IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 May 2023, accepted 5 June 2023, date of publication 9 June 2023, date of current version 15 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3284973

RESEARCH ARTICLE

CNN-Based Two-Stage Parking Slot Detection Using Region-Specific Multi-Scale Feature Extraction

QUANG HUY BUI AND JAE KYU SUHR, (Member, IEEE)
Department of Intelligent Mechatronics Engineering, Sejong University, Seoul 05006, South Korea

Corresponding author: Jae Kyu Suhr (jksuhr@sejong.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2022R1F1A1074708), and in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2020R1A6A1A03038540).

ABSTRACT Although it is well-known that the two-stage approach outperforms the one-stage approach in general object detection, they have similarly performed in parking slot detection so far. We consider this is because the two-stage approach has not yet been adequately specialized for parking slot detection. Thus, this paper proposes a highly specialized two-stage parking slot detector that uses region-specific multi-scale feature extraction. In the first stage, the proposed method finds the entrance of the parking slot as a region proposal by estimating its center, length, and orientation. The second stage of this method designates specific regions that most contain the desired information and extracts features from them. That is, features for the location and orientation are separately extracted from only the specific regions that most contain the locational and orientational information. In addition, multi-resolution feature maps are utilized to increase both positioning and classification accuracies. A high-resolution feature map is used to extract detailed information (location and orientation), while another low-resolution feature map is used to extract semantic information (type and occupancy). In experiments, the proposed method was quantitatively evaluated with two large-scale public parking slot detection datasets: SNU and PS2.0 datasets. In SNU dataset, the proposed method achieved state-of-the-art performance with 95.75% recall and 95.78% precision.

INDEX TERMS Parking slot detection, deep learning, convolutional neural network (CNN), two-stage detector, around view monitor (AVM), automatic parking system.

I. INTRODUCTION

As a result of the growing interest in autonomous driving, autonomous parking systems have gained more attention. Such systems have proven their role by providing drivers convenience and reducing vehicle damage [1], [2], [3]. In autonomous parking, the first step is to precisely detect an available parking space. Recently, a soaring number of vehicles are equipped with vision systems that enhance the drivers' awareness of their surroundings. Some clear examples are the rearview camera and around view monitor (AVM) system, which eliminates the rear blind spot and provides 360 degrees observation around the vehicle, respectively. This tendency has led to the significant development of vision-based parking slot detection.

The initial methods for the vision-based parking slot detection are based on hand-crafted features. These methods extract line or corner features from images and combine them using geometric rules to find parking slots. Although they have shown noticeable performances, the inconvenience of designing adequate geometric rules and the fragility of those rules to various environmental conditions have been revealed as their significant drawbacks. In recent years, with the rise of deep learning, convolutional neural network (CNN) has made considerable breakthroughs in numerous object detection tasks. CNN-based general object detection methods

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

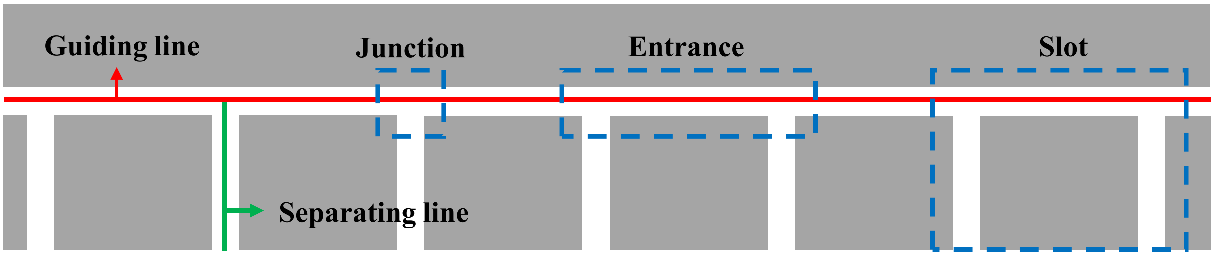


FIGURE 1. Terminologies for parking slot markings.

can be categorized into two main approaches: two-stage and one-stage. The two-stage approach consists of one step to generate region proposals and the other step to classify the objects inside those regions and refine their bounding boxes. Region-based CNN (RCNN) [4], Fast RCNN [5], Faster RCNN [6], RFCN [7], and Mask-RCNN [8] are representative methods for this approach. On the other hand, the one-stage approach directly acquires bounding boxes for the objects along with their classes without generating region proposals. You only look once (YOLO) [9], YOLOv2 [10], YOLOv3 [11], YOLOv4 [12], single shot multibox detector (SSD) [13], and RetinaNet [14] are representative methods for this approach. Through various applications, the two-stage approach has shown a high detection performance with a slow processing speed, while the one-stage approach has shown a moderate detection performance with a fast processing speed. Witnessing the success of CNN-based object detection, many research works have been conducted to utilize it for parking slot detection tasks.

Similar to general object detection, CNN-based parking slot detection methods can be categorized into two approaches: two (or multi)-stage and one-stage. In multi-stage parking slot detection methods, the first stage generates region proposals by finding two or four corners of the parking slots [15], [16] or by combining parts of the parking slots found by CNNs using geometrics rules [17], [18], [19], [20], [21]. Then, the following stages refine the positions or classify types and occupancies of the parking slots by extracting features of the region proposals from the corresponding regions of the feature map or input image. On the other hand, one-stage parking slot detection methods directly acquire all information of the parking slot such as location, orientation, type, and occupancy in a single step without generating region proposals [22], [23]. Even though the two-stage detection approach has been known to outperform the one-stage detection approach in general object detection tasks, their performances have been reported to be similar in parking slot detection tasks. The state-of-the-art one-stage parking slot detector has shown a slightly better performance than the two-stage parking slot detectors [22], [23]. We consider this is because the two-stage approach has not yet been adequately specialized for parking slot detection tasks.

Therefore, this paper proposes a highly specialized two-stage parking slot detector that uses region-specific multi-scale feature extraction. In the first stage, the proposed method finds the entrance of the parking slot as a region proposal by predicting its location, orientation, and length. It is unlike the previous methods that adopt an upright rectangle [15] or four corners of the parking slot [16] as a region proposal. In the second stage, this method uses a region-specific feature extraction method that extracts features only from the specific regions of the feature map that most contain the desired information. For instance, features for predicting the location and orientation of the parking slot are separately extracted from only the specific regions that most contain the corresponding information. This is possible because the parking slot is a planar rigid object on the ground plane and captured in an AVM image after removing perspective distortion. It is unlike the previous methods that extract the features of the entire region proposal from the feature map [16] or crop the whole area of the region proposal from the input image [15]. In addition, the proposed method utilizes multi-resolution feature maps to increase both positioning and classification accuracies. It uses a high-resolution feature map for extracting detailed information (location and orientation) and a low-resolution feature map for extracting semantic information (type and occupancy). Finally, from the extracted features, the proposed method refines the locations and orientations of the parking slots and classifies their types and occupancies. In experiments, the proposed method was quantitatively evaluated with two large-scale public parking slot detection datasets and outperformed previous methods, including both one-stage and two-stage approaches. The contributions of this paper can be summarized as follows:

- It suggests an effective way to apply the two-stage general object detection to the parking slot detection tasks.
- It proposes a region-specific multi-scale feature extraction that increases both detection performance and positioning accuracy by effectively extracting the precise information of the parking slot from the region proposal.
- It presents quantitative evaluation results using two large-scale public datasets and shows that the proposed method gives a state-of-the-art performance.

II. RELATED WORKS

Previous vision-based parking slot detection methods can be categorized into hand-crafted feature-based and deep learning-based (or CNN-based). Since these methods exploit parking slot markings on the ground, terminologies for the parking slot markings are briefly introduced in Fig. 1. In this figure, the guiding line segregates the parking slots from the roadway, and separating lines divide individual parking slots. Junctions are the intersections of the guiding line and separating lines, and the entrance of a parking slot is the segment between two adjacent junctions. A parking slot is formed by the entrance and a pair of separating lines connecting to it.

