diagram of our algorithm is illustrated in Fig. 4. The proposed algorithm can be divided into three main steps:

1) License Plate Detection
2) Unified Character Recognition
3) Multinational License Plate Layout Detection

A. LICENSE PLATE DETECTION

This step of the algorithm deals with the detection and localization of the LP. The primary goal of this stage is to constrain or restrict the search area for the character recognition step. The localized LP plays is a crucial role in reducing the number of false character detections outside the LP area. We propose to use the YOLO network architecture in this stage of our system. Specifically, we use tiny YOLOv3 for this step of our proposed algorithm. Tiny YOLOv3 is a smaller version of YOLOv3 and achieves high FPS at the expense of decrease in mAP score. Tiny YOLOv3 is feasible for our application since LP detection is a simpler task as compared to LP recognition.

The YOLO object detector works by splitting an image into an $S \times S$ sized grid. For each grid cell, K number of bounding boxes along with the confidence scores $Pr(\text{Object}) \times IOU^{\text{truth}}_{\text{pred}}$ are predicted. The confidence score indicates the extent to which the model is confident about the existence of an object. Confidence score is zero in the absence of an object. In the presence of an object, confidence score is equal to the IOU between the predicted and the ground truth bounding box. A conditional class probability score $Pr\,(\text{Class}_i|\text{Object})$ is also predicted for each grid cell containing an object. The class-specific confidence score for each box is calculated using Equation 3 and is encoded as an $S \times S \times (K \times (5 + C))$ tensor.

$$PR\,(Class_i|Object) * \Pr(Object) * IoU^{truth}_{pred} = \Pr\,(Class_i) * IoU^{truth}_{pred} \qquad (3)$$

The main reason for selection of YOLO is its real-time performance and high accuracy. Specifically, we utilize the latest YOLOv3 instead of YOLOv2 even though YOLOv2 achieves higher frames per second (fps). This is because YOLOv3 achieves a higher mean average precision (mAP) score and makes predictions at three different scales. An accurate system with a reasonable level of real-time speed is preferred over a high fps system with a low accuracy. The ability of YOLOv3 to make predictions at varying scales preserves fine features which enables the network to detect small objects. Previous versions of YOLO struggled to detect small objects. The YOLOv3 used in this step is referred to as the attention network since it extracts important region for the recognition network. The network was trained on images resized at 640 × 640 for KarPlate subset LPD, AOLP [26], and Medialab [29] datasets. In case of Caltech Cars [28] and University of Zagreb [30] datasets, network input image size of 704 × 416 and 640 × 480 was selected based on LP aspect ratio, respectively. We changed the number of filters used in the last convolution layer of YOLOv3. The number of filters depends upon the number of classes (1 in this case) to predict and is calculated by using Equation 4. YOLO utilizes anchors A to predict bounding boxes with coordinates *coords* (x, y, w, h). The default value of A was used in our experiments. YOLO requires a confidence threshold value. An object's location will be returned only if it is over the confidence threshold. Sometimes, YOLO predicts other similar objects (like billboards, etc.) as LP. To solve this issue and improve LP detection and localization, we added negative image samples during training.

$$no.\ of\ filters = (no.\ of\ classes + coords + 1) \times A \quad (4)$$

B. UNIFIED CHARACTER RECOGNITION

This step is responsible for recognizing characters in the extracted LP from the previous step. Contrary to most previous works, which address character recognition as a two-step problem (segmentation and recognition), we pose character recognition as an object recognition problem. Using object recognition, we unify the character segmentation and recognition step into one by treating characters as objects. This stage uses YOLOv3-SPP which is an improved version of YOLOv3 that uses spatial pyramid pooling (SPP) block [23].

Contrary to traditional pooling, spatial pyramid pooling splits a feature map into $B_i = n_i \times n_i$ bins where B_i denotes the number of bins in the i-th layer of the pyramid. The feature maps are then pooled by using max pooling into the same size as its bin. This produces an $N \times B$ vector, where N denotes the number of filters in the convolution layer and B denotes the number of bins. By pooling in local spatial bins SPP generates fixed-length vectors by pooling the features together. The filter size in traditional pooling is fixed whereas the filter size in SPP depends upon the input and output size. The use of SPP block has shown to improve performance in various CNNs since SPP handles multi-scale images effectively.

The network was trained on images resized at 384×192, 288×224, 256×224, 384×224, and 384×224 for KarPlate subset LPR, AOLP [26], Caltech Cars [28], Medialab [29], and University of Zagreb datasets, respectively. The network input image sizes were chosen based on the aspect ratios of LPs. For the KarPlate LPR dataset, the network was trained to recognize 45 classes. Korean LPs consist of 35 Korean alphabetic (Hangul) characters and 10 numerical characters. The details about the 35 Hangul and 10 numerical characters are shown in Table 1. In case of other datasets, the network was trained to recognize 36 classes (A-Z and 0-9). Like the preceding step, YOLO filters were recalculated for improving object detection. Since the number of characters n_c in a specific country's LP are known, we use this information to filter out the top n_c characters from the detected characters. This way characters with low confidence scores can be filtered out and false positive can be reduced. In case the number of characters is variable, we compute IOU among the detected objects and reject the object with lower confidence if the IOU among two bounding boxes is greater than a threshold.

TABLE 1. Table showing the Hangul and numerical characters used in Korean LPs.

Type	Characters
Korean alphabetic (Hangul)	가 (Ga), 나 (Na), 다 (Da), 라 (Ra), 마 (Ma), 거 (Geo), 너 (Neo),더 (Deo), 러 (Reo), 머 (Meo), 버 (Beo), 서 (Seo), 어 (Eo), 저 (jeo), 고 (Go), 노 (No), 도 (Do), 로 (Ro), 모 (Mo), 보 (Bo), 소 (So), 오 (O), 조 (Jo), 구 (Gu), 누 (Nu), 두 (Du), 루 (Ru), 무 (Mu), 부 (Bu), 수 (Su), 우 (U), 주 (Ju), 허 (Heo), 하 (Ha), 호 (Ho)
Numerical	0, 1, 2, 3, 4, 5, 6, 7, 8, 9

C. MULTINATIONAL LICENSE PLATE LAYOUT DETECTION

There is a specific layout for every country's LP. Every LP number should be extracted in the correct order. The output from the recognition network does not provide information about the order of the LP number. Therefore, certain heuristics are required to extract the final number. Most previous works design methods for extracting the correct order of license number. However, these methods work only on specific countries' LPs and fail when applied to other countries' LPs. In this study, we propose an algorithm for extracting the correct order of the LP number that generalizes to multinational LPs.