Hand-crafted feature-based methods detect parking slots by extracting manually designed features of the parking slot and combining them using traditional rule-based techniques. Since this paper concentrates mainly on the deep learning-based methods, the hand-crafted feature-based methods are briefly introduced. Based on the type of extracted features, the hand-crafted feature-based methods can be categorized into line-based and junction-based. The line-based methods first find the guiding lines and separating lines and then group them to generate parking slots. Various techniques have been employed for detecting and combining line features. For line detection, Hough transform [24], [25], Radon transform [26], [27], or random sample consensus (RANSAC) algorithm [28], [29], [30], [31] have been utilized. For line combination, K-means clustering [26], grouping based on predetermined distances and parallel and perpendicular properties [25], [31], [32], [33] have been used. Different from the line-based methods, the junction-based methods first find junctions of the parking slots and then pair them to generate parking slot candidates. For junction detection, Harris corner detector [34], [35], [36] and Viola-Jones detector [37] have been applied. The detected junctions are paired by various geometric rules based on their types, locations, and orientations. Once parking slots are detected by the line-based or junction-based methods, their occupancies are then classified. To this end, difference-of-Gaussians-based histogram with linear discriminant analysis (LDA) classifier [38], Canny edges with naïve Bayes classifier [32], color histogram with support vector machine (SVM) classifier [27], ultrasonic sensor-based occupancy grid [33], [36] have been exploited.

As CNN-based object detection has shown significant results in recent years, various research works have been done to apply this technique to the parking slot detection task. CNN-based parking slot detection methods can be categorized into two approaches: multi-stage and one-stage. The first multi-stage parking slot detection method applying deep learning technique was proposed by Zhang et al. [17]. The first stage of this method finds junctions using YOLOv2 and its second stage generates parking slot candidates by combining the junctions using geometric rules. Finally, a CNN-based classifier verifies the candidates whose orientations are determined by a template matching technique in the last stage. Similarly, Huang et al. [18] customized a CNN to find locations, orientations, and types of junctions and then grouped them using geometric rules to generate parking slot candidates. The method proposed by Li et al. [15] detects junctions and entrances using YOLOv3 with upright bounding boxes and finds parking slots by means of geometric rules and relation between the detected junctions and entrances in the first stage. Its second stage separately crops the regions of the parking slots from the input image and forwards them to an additional CNN for occupancy classification. From a different approach, Jang and Sunwoo [19] and Jiang et al. [20] proposed methods that extract the marking lines and junctions of parking slots using semantic segmentation techniques in the first stage. They generate parking slots using extracted lines and junctions along with geometric rules and classify their occupancies based on the semantic segmentation results in the second stage. All aforementioned methods have shown the potential of deep learning techniques in parking slot detection tasks. However, they cannot be trained end-to-end due to the manual selection of geometric rules and associated parameters, which is inconvenient and complicated to set. To overcome this limitation and benefit the training process, end-to-end trainable methods have been proposed. Zinelli et al. [16] presented the first end-to-end trainable parking slot detection method utilizing anchor-free faster R-CNN [39]. The first stage of this method roughly estimates four corners of the parking slot as a region proposal. RoIAlign [8] is then used to extract features from the proposed region for location refinement and occupancy classification in the second stage. Trying to apply a general object detection to the parking slot detection task, this method, however, shows clear limitations of detection performance and positioning accuracy because it uses the general object detector without sufficient modification. Another end-to-end trainable two-stage parking slot detection method was proposed by Do and Choi [40]. In the first stage of this method, the context recognizer predicts the common type and orientation of all parking slots in the input image. Then, the parking slot detector estimates the exact positions of the parking slots using rotated anchor boxes in the second stage. Although this method can obtain all information of the parking slots, including location, orientation, type, and occupancy, it handles only the cases where all parking slots in the input images have the same type and orientation and requires a high computational cost due to the use of two separate backbone networks. Min et al. [21] proposed a three-stage parking slot detection method. It finds junctions and extracts their features in the first stage and aggregates the junctions to generate parking slot candidates using an attentional graph neural network in the second stage. Finally, those candidates are verified based on a multilayer perceptron in the last stage. This method is limited in dealing with slanted parking slots due to the absence of orientation information extraction.

Since multi-stage parking slot detection methods, in general, are mediocre in terms of inference speed, one-stage parking slot detection methods have also been suggested. Li et al. [22] introduced a one-stage parking slot detection method focusing on locating the entrance of the parking slot.

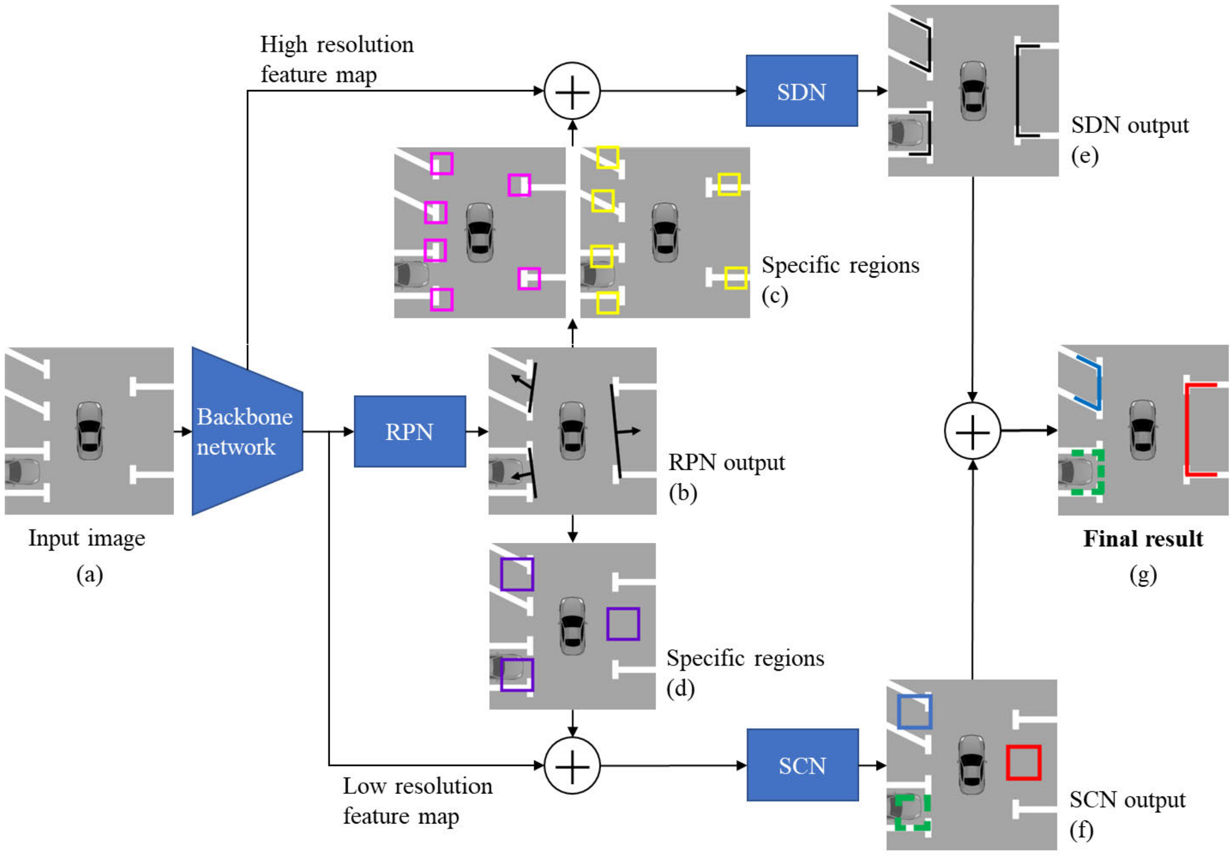


FIGURE 2. Overall architecture of the two-stage method utilizing region-specific multi-scale feature extraction.