In order to develop a universal algorithm, we first analyze the layouts of LPs existing in the world. We examine LPs belonging to various countries from all continents of the world (except Antarctica). Particularly, we analyze LPs from 17 countries belonging to different continents as shown in Fig. 5. Based on our analyses, we observed that most LPs in the world can be classified as single line or double line license plate as shown in Fig. 5.

The block diagram of our algorithm is shown in Fig. 6. First, we sought all the recognized bounding boxes by x_n^{TL} (top left x coordinate). Let *bboxes* be the sorted list of lists in ascending order.

$$\begin{aligned} bboxes = {} & (x_1^{TL}, y_1^{TL}, x_1^{BR}, y_1^{BR}), \\ & (x_2^{TL}, y_2^{TL}, x_2^{BR}, y_2^{BR}) \ldots, [(x_n^{TL}, y_n^{TL}), (x_n^{TL}, y_n^{BR})] \end{aligned} \quad (5)$$

where x_n^{TL} and y_n^{TL} represent the top left x and y coordinate whereas x_n^{BR} and y_n^{BR} represent the bottom right x and y coordinate of n-th bounding box. After sorting, we make line segments on the left side of all bounding boxes. The end

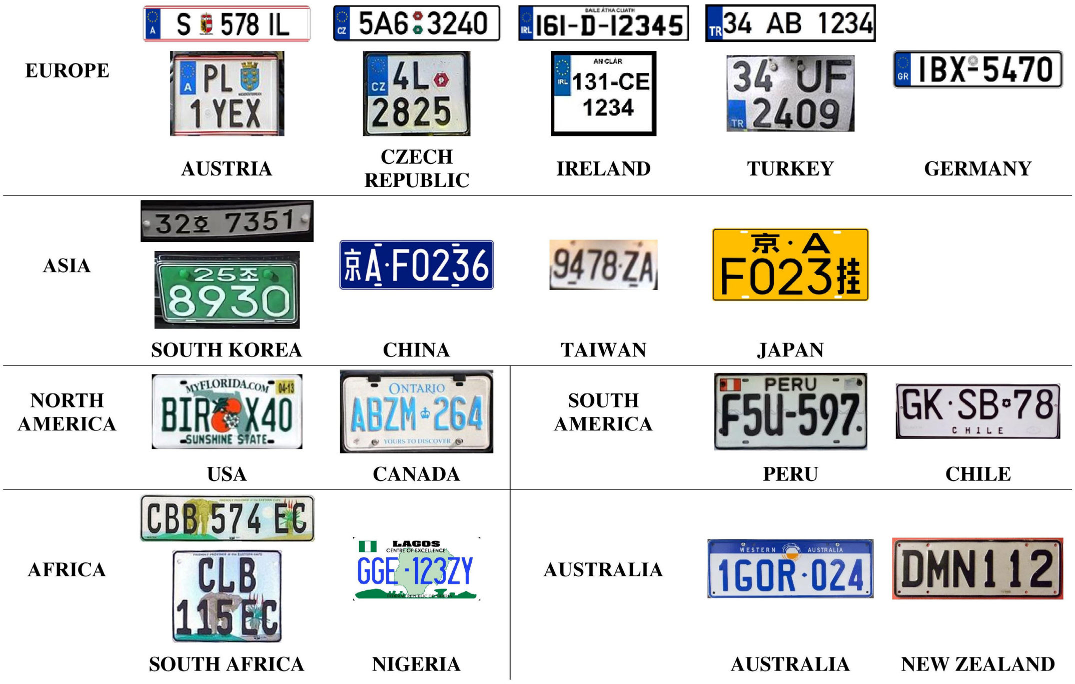

FIGURE 5. License plates from 17 different countries belonging to various continents.

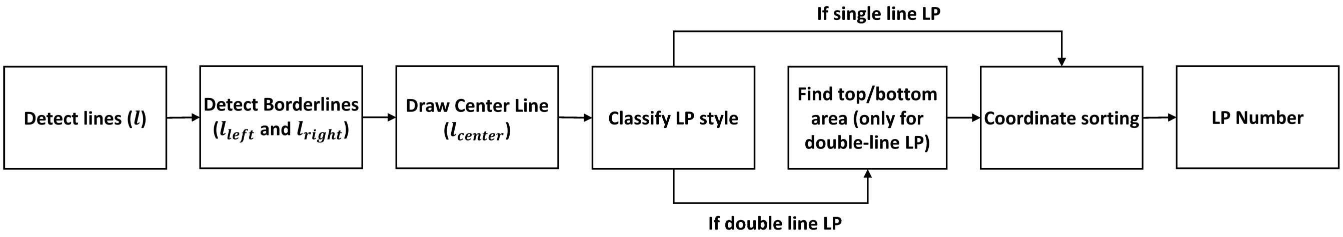
Detect lines (l)
Detect Borderlines (l_{left} and l_{right})
Draw Center Line (l_{center})
Classify LP style
If single line LP
Find top/bottom area (only for double-line LP)
If double line LP
Coordinate sorting
LP Number

FIGURE 6. Block diagram of multinational license plate layout detection.

points of the line segments l (see blue line in Fig. 7 and 8) are given by:

$$l = [(x_1^{TL}, y_1^{TL}), (x_1^{TL}, y_1^{BR})], [(x_2^{TL}, y_2^{TL}), (x_2^{TL}, y_2^{BR})], \ldots, [(x_n^{TL}, y_n^{TL}), (x_n^{TL}, y_n^{BR})] \quad (6)$$

This is followed by extracting the left (l_{left}) and right (l_{right}) borderline (see yellow line in Fig. 7 and 8). Since the bboxes list is sorted in ascending order, therefore, the end points of l_{left} and l_{right} borderline can be found by using the first and the last bounding box coordinates in *bboxes*, respectively. The coordinates for l_{left} and l_{right} are:

$$l_{left} = (x_1^{TL}, y_1^{TL}), (x_1^{TL}, y_1^{BR}) \quad (7)$$

$$l_{\text{right}} = (x_n^{TL}, y_n^{TL}), (x_n^{TL}, y_n^{BR}) \quad (8)$$

Then, we draw a line segment l_{center} (see purple line in Fig. 7 and 8) with the midpoints of the borderlines (l_{left} and l_{right}) as its endpoints. The end points of l_{center} are:

$$l_{\text{center}} = (x_1^{TL}, \frac{y_1^{BR} - y_1^{TL}}{2}), (x_n^{TL}, \frac{y_n^{BR} - y_n^{TL}}{2}) \quad (9)$$