This method predicts the location, orientation, and type of the parking slot entrance using a customized CNN. Although it shows a fast inference speed with an adequate detection performance, it provides no occupancy information and unsatisfactory orientation accuracy due to the predefined orientations for slanted parking slots. Suhr and Jung [23] suggested another one-stage parking slot detection method. This method simultaneously extracts global information (rough location, type, and occupancy of the parking slot) and local information (precise location and orientation of junctions) and combines them to provide final parking slots. This method achieves a high detection performance requiring only a low computational cost while providing all information of the parking slot (location, orientation, type, and occupancy).

As a thorough literature review, it is observed that currently, for parking slot detection tasks, one-stage detection methods slightly outperform multi-stage detection methods in both aspects: detection performance and positioning accuracy. This is unlike general object detection tasks, where the two-stage approach outperforms the one-stage approach. We consider one of the main reasons is that the two-stage approach has not yet been adequately specialized for parking slot detection tasks. Therefore, this paper proposes a highly specialized two-stage parking slot detector. In experiments, it has been revealed that the adequately designed two-stage parking slot detection method outperforms the one-stage parking slot detection methods.

III. PROPOSED METHOD

A. OVERALL ARCHITECTURE

This paper proposes a novel two-stage parking slot detection method using region-specific multi-scale feature extraction. The proposed method roughly locates parking slot entrances using the region proposal network (RPN) in the first stage and precisely estimates positions and properties of parking slots using the slot detection network (SDN) and slot classification network (SCN) in the second stage. Fig. 2 illustrates the overall architecture of the proposed method. An input AVM image, as in Fig. 2(a), is inserted into the backbone network for feature maps extraction. This paper tried several backbone networks and selected DenseNet121 [41], whose performance has been proven in various applications. After acquiring the feature maps, the RPN with one convolutional layer is applied to the low-resolution feature map to generate rough positions of parking slot entrances as region proposals. Fig. 2(b) shows the output of the RPN, where solid black lines and arrows indicate the entrances and orientations of the parking slots, respectively. Once region proposals are generated, this paper applies the region-specific multi-scale feature extraction to estimate the positions and properties of parking slots more accurately. Rather than utilizing features of the entire parking slot, the region-specific approach extracts features from only the regions that most contain the desired information. Magenta and yellow squares in Fig. 2(c) are the specific regions used to extract features for estimating the locations and orientations of the parking slot, respectively. These regions include junctions and separating lines, thus containing rich locational and orientational information. Purple squares in Fig. 2(d) are the specific regions used to extract features for type and occupancy classification. These regions include the center areas of parking slots that contain overall shape and texture information. In addition, multi-resolution feature maps are utilized to enhance positioning and classification performances. The high-resolution feature map, containing more detailed information, is used to extract features for estimating the locations and orientations of parking slots, while the low-resolution feature map, containing more semantic information, is used to extract features for classifying their types and occupancies. After obtaining the features using the proposed region-specific multi-scale feature extraction, the SDN with a set of fully connected layers is applied to estimate precise positions of the parking slots, as marked with black lines in Fig. 2(e). Concurrently, the SCN with a set of fully connected layers is applied to estimate types and occupancies of the parking slots. Fig. 2(f) shows the output of the SCN where blue solid, red solid, and green dashed rectangles indicate vacant slanted, vacant parallel, and occupied perpendicular parking slots, respectively. The proposed method determines the final parking slots by combining their positions, types, and occupancies, as illustrated in Fig. 2(g).

B. REGION PROPOSAL NETWORK

The proposed method generates the parking slot entrance as a region proposal, unlike previous methods that capture the whole parking slot using a parallelogram [15], quadrilateral [16], or rotated rectangle [40]. The reason for this selection is that AVM images do not usually include the whole parking slot, and the parking slot entrance itself contains enough information for vehicles to start parking. Additionally, the proposed method differs from the methods in [17], [18], and [20], which depend on hand-crafted geometric rules to find parking slots. The proposed method geometrically models a parking slot, but all parameters used for the parking slot model are predicted by the network. This allows the method to be end-to-end trainable. To represent the parking slot entrance, this paper considered two approaches suggested by Li et al. [22] and Suhr and Jung [23]. The former uses the location and orientation of the entrance center, and the latter uses the locations of the junction pair. Based on the experimental comparison, this paper modifies the approach suggested by Li et al. [22] and represents the parking slot entrance by its center location (x, y), orientation $(\cos\theta_e, \sin\theta_e)$, length (l), and the orientation of the parking slot $(\cos\theta_s, \sin\theta_s)$ as shown in Fig. 3.

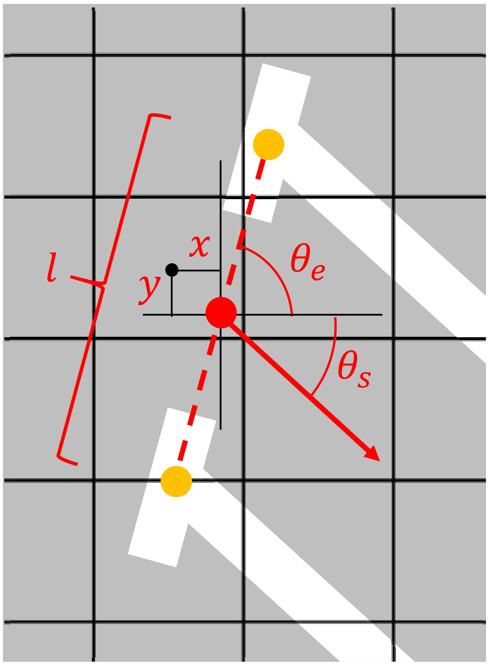


FIGURE 3. Representation of the parking slot entrance using its center location (x, y), orientation $(\cos\theta_e, \sin\theta_e)$, length (l), and parking slot orientation $(\cos\theta_s, \sin\theta_s)$.

Fig. 4 gives a detailed description of the RPN. In the RPN, one convolutional layer with eight 3 × 3 filters is applied to the low-resolution feature map produced by the backbone network, as illustrated at the top of Fig. 4. The spatial dimension of the RPN output is $h \times w$. This means that the input image is divided into a grid of $h \times w$ cells. Since one cell is responsible for at most one parking slot entrance, the cell size should be set smaller than the minimum size of the parking slots. In Fig. 4, the illustrations are intentionally depicted with a grid of 6 × 6 cells for ease of understanding. At the top of Fig. 4(a), the possibility that a cell contains any entrance center is estimated using one 3×3 filter followed by the sigmoid function. At the bottom of Fig. 4(a), green cells indicate the cells with high possibilities to contain entrance centers of parking slots. At the top of Fig. 4(b), the relative position from a cell center to an entrance center is calculated using two 3 × 3 filters followed by the sigmoid function. At the bottom of Fig. 4(b), blue arrows indicate 2D vectors connecting the cell centers to the entrance centers contained in corresponding cells. In this figure, only the results obtained from the cells containing the entrance centers are drawn. At the top of Fig. 4(c), orientations of the entrances are obtained using two 3 × 3 filters followed by the tanh function. Because the unit vector representing the orientation consists of values in the range of $[-1.0, 1.0]$, the tanh function is used. At the bottom of Fig. 4(c), magenta arrows indicate 2D vectors that represent the orientations of the entrances whose centers are contained in corresponding cells. At the top of Fig. 4(d), lengths of the entrances are estimated using one 3 × 3 filter followed by the sigmoid