Next, we need to check the number of line segments intersecting the center line l_{center}. If l_{center} intersects all the line segments, then the LP is a single line LP (see Fig. 7's right most image). On the contrary, if l_{center} intersects a few line segments then the LP is a double line LP (see Fig. 8's right most image). To find the number of intersections, we generate line equations in general form ($ax+by = c$) by converting the end points of line segments (l). The intersection point $P(x, y)$ can be calculated by:

$$P(x, y) = (\frac{c_1b_2 - c_2b_1}{a_1b_2 - a_2b_1}, \frac{a_1c_2 - a_2c_1}{a_1b_2 - a_2b_1}) \quad (10)$$

where a_1, b_1, and c_1 represent the coefficients of one line while a_2, b_2, and c_2 represent the coefficients of the other line. Note that equation 10 finds the point of intersection for

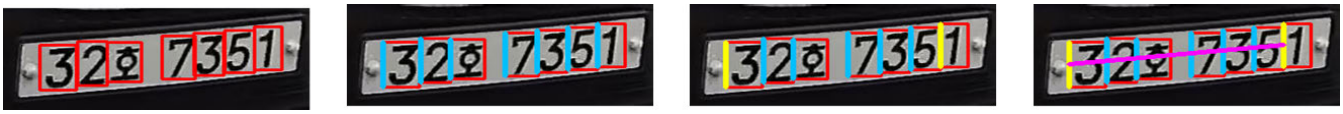

FIGURE 7. Explanatory image for single line LP. From left to right: input image with bounding boxes around detected characters, detected lines (see blue lines), detected borderlines (see yellow lines), and the center line (see purple line).

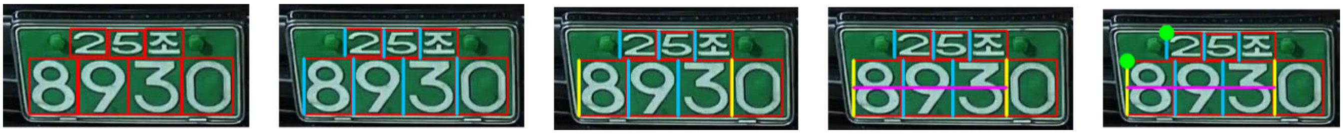

FIGURE 8. Explanatory image for double line LP. From left to right: input image with bounding boxes around detected characters, detected lines (see blue lines), detected borderlines (see yellow lines), center line (see purple line), and the top and bottom line (see green dots).

infinitely long lines defined by end points, rather than the line segments between the end points. Therefore, to find whether an intersection point exists within the line segments, we apply the following criterion:

$$f(x) = \begin{cases} 1, & \text{if } \min\left(x_n^{TL}, x_n^{BR}\right) \leq P(x) \leq \max\left(x_n^{TL}, x_n^{BR}\right) \\ & \text{and } \min\left(y_n^{TL}, y_n^{BR}\right) \leq P(y) \leq \max\left(y_n^{TL}, y_n^{BR}\right) \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

If $f(x) = 1$, it means that an intersection point exists within the two line segments while $f(x) = 0$ means that an intersection point does not exist within the two line segments. Based on the number of line segments that intersect l_{center}, we can find the type of LP by using the criterion:

$$g(x) = \begin{cases} 1, & if N_{int} = N_{bboxes} - 2 \\ 2, & otherwise \end{cases} \tag{12}$$

where N_{int} is the number of intersections and N_{bboxes} is the total number of bounding boxes in *bboxes*. The function $g(x)$ returns a 1 for a single line LP and 2 for double line LP. For a single-line LP, the final number will have the same order as *bboxes*. For double-line LP, the bounding boxes lying on the top and bottom area of the LP must be found. To find the bounding boxes lying in the top part of the LP, the following criterion is applied:

$$top_{line} = \begin{cases} bboxes_{int}, & if\ y_1^{TL} \in bboxes_{int} < y_1^{TL} \in bboxes_{\overline{int}} \\ bboxes_{\overline{int}}, & if\ y_1^{TL} \in bboxes_{int} > y_1^{TL} \in bboxes_{\overline{int}} \end{cases} \tag{13}$$

where $bboxes_{int}$ and $bboxes_{\overline{int}}$ are lists of bounding boxes that intersect and do not intersect l_{center}, respectively. $bboxes_{int}$ and $bboxes_{\overline{int}}$ are sorted by x_n^{TL} (top left x coordinate) in ascending order. $bboxes_{int}$ includes bounding boxes that are associated with the borderlines (l_{left} and l_{right}). y_1^{TL} is the top left y coordinate of the $1st$ bounding box in $bboxes_{int}$ or $bboxes_{\overline{int}}$. The bounding boxes in the top area will be written first while the ones in the bottom area will be written at the end in the final string.

V. EXPERIMENTAL RESULTS AND DISCUSSION

A. DATASETS

The proposed approach was validated on datasets containing license plates from South Korea, Taiwan, United States of America, Greece, and Croatia. Dataset details about the Korean (KarPlate dataset), Taiwanese (AOLP dataset [26]), American (Caltech Cars (Rear) 1999 [28]), Greek (Medialab LPR Database [29]), and Croatian (University of Zagreb [30]) datasets are briefly described in the following subsections. The publicly available datasets [26], [28]–[30] were manually annotated since none of these datasets provided bounding box annotations for LP detection (except for AOLP dataset) or character recognition.

1) KARPLATE DATASET (SOUTH KOREA)

The KarPlate (Korean car plate) dataset is publicly available and can be downloaded from our project webpage[1]. The details about our dataset generated using the semi-automatic dataset generation strategy will be discussed in this section. The KarPlate dataset is divided into subset LPD, subset LPR, and subset EER. Subsets LPD and LPR each contain 3,417 images for training and 850 images for testing. Subset EER contains only 929 test images along with the LP numbers. Among the 929 images in subset EER, 850 images are same as the test images in subset LPD while the remaining 79 images were captured by a mobile phone's camera. These 79 images are relatively more challenging since each image contains multiple LPs.