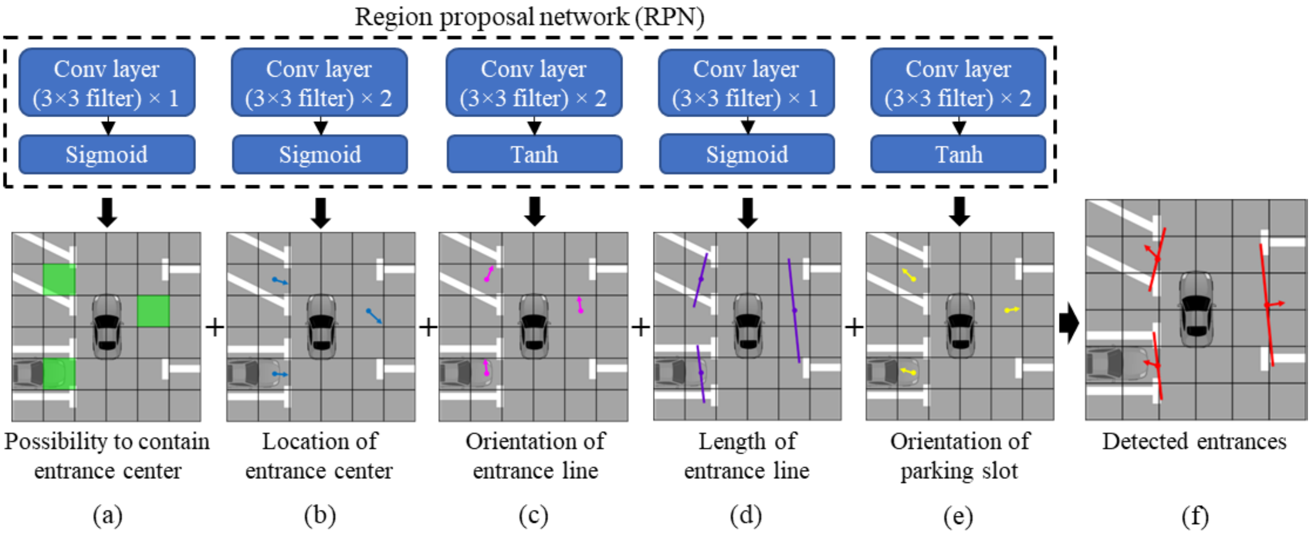

Region proposal network (RPN)
Conv layer (3×3 filter) × 1
Sigmoid
Conv layer (3×3 filter) × 2
Sigmoid
Conv layer (3×3 filter) × 2
Tanh
Conv layer (3×3 filter) × 1
Sigmoid
Conv layer (3×3 filter) × 2
Tanh
Possibility to contain entrance center
(a)
Location of entrance center
(b)
Orientation of entrance line
(c)
Length of entrance line
(d)
Orientation of parking slot
(e)
Detected entrances
(f)

FIGURE 4. Region proposal network (RPN) and the detailed information obtained from it.

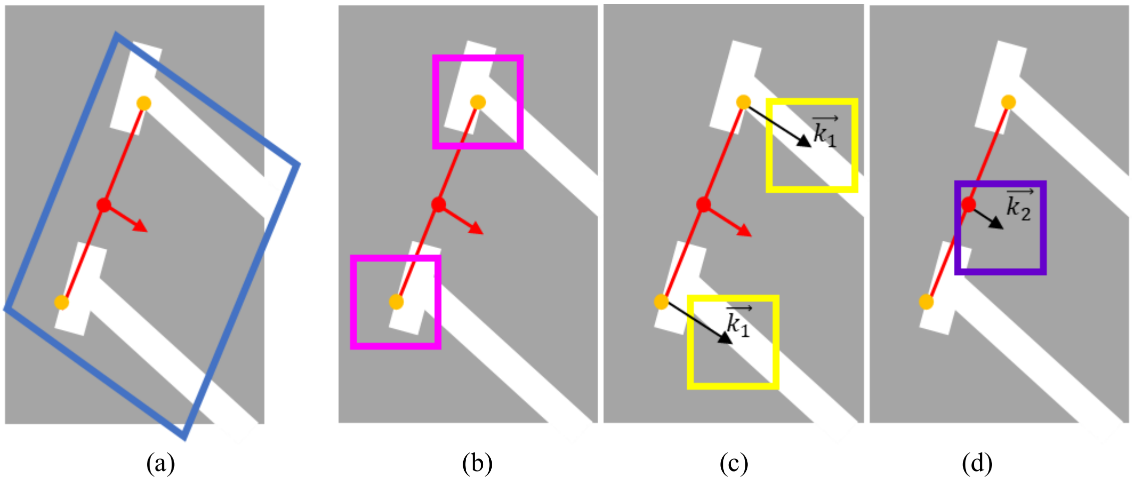


FIGURE 5. (a) Parallelogram-based ROI designation; (b)-(d) Region-specific ROI designation, (b) shows location regions, (c) shows orientation regions, (d) shows type and occupancy region.

function. At the bottom of Fig. 4(d), purple lines indicate the estimated lengths of the entrances. At the top of Fig. 4(e), orientations of the parking slot are calculated using two 3 × 3 filters followed by the tanh function. At the bottom of Fig. 4(e), yellow arrows indicate 2D vectors that represent the orientations of the parking slots whose entrance centers are contained in corresponding cells. Fig. 4(f) illustrates the output of the RPN obtained by combining all the information shown in Fig. 4(a)-(e). Solid red lines and arrows indicate the generated parking slot entrances and the orientations of the parking slots, respectively. Because the RPN can find multiple entrances for a single parking slot, non-maximum suppression (NMS) is utilized to remove duplicate detections based on the fact that two parking slots cannot overlap. Two entrances are considered as duplicates if their centers are closely located.

C. REGION-SPECIFIC MULTI-SCALE FEATURE EXTRACTION

After generating the parking slot entrance as a region proposal, the proposed method extracts features from the region of interest (ROI) specified by the generated region proposal. General object detection methods use upright rectangles as ROIs for feature extraction [42], [43]. Still, upright rectangles are inappropriate for parking slot detection because parking slots can appear with arbitrary orientations in AVM images. To tackle this problem, previous parking slot detection methods suggested other ways to designate ROIs for feature extraction, such as using parallelograms [15] or quadrilaterals [16]. Fig. 5(a) shows a parallelogram-based ROI designation. In this figure, a blue parallelogram, inferred from the parking slot entrance, indicates the ROI for feature extraction. Since this ROI contains the whole parking slot, the features extracted from this region can predict all the

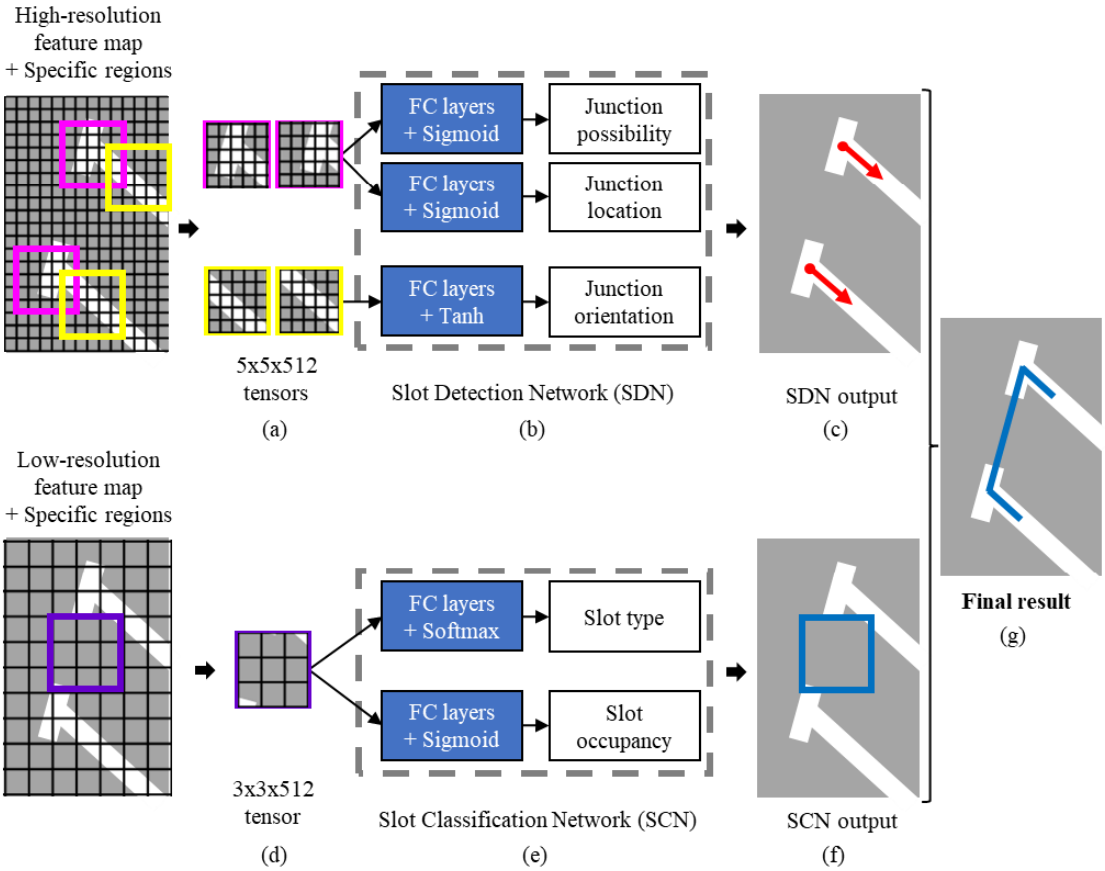


FIGURE 6. Slot detection network (SDN) and slot classification network (SCN), and the detailed information obtained from them.