Each subset is intended to be used for a specific task. Subset LPD is for license plate detection, subset LPR is for license plate recognition, and subset EER is for end-to-end recognition. Each image in subset LPD and subset EER have a resolution of 1920×1080. The train-split is augmented using the augmentation strategy shown in Fig. 2. After augmentation, more than 30,000 and 60,000 images for training were generated using subset LPD and subset LPR, respectively. The annotations of the augmented images were also augmented using the imgaug library [25]. Sample images from each subset can be visualized in Fig. 10. It should be

[1] http://pr.gachon.ac.kr/ALPR.html

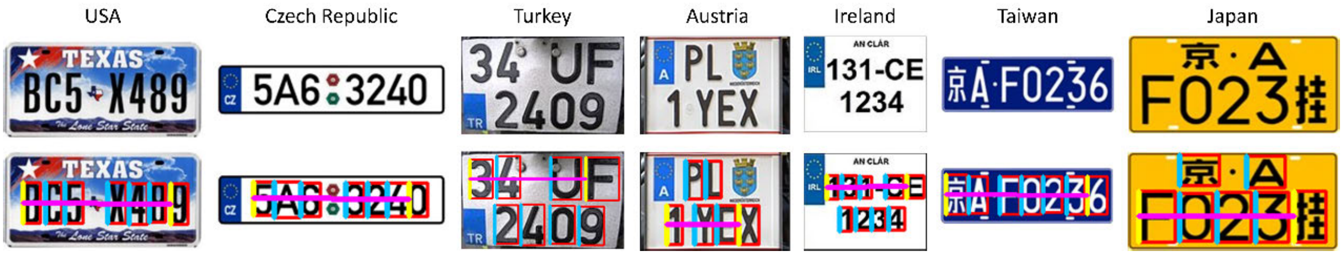

FIGURE 9. Figure showing that our proposed criterion for classifying single and double lined LP is true for LPs from various countries.

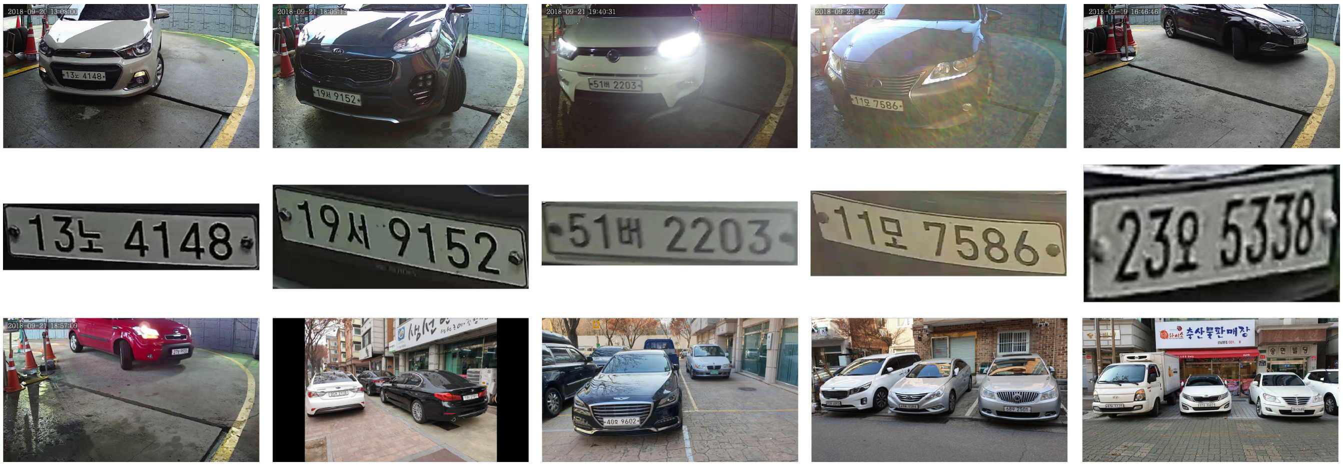

FIGURE 10. Samples from KarPlate dataset. From top to bottom; images from subset LPD, images from subset LPR, and images from subset EER.

noted that our dataset does not contain images of commercial and rental vehicles in South Korea since none passed through our camera setup.

2) AOLP DATASET (TAIWAN)

The application-oriented license plate (AOLP) [26] database is a public dataset consisting of 2,049 images of Taiwanese license plates. The dataset is divided into access control (AC), law enforcement (LE), and road patrol (RP) subsets. Subset AC, subset LE, and subset RP consist of 681, 757, and 611 images, respectively. License plates in each subset present varying application parameters (like tilt, width ratio, distance, etc.) depending upon the application case. Subset AC comprises of images of vehicles passing through fixed passages such as toll stations. Images captured by roadside cameras which are used for checking traffic law violations are included in Subset LE. Lastly, subset RP contains images taken by handheld cameras which are used for finding parking violations, searching lost vehicles, etc. Following previous works, two subsets were used for training while the third subset was used for testing.

3) MEDIALAB LPR DATABASE (GREECE)

The Medialab LPR database consists of 716 images containing Greek license plates and is provided by the National Technical University of Athens. It is divided into a normal subset and a difficult subset which covers situations like shadows, low-light, blur, dirt, etc. The normal subset contains 437 images while the difficult subset contains 279 images.

Considering the previous works, normal subset was used for testing and difficult subset was used for training our system. However, it must be noted that the difficult subset contains only 279 images out of which 20 images contain unreadable license plates due to small image dimensions. This leaves us with 259 images which are extremely less for training our deep neural networks even after applying data augmentation. To tackle this issue, we used 501 images from the University of Zagreb [30] dataset, 108 images from the OpenALPR Europe dataset [31], and 1,428 images from ReId dataset [32]. All the images taken from other datasets contain European LPs. For testing, 431 out of 437 images were used from the normal subset since the remaining images contained unreadable LPs. The characters ‘1’ and ‘I’ were trained as a single class due same appearance in Greek LPs. ‘0’ and ‘O’ were also trained as single class due to the same fact. These characters were swapped during the testing phase since the first two or three characters in Greek LPs are always letters while the remaining four characters are always digits.

4) CALTECH CARS (REAR) 1999 DATASET (USA)

The Caltech Cars (Rear) 1999 dataset contains 126 images of vehicles containing license plates from different states of USA. The images have a resolution of 896×592 pixels and were captured at Caltech parking lot with a cluttered

TABLE 2. LP detection comparison in terms of precision (P) and recall (R) at IOU of 0.5.