information, including location, orientation, type, and occupancy. However, this approach is not optimal to designate the ROI for feature extraction in parking slot detection because specific regions of the parking slot contain features for specific information. For instance, features including locational and orientational information are mostly found in regions around junctions and separating lines, respectively. Because of this characteristic, if features are extracted from the whole parking slot region, the network can have difficulty finding where to focus on. Our experiment has revealed that the approach using the whole region degrades the detection performance.

Therefore, to overcome the disadvantage coming from using features of the whole parking slot and enhance the detection performance, this paper proposes a region-specific ROI designation using multi-scale feature maps, called region-specific multi-scale feature extraction. The region-specific ROI designation is illustrated in Fig. 5(b)-(d). The proposed method defines only the specific regions that most contain the desired information as ROIs for feature extraction. This is possible because the parking slot is a planar rigid object on the ground plane and captured in an AVM image after removing perspective distortion, so its components, such as junctions and separating lines, can roughly be localized based on the parking slot entrance generated by the RPN. Magenta squares in Fig. 5(b) are the designated ROIs to extract features for precise location prediction. Regions around two junctions are chosen as ROIs because they contain most of the locational information. In this figure, a red line and arrow indicate the parking slot entrance generated by the RPN, and both ends of the red line are rough locations of two junctions. Yellow squares in Fig. 5(c) are the designated ROIs to extract features for precise orientation prediction. Regions around two separating lines are chosen as ROIs because they contain most of the orientational information. A purple square in Fig. 5(d) is the designated ROI to extract features for type and occupancy classification. The central region of the parking slot is used for this ROI because it contains information about the overall properties of the parking slot. The location of the ROIs in Figs. 5(c) and (d) are determined by two vectors, $\vec{k}_1$ and $\vec{k}_2$, whose directions are set to the orientation of the parking slot (red arrow), and lengths are empirically set to 50 and 32 pixels, respectively. ROIs generated by the proposed method are all upright squares. Rotated rectangles have been tried, but they did not improve the performance while increasing computational cost. Furthermore, to reduce the volume and computation of the network, this method does not crop the regions from the input image but the regions from the feature maps. This means that both its first and second stages share the backbone network, unlike some of the previous methods that crop the regions from the input image and use additional backbone networks to extract features for the second stage [15], [17].

In addition to the region-specific ROI designation, this paper suggests extracting features in different scales according to types of information. The proposed method extracts features for predicting the location and orientation from the high-resolution feature map that keeps more detailed information. On the other hand, features for predicting the type and occupancy are extracted from the low-resolution feature map that contains more semantic information. Experimental results have shown that the use of the region-specific multi-scale feature extraction remarkably increases the detection performance as well as the positioning accuracy.

D. PARKING SLOT DETECTION AND CLASSIFICATION NETWORKS

Utilizing the features obtained by the proposed region-specific multi-scale feature extraction, the SDN detects the precise locations and orientations of the parking slots while the SCN classifies their types and occupancies. The top and bottom parts of Fig. 6 give detailed descriptions of the SDN and SCN, respectively. As illustrated in Fig. 6(a), for every parking slot, the region-specific multi-scale feature extractor extracts four 5 × 5 × 512 tensors from the high-resolution feature map, in which two tensors are from the two junctions (in magenta squares), and the other two are from the two separating lines (in yellow squares). The SDN uses those tensors as inputs after flattening them. Fig. 6(b) shows the architecture of the SDN. Using the tensor from one magenta square, the SDN predicts three values: one for the possibility that this region contains a junction and two for the relative location from the region center to the junction. For this, two sets of fully connected layers followed by the sigmoid function are utilized. This process is separately applied to the tensors from the two magenta squares. The SDN also predicts a unit vector that describes the orientation of the separating line using the tensor from a yellow square. For this, one set of fully connected layers followed by the tanh function is utilized. This process is separately applied to the tensors from the two yellow squares. Fig. 6(c) gives a visual representation for the output of the SDN, where the red dots and arrows indicate the precisely predicted locations of the junctions and orientations of the separating lines, respectively. Similarly, as shown in Fig. 6(d), the SCN uses one 3 × 3 × 512 tensor extracted from the purple square of the low-resolution feature map as an input after flattening it. From this tensor, the SCN predicts four values: one for occupancy (vacant or occupied) and three for the parking slot type (perpendicular, parallel, or slanted). For this, one set of fully connected layers followed by the sigmoid function and another set of fully connected layers followed by the softmax function are utilized, as presented in Fig. 6(e). Fig. 6(f) gives a visual representation for the output of the SCN, where the blue color and solid line indicate slanted and vacant properties, respectively. The final parking slot detection result is obtained by combining the outputs of the SDN and SCD as shown in Fig. 6(g).

E. LOSSES

1) LOSSES FOR THE FIRST STAGE

The loss for the first stage (RPN), $loss_{first}$ is a weighted sum of five losses corresponding to five information that represents the parking slot entrance as

$$loss_{first} = w_{ep}loss_{ep} + w_{exy}loss_{exy} + w_{el}loss_{el} + w_{eo}loss_{eo} + w_{so}loss_{so} \quad (1)$$

where w_{ep}, w_{exy}, w_{el}, w_{eo}, and w_{so} are the weights for the five losses and experimentally set. Each loss will be described in detail one by one.

The loss for the possibility that a grid cell contains an entrance center, $loss_{ep}$ is calculated as

$$loss_{ep} = \sum_{i=1}^{h\times w}\left[I_e^i\left(ep_{pred}^i - ep_{true}^i\right)^2 + \lambda_e\left(1 - I_e^i\right)\left(ep_{pred}^i - ep_{true}^i\right)^2\right] \quad (2)$$

where ep_{true}^i is the ground truth for the possibility that the i-th cell includes any parking slot entrance center. This value is 1 if it includes or 0 if it does not. The input image is assumed to be divided into a grid of $h \times w$ cells. ep_{pred}^i is the prediction of the network for ep_{true}^i. I_e^i indicates whether the i-th cell includes any entrance center and is set to 1 if it includes or 0 if it does not. Because the number of cells that contain the entrance center is much smaller than the number of cells that do not, λ_e is multiplied to compensate for this imbalance. It is set based on the ratio of those numbers in the training dataset.

The loss for the relative location from the cell center to the entrance center included in that cell, $loss_{exy}$ is calculated as

$$loss_{exy} = \sum_{i=1}^{h\times w} I_e^i\left[\left\{\left(ex_{pred}^i - 0.5\right) - \frac{ex_{true}^i}{W_{cell}}\right\}^2 + \left\{\left(ey_{pred}^i - 0.5\right) - \frac{ey_{true}^i}{H_{cell}}\right\}^2\right] \quad (3)$$

where $\left(ex_{true}^i, ey_{true}^i\right)$ is the ground truth for the relative location from the center of the i-th cell to the entrance center included in it. These values are divided by W_{cell} and H_{cell} to be normalized to the range of $[-0.5, 0.5]$. W_{cell} and H_{cell} are the width and height of the region corresponding to a single cell of the low-resolution feature map in the original image, respectively. They are 32 pixels because the backbone network includes four 2 × 2 pooling layers whose strides are 2. $\left(ex_{pred}^i, ey_{pred}^i\right)$ is the prediction of the network for $\left(ex_{true}^i, ey_{true}^i\right)$. Because of the sigmoid function, the predicted values are in the range of $[0, 1]$, so we subtract 0.5 from them to match their ranges with the ground truth values.

The loss for the entrance length, $loss_{el}$ is calculated as

$$loss_{el} = \sum_{i=1}^{h\times w} I_e^i \left[el_{pred}^i - \frac{el_{true}^i}{L_{max}} \right]^2 \tag{4}$$

where el_{true}^i is the ground truth for the entrance length. It is divided by L_{max} to be normalized to the range of [0, 1]. L_{max} is the maximum length of the parking slot entrance and is set based on the training dataset. el_{pred}^i is the prediction of the network for el_{true}^i.

The loss for the orientation of the parking slot entrance, $loss_{eo}$ is calculated as

$$loss_{eo} = \sum_{i=1}^{h\times w} I_e^i \left[\left(eox_{pred}^i - eox_{true}^i \right)^2 + \left(eoy_{pred}^i - eoy_{true}^i \right)^2 \right] \tag{5}$$

where $\left(eox_{true}^i, eoy_{true}^i\right)$ is a unit vector representing the ground truth for the orientation of the entrance whose center is included in the i-th cell. $\left(eox_{pred}^i, eoy_{pred}^i\right)$ is the prediction of the network for $\left(eox_{true}^i, eoy_{true}^i\right)$. These values are in the range of [−1, 1] because of the tanh activation function.

The loss for the orientation of the parking slot, $loss_{so}$ is calculated as

$$loss_{so} = \sum_{i=1}^{h\times w} I_e^i \left[\left(sox_{pred}^i - sox_{true}^i \right)^2 + \left(soy_{pred}^i - soy_{true}^i \right)^2 \right] \tag{6}$$

where $\left(sox_{true}^i, soy_{true}^i\right)$ is a unit vector representing the ground truth for the orientation of the parking slot whose entrance center is included in the i-th cell. $\left(sox_{pred}^i, soy_{pred}^i\right)$ is the prediction of the network for $\left(sox_{true}^i, soy_{true}^i\right)$. These values are in the range of [−1, 1] because of the tanh activation function.

2) LOSSES FOR THE SECOND STAGE

The loss for the second stage, $loss_{second}$ is the sum of the loss for the SDN ($loss_{SDN}$) and the loss for the SCN ($loss_{SCN}$) as

$$loss_{second} = loss_{SDN} + loss_{SCN} \tag{7}$$

The loss for the SDN is a weighted sum of three losses corresponding to three information that represents the junction of the parking slot as

$$loss_{SDN} = w_{jp} loss_{jp} + w_{jxy} loss_{jxy} + w_{jo} loss_{jo} \tag{8}$$

where w_{jp}, w_{jxy}, and w_{jo} are the weights for the three losses and experimentally set.

The loss for the possibility that the magenta ROIs in Fig. 6(a) include junctions, $loss_{jp}$ is calculated as

$$loss_{jp} = \sum_{i=1}^{R} \left[jp_{pred}^i - jp_{true}^i \right]^2 \tag{9}$$

where jp_{true}^i is the ground truth for the possibility that the i-th ROI contains a junction. R is the number of ROIs contained in an input image. jp_{pred}^i is the prediction of the network for jp_{true}^i. This value is in the range of [0, 1] because of the sigmoid activation function.

The loss for the relative location from the center of the magenta ROI in Fig. 6(a) to the junction included in that ROI, $loss_{jxy}$ is calculated as

$$loss_{jxy} = \sum_{i=1}^{R} I_j^i \left[\left\{ \left(jx_{pred}^i - 0.5 \right) - \frac{jx_{true}^i}{W_{ROI}} \right\}^2 + \left\{ \left(jy_{pred}^i - 0.5 \right) - \frac{jy_{true}^i}{H_{ROI}} \right\}^2 \right] \tag{10}$$

where $\left(jx_{true}^i, jy_{true}^i\right)$ is the ground truth for the relative location from the center of the i-th ROI to the junction included in it. These values are divided by W_{ROI} and H_{ROI} to be normalized to the range of [−0.5, 0.5]. W_{ROI} and H_{ROI} are the width and height of the region corresponding to a 5 × 5 area of the high-resolution feature map in the original image. They are 80 pixels because the high-resolution feature map is taken from the third pooling layer of the backbone network. I_j^i indicates whether the i-th ROI contains any junction and is set to 1 if it contains or 0 if it does not. $\left(jx_{pred}^i, jy_{pred}^i\right)$ is the prediction of the network for $\left(jx_{true}^i, jy_{true}^i\right)$. Because of the sigmoid function, the predicted values are in the range of [0, 1], so we subtract 0.5 from them to match their ranges with the ground truth values.

The loss for the orientation of the separating lines in the yellow ROIs of Fig. 6(a), $loss_{jo}$ is calculated as

$$loss_{jo} = \sum_{i=1}^{R} I_j^i \left[\left(jox_{pred}^i - jox_{true}^i \right)^2 + \left(joy_{pred}^i - joy_{true}^i \right)^2 \right] \tag{11}$$

where $\left(jox_{true}^i, joy_{true}^i\right)$ is a unit vector representing the ground truth for the orientation of the separating line included in the i-th ROI. $\left(jox_{pred}^i, joy_{pred}^i\right)$ is the prediction of the network for $\left(jox_{true}^i, joy_{true}^i\right)$. These values are in the range of [−1, 1] because of the tanh activation function.

The loss for the SCN is a weighted sum of two losses corresponding to the type and occupancy of the parking slot as

$$loss_{SCN} = w_{st} loss_{st} + w_{socc} loss_{socc} \tag{12}$$

where w_{st}, and w_{socc} are the weights for the two losses and experimentally set.

The loss for the type of the parking slot that contains the center of the purple ROI in Fig. 6(d), $loss_{st}$ is calculated based on the categorical cross-entropy as

$$loss_{st} = \sum_{i=1}^{R/2} I_{slot}^i \left[- \sum_{c=1}^{3} \left\{ \lambda_{st,c} st_{true,c}^i \log \left(st_{pred,c}^i \right) \right\} \right] \tag{13}$$

where $st^i_{true,c}$ is the ground truth for the probability that the type of the parking slot containing the center of the i-th ROI is c. st^i_{true} is represented in one-hot encoding and c has a value of 1, 2, or 3. So $\left(st^i_{true,1}, st^i_{true,2}, st^i_{true,3}\right)$ for the perpendicular, parallel, or slanted type is set to (1, 0, 0), (0, 1, 0), or (0, 0, 1), respectively. The number of ROIs for the SCN is $R/2$ when there are R ROIs for the SDN because one region proposal contains one purple ROI and two magenta and yellow ROIs. I^i_{slot} indicates whether the i-th ROI is included in a parking slot or not. Its value is set to 1 if included or 0 if not. $st^i_{pred,c}$ is the prediction of the network for $st^i_{true,c}$. $\lambda_{st,c}$ is the parameter that compensates for the imbalance of the numbers of different types of parking slots and is set based on the ratio of those numbers in the training dataset.

The loss for the occupancy of the parking slot that contains the center of the purple ROI in Fig. 6(d), $loss_{socc}$ is calculated as

$$loss_{socc} = \sum_{i=1}^{R/2} \Big[I^i_{occ} \left(socc^i_{pred} - socc^i_{true} \right)^2 + \lambda_{vac} I^i_{vac} \left(socc^i_{pred} - socc^i_{true} \right)^2 \Big] \quad (14)$$

where $socc^i_{true}$ is the ground truth for the occupancy of the parking slot containing the center of the i-th ROI. This value is 1 if occupied or 0 if vacant. $socc^i_{pred}$ is the prediction of the network for $socc^i_{true}$. I^i_{occ} indicates whether the center of the i-th ROI is included in an occupied parking slot and is set to 1 if included or 0 if not. I^i_{vac} indicates whether the center of the i-th ROI is included in a vacant parking slot and is set to 1 if included or 0 if not. λ_{vac} is the parameter that compensates for the imbalance of the numbers of occupied and vacant parking slots. This value is set based on the ratio of those numbers in the training dataset.