Approach	AOLP			MediaLab	Caltech Cars	University of Zagreb	KarPlate Subset LPD
	AC	LE	RP				
	P/R	P/R	P/R	P/R	P/R	P/R	P/R
[34]	-	-	-	87.68/91.30	-	-	-
[1]	98.53/98.38	97.75/97.62	95.28/95.58	-	-	-	-
[8]	99.51/99.51	99.43/99.43	99.46/99.46	-	-	-	-
[17]	100.00/99.53	99.01/98.62	95.67/95.71	-	-	-	-
[35]	-	-	-	-	98.64/97.95	-	-
[36]	-	-	-	-	99.13/99.13	-	-
Proposed	98.98/100.00	97.70/98.96	98.23/100.00	98.85/99.76	100.00/100.00	98.01/99.00	100.00/100.00

background. The dataset was randomly spitted into 80 images for training and 46 images for testing. The train-test split was inspired by previous works. In addition to the 80 images, 244 images from OpenALPR US dataset, 108 images from OpenALPR Europe dataset [31], and 501 images from the University of Zagreb [30] dataset were used for training the character recognition network. On the other hand, only OpenALPR US dataset [31] was used for training the LP detection network.

5) UNIVERSITY OF ZAGREB DATASET (CROATIA)

This database comprises of 510 images of vehicles containing Croatian license plates collected by the University of Zagreb, Croatia. Among the 510 images, 9 images were discarded, and the remaining 501 images were utilized for training and testing. The dataset was randomly spitted which resulted in 401 images for training and 100 images for testing. 108 images from OpenALPR Europe dataset [31] were also used for training. The characters '0' and 'O' were trained and tested as a single due to the same appearance in Croatian license plates.

B. IMPLEMENTATION DETAILS

The proposed system was trained and tested on a personal computer containing an Intel Core i7-4770 processor along with NVIDA Titan X Pascal and 24 gigabytes of RAM. AlexayAB's version of Darknet [33] was used to train the YOLO networks. The overall system was programmed using python and a python wrapper was used for incorporating Darknet in the system. Imgaug [25] was used for data augmentation along with the bounding boxes.

C. EXPERIMENTAL RESULTS

The proposed method was evaluated on five LP datasets each belonging to a different country to validate the effectiveness of our approach. The performance evaluation is divided into five sections – license plate detection, license plate recognition, end-to-end recognition, multinational LP layout detection, and time consumption. The sections are described below:

1) LICENSE PLATE DETECTION

The LP detection performance was evaluated on AOLP [26], Caltech Cars (Rear) 1999 [28]), Medialab LPR Database [29], University of Zagreb [30], and KarPlate subset LPD datasets in terms of precision and recall and is presented in Table 2. Precision can be calculated by dividing the number of correctly detected LPs by the total number of detected LPs. Recall can be calculated by dividing the number of correctly detected LPs by the total number of ground truth LPs. Precision and recall are mathematically defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{14}$$

$$Recall = \frac{TP}{TP + FN} \tag{15}$$

where TP, FP, and FN represent true positive, false positive, and false negative, respectively. The detected LP bounding box is considered correct if it there is an overlap greater than 0.5 between the predicted and the ground truth bounding box. Considering the evaluation criterion of previous works, we select the threshold value of 0.5 for a fair comparison.

The LP detection performance was compared the recent methods [1], [8], [17], [34]–[36]. It is evident from Table 2 that the proposed approach outperformed most previous works on the five datasets. The proposed approach outperforms the work by [34] with precision/recall rate of 98.85/99.76% on the Medialab dataset [29]. Similarly, a high precision/recall rate of 100.00/100.00% surpassed the results by the methods in [35-36] on the Caltech Cars dataset [28]. A high precision/recall rate of 98.01/99.00 and 100.00/100.00 was obtained on the University of Zagreb [30] and KarPlate subset LPD datasets, respectively. In addition, our system achieves high recall rate of 100.00% on both AOLP subset AC and AOLP subset RP.

2) LICENSE PLATE RECOGNITION

In this section, the LP recognition performance is compared with the previous works [1], [11], [13], [17], [37]. The accuracy shown in this section corresponds to the combined accuracy of the character recognition network and the layout detection algorithm. It should be noted that the input to the character recognition network is the cropped LP image. Given the cropped LP image, the character recognition network detects characters in the image which are later fed into the layout detection algorithm resulting in a character sequence which represents the LP number. The result is considered as correct only if all the characters are detected correctly and the LP number is in the correct sequence. The result is declared incorrect if certain characters are not detected or if more characters are detected.

TABLE 3. LP recognition comparison in terms of accuracy (%).

Approach	AOLP				MediaLab	Caltech Cars	University of Zagreb	KarPlate Subset LPR
	AC	LE	RP	Average				
LSTM [1]	94.85	94.19	88.38	92.47	-	-	-	-
DeepFCN [37]	97.9	97.6	98.2	97.9	-	-	-	-
CR-Net [11]	-	-	79.21	-	-	-	-	-
DeepLabv2 ResNet-101 [13]	99.41	99.31[a]	99.02	99.25	97.89[b]	-	-	-
Sliding-window YOLO [17]	-	-	-	78	-	-	-	-
Proposed	99.41	99.1	99.53	99.34	98.36[b]	95.65	98	98.59

[a] Accuracy calculated only for 582 out of 757 images in AOLP subset LE.

[b] Accuracy calculated only for 427 out of 437 images in Medialab normal subset.

TABLE 4. End-to-end performance comparison in terms of accuracy (%).

Approach	AOLP				MediaLab	Caltech Cars	University of Zagreb	KarPlate Subset EER
	AC	LE	RP	Average				
OpenALPR [14]	58.88	67.5	64.98	63.78	96.52	89.13	93	97.95
Sighthound [2]	95.01	98.01	90.18	94.4	83.29	95.65	96	-
BRNNs [6]	95.29	96.57	83.63	91.83	-	-	-	-
YOLOv2 + WPOD-Net [11]	-	-	98.36	-	-	-	-	-
YOLOv2 + CR-Net [36]	-	-	-	-	-	98.7	-	-
Proposed	99.41	97.88	99.51	98.93	96.98	97.83	97	98.17

The LP recognition results on the AOLP, Medialab, Caltech Cars, University of Zagreb, and KarPlate subset LPR datasets were compared with the results from previous works [1], [11], [13], [17], [37] and are presented in Table 3. The proposed approach achieves high accuracy on all the five datasets. The results show that our method achieves an average accuracy of 99.34% on the AOLP dataset outperforming the previous state-of-the-art method [13]. Our approach performed well on all subsets of AOLP dataset. It should be noted that the accuracy for AOLP subset LE by [13] was calculated only for 582 out of 757 images whereas we present result for all 757 images in AOLP subset LE. Hence, our method is superior in results when compared to the results in [13]. In case of AOLP subset RP, the toughest subset in AOLP dataset, we achieved an accuracy of 99.53%. It should be noted that the study in [7] used Hough transform to straighten the LPs as a preprocessing step before inputting it to their proposed algorithm. Our algorithm can effectively handle rotated LPs without requiring any pre-processing. The study in [11] did not include results on subset AC and subset LE while the study in [17] only provided average accuracy, hence, our results could not be compared with theirs.