IV. EXPERIMENTS

A. DATASET

The proposed method was quantitatively evaluated using two datasets: Seoul National University dataset [40] and Tongji Parking Slot Dataset 2.0 [17]. This paper will call them the SNU dataset and PS2.0 dataset, respectively. These two datasets are the only publicly available large-scale datasets currently. This is because creating a dataset specifically for parking slot detection in AVM images is particularly challenging. It requires an AVM system with multiple fisheye cameras installed on an actual vehicle to capture synchronized images under various lighting conditions while driving.

Table 1 shows the summary of the two datasets. The SNU dataset consists of half AVM images obtained by two fisheye cameras attached to both side-view mirrors. This dataset includes 22817 images (18299 for training and 4518 for test) taken in 571 parking situations, and the image resolution is 768 × 256 pixels that correspond to 14.4 × 4.8 meters. Its labels contain locations, orientations, types, and occupancies of the parking slots. On the other hand, the PS2.0 dataset consists of full AVM images obtained by four fisheye cameras of the AVM system. It includes 12165 images (9827 for training and 2338 for test) taken in 166 parking situations, and the image resolution is 600 × 600 pixels that correspond to 10.0×10.0 meters. Its labels contain only locations and orientations of the parking slots, so we manually designated their types and occupancies. The two datasets include three types of parking slots (perpendicular, parallel, and slanted) taken indoors and outdoors in daytime and nighttime under sunny and rainy weather conditions. While the PS2.0 dataset is commonly utilized in parking slot detection-related research, it exhibits a possibility of overfitting due to the substantial similarity between the parking situations included in its training and test sets. To this end, the SNU dataset is more challenging because it contains various parking situations, and the images included in its training and test sets were taken from different parking situations.

TABLE 1. SUMMARY of PS2.0 and SNU datasets.

		SNU dataset	PS2.0 dataset
Parking situations		571	166
Image resolution (pixels)		768×256	600×600
Corresponding area (m)		14.4×4.8	10.0×10.0
No. of images	Training	18299	9827
	Test	4518	2338
	Total	22817	12165
No. of slots in train set	Perpendicular	39743	5668
	Parallel	5867	3492
	Slanted	3276	316
	Total	48886	9476
No. of slots in test set	Perpendicular	888	1151
	Parallel	11653	936
	Slanted	1004	81
	Total	13545	2168

TABLE 2. Hyperparameters used to calculate losses.

Parameter			SNU dataset	PS2.0 dataset
$loss_{first}$		$w_{ep}, w_{exy}, w_{el}, w_{eo}, w_{so}$	400, 400, 1000, 1000, 400	500, 400, 1000, 1500, 500
		λ_e	0.03	0.01
		L_{max}	400	291
$loss_{second}$	$loss_{SDN}$	w_{jp}, w_{jxy}, w_{jo}	1500, 2000, 6000	1000, 3000, 4000
		R	12	8
	$loss_{SCN}$	w_{st}, w_{socc}	0.5, 100	0.5, 100
		$\lambda_{st,1}, \lambda_{st,2}, \lambda_{st,3}$	8.33, 1.23, 14.92	1.76, 2.86, 31.65
		λ_{vac}	0.74	0.47

B. EXPERIMENTAL SETTING

The input images were resized to 576 × 192 pixels and 416 × 416 pixels for the SNU and PS2.0 datasets, respectively. The backbone network was initialized by the weights pre-trained on ImageNet, and the RPN, SDN, and SCN were

TABLE 3. Detection performances of the proposed method with different backbone networks on the SNU dataset.

Backbone network	Recall	Precision
VGG16	91.28%	91.78%
ResNet50	94.42%	94.19%
DenseNet121	**95.75%**	**95.78%**

TABLE 4. Comparison of parking slot detection performances on the SNU dataset.

Method	Loose criteria (12 pixels, 10 degrees)		Tight criteria (6 pixels, 5 degrees)	
	Recall	Precision	Recall	Precision
Proposed method (Two-stage)	**95.75%**	**95.78%**	**83.14%**	**83.16%**
Method in [23] (One-stage)	92.00%	92.00%	72.37%	72.37%
Method in [40] (Two-stage)	91.47%	90.88%	70.67%	70.21%

initialized by Xavier uniform initializer. The proposed network was trained for 80 epochs with a batch size of 32. In the first 60 epochs, the first stage (RPN) and the second stage (SDN and SCN) were trained alternately for one epoch each, and in the rest 20 epochs, both stages were trained simultaneously. The proposed network was optimized by Adam optimizer whose learning rate, β_1, β_2, and ϵ were set to 10^{-4}, 0.9, 0.999, and 10^{-8}, respectively. Hyperparameters used to calculate losses are presented in Table 2. All the experiments were conducted using TensorFlow and Nvidia GeForce RTX 3090 GPU.

For proper evaluation and comparison, this paper utilizes the criteria suggested by Zhang et al. [17], which is most widely used in previous parking slot detection papers. According to the criteria, a detected parking slot is considered as a true positive if the locations of its two junctions are within M pixels from the ground truth and their orientations are within N degrees from the ground truth. Otherwise, it is considered a false positive. For M and N, Zhang et al. [17] used 12 pixels and 10 degrees (loose criteria), but this paper additionally uses 6 pixels and 5 degrees (tight criteria) for more detailed comparisons. Recall and precision are calculated as

$$Recall = \frac{\#TruePositive}{\#GroundTruth} \tag{15}$$

$$Precision = \frac{\#TruePositive}{\#TruePositive + \#FalsePositive} \tag{16}$$

C. PERFORMANCE ON THE SNU DATASET

This paper has considered several backbone networks and selected DenseNet121. Table 3 shows the detection performance of the proposed method with three different backbone networks: VGG16 [44], ResNet50 [45], and DenseNet121 [41]. Since DenseNet121 outperforms the others, we utilize it to obtain the experimental results of the proposed method in the rest of this paper.

TABLE 5. Comparison of parking slot positioning errors on the SNU dataset.

Method	Location error (pixel/cm)		Orientation error (degree)	
	Mean	Std	Mean	Std
Proposed method (Two-stage)	**2.12 / 5.30**	**1.39 / 3.48**	**1.12**	**1.06**
Method in [23] (One-stage)	2.43 / 6.08	1.50 / 3.75	1.57	1.47
Method in [40] (Two-stage)	3.48 / 8.70	2.19 / 5.48	1.16	1.10

TABLE 6. Ablation experiment of the proposed method on the SNU dataset.

Method			Loose criteria (12 pixels, 10^0)		Tight criteria (6 pixels, 5^0)	
			Recall	Precision	Recall	Precision
I	Region-specific ROIs		93.02%	92.95%	68.92%	68.87%
	Multi-scale feature maps					
II	Region-specific ROIs	✓	95.08%	95.05%	80.99%	80.96%
	Multi-scale feature maps					
III	Region-specific ROIs	✓	**95.75%**	**95.78%**	**83.14%**	**83.16%**
	Multi-scale feature maps	✓				

(✓ indicates included)

Table 4 presents the detection performances of the proposed method and two recently released methods. The two previous methods are the one-stage method by Suhr and Jung [23] and the two-stage method by Do and Choi [40]. They were selected for the comparison because they achieved state-of-the-art performances on the PS2.0 and SNU dataset, respectively. In Table 4, the one-stage method in [23] shows a slightly higher performance than the two-stage method in [40]. As mentioned in the introduction, it is mainly because the two-stage approach has not yet been adequately specialized for parking slot detection. It can be noticed that the proposed method, a highly specialized two-stage parking slot detector, outperforms the others roughly by 3% to 5% with the loose criteria and by 11% to 13% with the tight criteria. This result signifies that the two-stage approach can outperform the one-stage approach in parking slot detection if it is well-specialized, the same as the case of general object detection. In addition, when tightening the criteria, the performance of the proposed method drops only about 12%, while those of the others dramatically drop about 20%. This is primarily because the proposed method provides more accurate positions of the parking slots compared to the others. Table 5 gives the detailed positioning accuracies of the three methods. These errors were calculated from the correctly detected parking slots only. This table clearly shows that both the location and orientation errors of the proposed method are smaller than those of the others. In autonomous parking systems, positioning accuracy is significantly important because vehicles should be controlled based on the detected position of the parking slots.