It must be considered that the accuracy mentioned by [13] for Medialab dataset was calculated only for 427 images. Hence, for a fair comparison, the accuracy of 98.36% presented in Table 3 for proposed approach was calculated only for 427 images. The accuracy achieved by our method for 431 images was 97.75%. In case of Caltech Cars, University of Zagreb, and KarPlate subset LPR datasets, an accuracy of 95.65, 98.00, and 98.59 was achieved, respectively. To the best of our knowledge, none of the previous works mentioned accuracy only for LP recognition on Caltech Cars and University of Zagreb datasets.

3) END-TO-END RECOGNITION

This section evaluates the end-to-end performance of our approach. The end-to-end performance indicates the overall performance of the proposed solution. The uncropped LP image is passed into our three-step algorithm and the result is the LP number. Note that the input image is not a cropped-out LP image. The result is correct if LP is detected correctly, recognized correctly, and layout is detected correctly. Incorrect result at any of the three step leads to an incorrect recognition. Simply, given an uncropped LP image, the resulting character sequence must match the ground truth character sequence.

Table 4 presents the end-to-end performance of our method and compares it with previous academic [6], [11], [36] and industrial works [2], [14]. It is evident that the proposed method surpasses results by commercial software [2], [14] and academic works [6], [11], [36] on all five datasets. It is worth mentioning that commercial software are generally trained on much larger datasets which is huge advantage in deep learning. Our method outperforms previous academic works [6], [11], [36] and both the commercially available software [2], [14] with an average accuracy of 98.93% on the AOLP dataset. For OpenALPR [2], the region was set to Europe when testing AOLP dataset since OpenALPR lacks Taiwan region. Europe was chosen since both European and Taiwanese LPs are single line LPs and use the same characters. The work by [11] used artificial data to achieve the accuracy of 98.36% on AOLP subset RP while ours did not use artificial data. The performance of the work by [11] on the AOLP subset RP drops down to 93.29% without using artificial data while our work achieves 99.51% accuracy without artificial data. Also, the work by [6] cannot deal with rotated LPs in subset RP which is evident from the low 83.63% accuracy.

The proposed method outperformed commercial software [2], [14] and achieved an accuracy of 96.98% and 97.00% on Medialab and University of Zagreb datasets, respectively. An accuracy of 97.83% was obtained by our method as compared to 98.70% by [36] on the Caltech Cars dataset. However, it must be taken into consideration that test data was randomly selected in [36] and in our work.

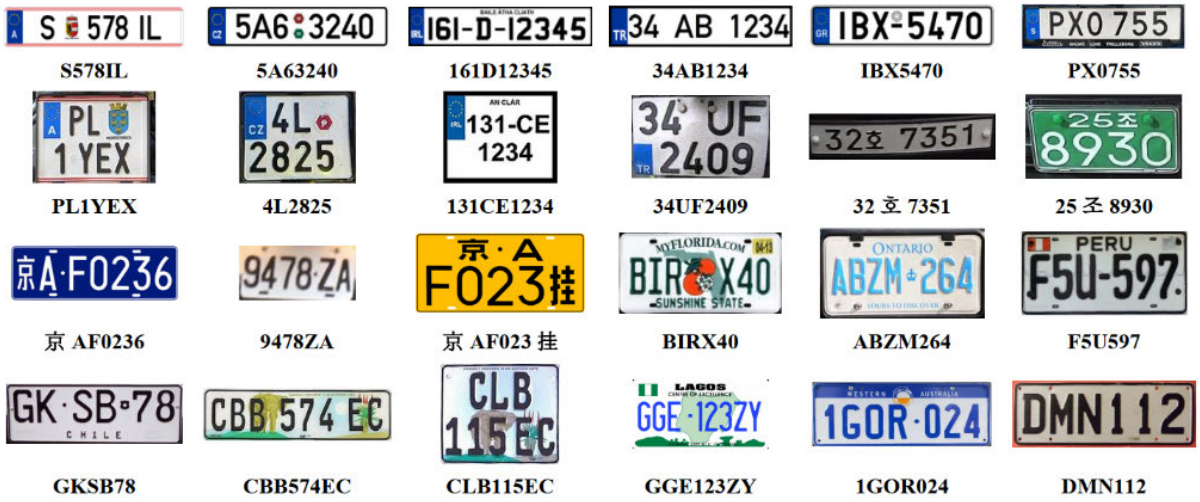

FIGURE 11. The output from the multinational LP layout detection step (see bottom of every LP image shown).

TABLE 5. Time consumption (ms/image) of proposed system for LPD, LPR, and end-to-end recognition.

Dataset	LPD	LPR	End-to-End Recognition
AOLP	15.54	15.65	29.19
MediaLab	30.34	19.21	47.02
Caltech Cars	19.36	14.94	34.3
University of Zagreb	14.69	17.03	32.18
KarPlate Dataset	47.33	15.91	65.12
Average	25.45	16.54	41.56

TABLE 6. Average execution time for end-to-end recognition assuming a certain number of LPs/vehicles exist in an image. The execution time was tested on KarPlate dataset (subset EER).

No. of Vehicles	Time (ms/image)
1	63.62
2	78.39
3	93.10

It may be possible that relatively difficult images might have been included in our test set which can be responsible for slightly lower accuracy. Lastly, a high accuracy of 98.82% was achieved on KarPlate Subset EER dataset. Since Sighthound [14] cannot detect Korean characters, hence, we could not test KarPlate Subset EER dataset on it.

4) MULTINATIONAL LP LAYOUT DETECTION

We already evaluated the proposed approach on five datasets from different countries. However, to further validate the applicability of our approach to multinational LPs, a demo evaluation on LPs from 17 countries is presented. Demo evaluation is presented since most countries do not have public LP datasets. Even if public LP dataset for a specific country is available, it must be annotated with bounding box annotations since our method deals character recognition as an object recognition problem. Annotating datasets for about 17 countries is extremely time consuming.

Hence, a demo dataset consisting of images each containing LP from a different country (some LPs belong to same country but differ in layout). The purpose of this evaluation is to demonstrate the applicability of our work to multinational LPs without the need of any additional steps. This evaluation assumes that the character recognition network outputs correct results. Fig. 11 shows the resulting correct sequence of characters for the demo dataset. It is obvious from Fig. 11 that the proposed layout detection algorithm can successfully extract the correct sequence of the LP number. However, it is worth mentioning that as time passes by, new LP layouts may be made available and it is possible for the algorithm to fail in that case. We show that our layout detection algorithm can work on most of the LP with different countries.

It is possible that the character recognition network might also recognize the state's name. For instance, 'FLORIDA' in USA LP can be detected as characters even though it might not be a part of the LP number. To cater this issue, such words can either be trained as another class which can be rejected later if recognized or can be used as a negative samples during training the character recognition network.

5) TIME CONSUMPTION

This section evaluates the time consumption of the proposed approach on three tasks. These three tasks include license plate detection (LPD), license plate recognition (LPR), and end-to-end recognition (EER).

The time consumption of our proposed system, presented in this section, was computed using a computer with an Intel i7-4770 processor, NVIDIA GTX Titan X Pascal and 24 gigabytes of RAM. It must be noted that the time consumption for LPR includes the time taken by the layout detection algorithm. Aslo, it is worth mentioning that the time consumption for our approach reported in this paper includes image reading time. The proposed system is faster if image reading time is excluded.

Table 5 presents the time consumption for LPD, LPR, and end-to-end recognition on all five datasets. For each dataset, we take all test images from that dataset and test

TABLE 7. Time consumption (ms/image) comparison with recent works for LPD and EER.

Method	GPU	AOLP		Caltech Cars		University of Zagreb	
		LPD	EER	LPD	EER	LPD	EER
BRNNs [6]	NVIDIA Titan X	-	400	-	-	-	-
[38]	NVIDIA Tesla K40c	-	-	20	-	-	-
		-	-	-	-	20	-
Proposed	NVIDIA Titan X Pascal	15.54	29.19	19.36	34.3	14.69	32.18

it on our ALPR system. The average of the time taken to process each image is reported in Table 5. It takes our system 25.45 ms per image, 16.54 ms per image, and 41.56 ms per image on average for LPD, LPR, and end-to-end recognition. The highest execution time is for KarPlate dataset. This is because the images in KarPlate dataset have a resolution of 1920×1080 pixels which is the largest among the other four datasets. Another reason is that KarPlate subset EER contains images consisting of multiple LPs whereas all other four datasets contain images with single LP.

In practical applications, ALPR systems are required to process images that contain multiple LPs which increases time consumption. Therefore, for a more realistic evaluation, Table 6 presents the time consumption assuming that a certain number of LPs are present in an image. The execution time presented in Table 6 is for KarPlate dataset (subset EER). To compute the execution time for multiple LPs, we had to manually divide the dataset into three subsets based on the number of LPs. For each subset, we test all images in that subset and report the average of the time taken to process each image in Table 6. Execution time for multiple LPs could not be computed for the other four datasets since none of the images in other four datasets contain multiple LPs. It is evident from Table 6 that the execution time increases by a few milliseconds with the increase in the number of LPs. The proposed system can process a full HD image with three LPs in 93.10 ms (about 11 frames per second). Although, 11 frames per second is not blazing fast, however, this is good enough for real-time usage. It is also worth mentioning that the execution time will be decreased if our system is implemented in C++ and if multi-threading is integrated in the source code. Unfortunately, due to time limitation, the results presented in this study were computed using the python implementation of our system which did not use multi-threading.

Table 7 compares time consumption for LPD and end-to-end recognition with recent works [6], [38]. For a fair comparison, the hardware used in previous works [6], [38] is mentioned in Table 7. The time consumption of previous works [6], [38] were taken from the respective studies. It is clear from Table 7 that our approach is faster than previous works [6], [38]. Our approach consumes 29.19 ms for end-to-end recognition on AOLP dataset, which is considerably less when compared to [6]. In case of LPD, the proposed method outperforms [38] with a time consumption of 19.36 ms and 14.69 ms for Caltech Cars and University of Zagreb datasets, respectively. The time consumption for LPR could not be compared since we could not find related works that report time consumption for LPR on the datasets used in this paper.

VI. CONCLUSION

This paper presented a generalized solution for multinational license plate recognition. Our algorithm consists of LP detection, unified character recognition, and multinational LP layout detection steps. LP recognition is posed as an object recognition problem which unifies the character segmentation and character recognition steps. Our system is applicable to license plates from multiple countries by using our proposed multinational LP layout detection algorithm. To the extent of our knowledge, LPs from most of the countries can be broadly classified into single line and double line LPs. The proposed layout detection algorithm is simple, yet it can effectively classify among various LP layouts. Given the correct bounding boxes, our algorithm can effectively extract the correct sequence of LP number from an image. A new Korean car plate (KarPlate) dataset was made publicly available for research purposes. The proposed solution was tested on license plates from South Korea, Taiwan, USA, Greece, and Croatia. Results show that our proposed approach outperforms previous research works and commercial software. In addition, we collected a small demo dataset consisting of LP images from 17 different countries and tested our layout detection algorithm on it. The layout detection algorithm extracted correct sequence of the LP number from 17 countries' LPs. The proposed solution consumes about 42 ms per image on average which is considerably faster than previous works. In a nutshell, our proposed algorithm can work on datasets from multiple countries without the need of any additional algorithms or country-specific information.

ACKNOWLEDGMENT

A preliminary version of this article was presented at the International Conference on Next Generation Computing, Vung Tau, Vietnam, December 2018. The authors would like to thank the assistance of Mr. R. Ghimire and Mr. N. Aryal in dataset labeling.

REFERENCES

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

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

[3] G.-S. Hsu, A. Ambikapathi, S.-L. Chung, and C.-P. Su, "Robust license plate detection in the wild," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Lecce, Italy, Aug. 2017, pp. 1–6.

[4] F. Delmar Kurpiel, R. Minetto, and B. T. Nassu, "Convolutional neural networks for license plate detection in images," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Beijing, China, Sep. 2017, pp. 3395–3399.

[5] O. Bulan, V. Kozitsky, P. Ramesh, and M. Shreve, "Segmentation- and annotation-free license plate recognition with deep localization and failure identification," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2351–2363, Sep. 2017.

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

[7] S. M. Silva and C. R. Jung, "Real-time brazilian license plate detection and recognition using deep convolutional neural networks," in *Proc. 30th SIBGRAPI Conf. Graph., Patterns Images (SIBGRAPI)*, Niterói, Brazil, Oct. 2017, pp. 55–62.

[8] L. Xie, T. Ahmad, L. Jin, Y. Liu, and S. Zhang, "A new CNN-based method for multi-directional car license plate detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 2, pp. 507–517, Feb. 2018.

[9] R. Laroca, E. Severo, L. A. Zanlorensi, L. S. Oliveira, G. R. Goncalves, W. R. Schwartz, and D. Menotti, "A robust real-time automatic license plate recognition based on the YOLO detector," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Rio de Janeiro, Brazil, Jul. 2018, pp. 1–10.

[10] S. Zherzdev and A. Gruzdev, "LPRNet: License plate recognition via deep neural networks," 2018, *arXiv:1806.10447*. [Online]. Available: http://arxiv.org/abs/1806.10447

[11] S. M. Silva and C. R. Jung, "License plate detection and recognition in unconstrained scenarios," in *Proc. ECCV*, Munich, Germany, pp. 593–609.

[12] Z. Xu, W. Yang, A. Meng, N. Lu, H. Huang, C. Ying, and L. Huang, "Towards end-to-end license plate detection and recognition: A large dataset and baseline," in *Proc. ECCV*, Munich, Germany, 2018, pp. 255–271.

[13] J. Zhuang, S. Hou, Z. Wang, and Z. J. Zha, "Towards human-level license plate recognition," in *Proc. ECCV*, Munich, Germany, 2018, pp. 306–321.

[14] *OpenALPR Cloud API*. Accessed: Nov. 5, 2019. [Online]. Available: http://www.openalpr.com/cloud-api.html

[15] M. R. Asif, Q. Chun, S. Hussain, M. S. Fareed, and S. Khan, "Multinational vehicle license plate detection in complex backgrounds," *J. Vis. Commun. Image Represent.*, vol. 46, pp. 176–186, Jul. 2017.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Carson, NV, USA, 2012, pp. 1097–1105.

[17] Hendry and R.-C. Chen, "Automatic license plate recognition via sliding-window darknet-YOLO deep learning," *Image Vis. Comput.*, vol. 87, pp. 47–56, Jul. 2019.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 779–788.

[19] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Hawaii, HI, USA, Jul. 2017, pp. 7263–7271.

[20] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 855–868, May 2009.

[21] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: http://arxiv.org/abs/1602.07360

[22] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[24] *LabelImg*. Accessed: Nov. 1, 2019. [Online]. Available: https://github.com/tzutalin/labelImg

[25] *ImgAug*. Accessed: Nov. 12, 2019. [Online]. Available: https://github.com/aleju/imgaug

[26] G.-S. Hsu, J.-C. Chen, and Y.-Z. Chung, "Application-oriented license plate recognition," *IEEE Trans. Veh. Technol.*, vol. 62, no. 2, pp. 552–561, Feb. 2013.

[27] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 580–587.

[28] (1999). *Caltech Cars (Rear)*. Accessed: Oct. 10, 2019. [Online]. Available: vision.caltech.edu/Image_Datasets/cars_markus/cars_markus.tar

[29] *Medialab LPR Database*. Accessed: Oct. 16, 2019. [Online]. Available: medialab.ntua.gr/research/LPRdatabase.html

[30] *University of Zagreb Dataset*. Accessed: Oct. 10, 2019. [Online]. Available: zemris.fer.hr/projects/LicensePlates/english/baza_slika.zip

[31] *OpenALPR Dataset*. Accessed: Oct. 10, 2019. [Online]. Available: github.com/openalpr/benchmarks/tree/master/endtoend

[32] J. Spanhel, J. Sochor, R. Juranek, A. Herout, L. Marsik, and P. Zemcik, "Holistic recognition of low quality license plates by CNN using track annotated data," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Lecce, Italy, Aug. 2017, pp. 1–6.

[33] *Darknet (AlexeyAB Version)*. Accessed: Jan. 14, 2020. [Online]. Available: github.com/AlexeyAB/darknet

[34] Z. Siyu, "An end-to-end license plate localization and recognition system," M.S. thesis, Dept. of Electrical Eng., Rochester Institute of Technology, New York, USA, 2015.

[35] X. Zhang, N. Gu, H. Ye, and C. Lin, "Vehicle license plate detection and recognition using deep neural networks and generative adversarial networks," *J. Electron. Imag.*, vol. 27, no. 04, Aug. 2018, 043056.

[36] R. Laroca, L. A. Zanlorensi, G. R. Gonçalves, E. Todt, W. Robson Schwartz, and D. Menotti, "An efficient and layout-independent automatic license plate recognition system based on the YOLO detector," 2019, *arXiv:1909.01754*. [Online]. Available: http://arxiv.org/abs/1909.01754

[37] Y. Wu and J. Li, "License plate recognition using deep FCN," in *Proc. Int. Conf. Cognit. Syst. Signal Process.*, Singapore, 2016, pp. 225–234.

[38] J. Yépez, R. D. Castro-Zunti, and S.-B. Ko, "Deep learning-based embedded license plate localisation system," *IET Intell. Transp. Syst.*, vol. 13, no. 10, pp. 1569–1578, Oct. 2019.

CHRIS HENRY received the B.E. degree in electronic engineering from Hamdard University, Pakistan, in 2015, and the M.E. degree in IT convergence engineering from Gachon University, South Korea, in 2019. His current research interests include object recognition, image matting, and deep learning.

SUNG YOON AHN is currently pursuing the B.S. degree in software engineering with Gachon University, Seongnam-si, South Korea. His research interests include computer vision and machine learning.

SANG-WOONG LEE received the B.S. degree in electronics and computer engineering and the M.S. and Ph.D. degrees in computer science and engineering from Korea University, Seoul, South Korea, in 1996, 2001, and 2006, respectively. From June 2006 to May 2007, he was a Visiting Scholar with the Robotics Institute, Carnegie Mellon University. From September 2007 to February 2017, he worked as a Professor with the Department of Computer Engineering, Chosun University, Gwangju, South Korea. He is currently a Professor with the Department of Software, Gachon University. His present research interests include face recognition, computational aesthetics, machine learning, and medical imaging analysis.