Table 6 shows the result of the ablation experiment. Since this paper proposes the region-specific multi-scale feature

TABLE 7. Comparison of type and occupancy classification performances on the SNU dataset.

Method	Type classification rate	Occupancy classification rate
Proposed method (Two-stage)	99.92%	99.07%
Method in [23] (One-stage)	99.84%	98.84%
Method in [40] (Two-stage)	100%	99.29%

TABLE 8. Comparison of inference time using Nvidia GeForce RTX 3090 on the SNU dataset.

Method	Time (ms)	Fps	Framework
Proposed method (Two-stage)	22.11	45	TensorFlow
Method in [23] (One-stage)	14.10	71	TensorFlow
Method in [40] (Two-stage)	35.42	28	TensorFlow

TABLE 9. Comparison of model size.

Method	FLOPs	Params	Weight
Proposed method (Two-stage)	22.4G	60.6M	237.7Mb
Method in [23] (One-stage)	67.7G	14.8M	173.4Mb
Method in [40] (Two-stage)	74.5G	64.0M	352.8Mb

extraction, this experiment was conducted focusing on the region-specific ROIs and multi-scale feature maps. In this table, from top to bottom, three cases present the detection results of using none of the region-specific ROIs and multi-scale feature maps, using only the region-specific ROIs without the multi-scale feature maps, and using both, respectively. In case I, the method designates the whole parking slot region as an ROI using a parallelogram as shown in Fig. 5(a). Compared to case I, with the tight criteria, case II reveals that the region-specific ROIs dramatically increase the detection performance by roughly 12%, and case III shows that the region-specific ROIs with the multi-scale feature maps enhance the detection performance by roughly 14%. The performances using the loose criteria have similar trends with smaller gaps. This ablation experiment clearly indicates that the proposed region-specific multi-scale feature extraction improves the parking slot detection performance.

Table 7 shows the type and occupancy classification performances of the three methods. Classification rates are also calculated from the correctly detected parking slots only. The type and occupancy classification rates of the proposed method are all over 99%, and those of the other methods are quite similar. Table 8 presents the inference times of the three methods using Nvidia GeForce RTX 3090. The proposed method is faster than the two-stage method in [40] because its first and second stages share the same backbone network while Do and Choi's method uses two separate backbone networks. Compared to the one-stage method in [23], the proposed method is slower. The inferior inference time observed in the two-stage approach is an expected characteristic. Compared to the one-stage approach, the two-stage

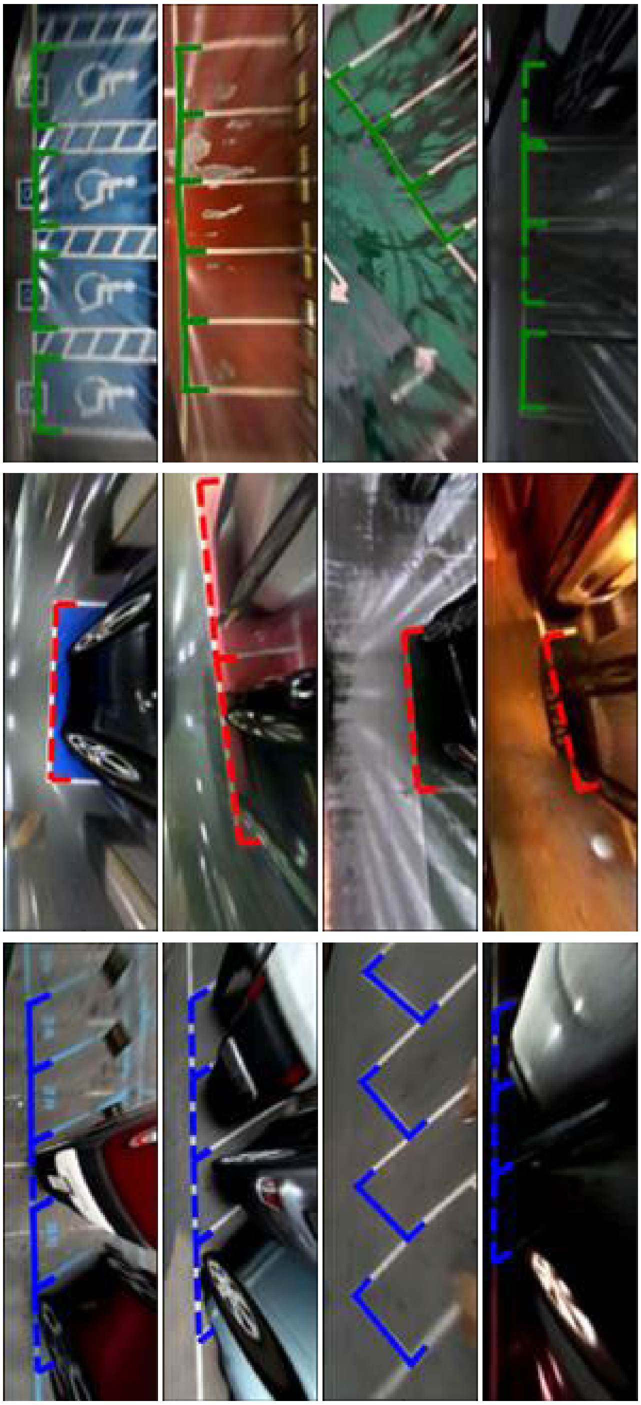


FIGURE 7. Parking slot detection results of the proposed method in various parking scenarios in the test images of the SNU dataset. The first, second, and third rows shows the detection results for perpendicular, parallel, and slanted parking slots, respectively. Green, red, and blue lines indicate perpendicular, parallel, and slanted parking slots, respectively; solid and dashed lines indicate vacant and occupied parking slots, respectively.

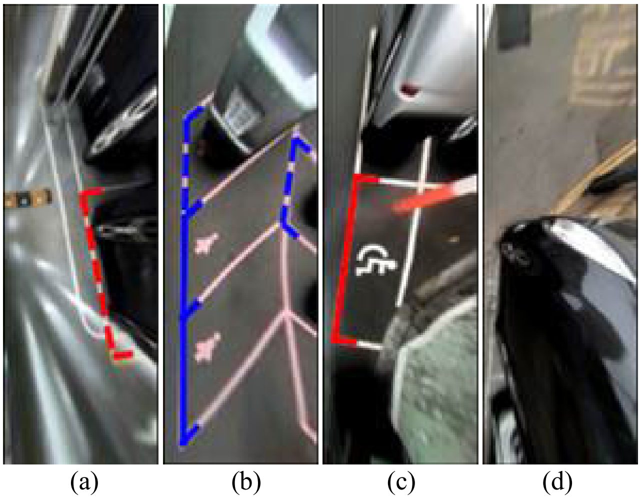


FIGURE 8. Failure cases of the proposed method in the test images of the SNU dataset. (a) and (b) show false positive cases, (c) shows an incorrect occupancy classification, and (d) shows a false negative.

approach achieves higher detection performance with the cost of slower processing time. Table 9 presents the model description of the three methods. It can be observed from the table that even though the proposed method has a slower processing speed than the one-stage method in [23], it has fewer FLOPs (floating-point operations) compared to the latter. This situation can be explained by the approach used in the region proposal network of the proposed method. Instead of using ROIAlign, which comprises lots of computation, the proposed method directly crops out the feature from the designated location of the region proposal. Although this approach still takes considerable time, it has significantly reduced the number of FLOPs.

Fig. 7 illustrates the parking slot detection results in various parking situations contained in the test images of the SNU dataset. In this figure, green, red, and blue lines indicate perpendicular, parallel, and slanted parking slots, respectively, and solid and dashed lines indicate vacant and occupied parking slots, respectively. It is apparent that the proposed method can successfully detect and classify parking slots under various illumination conditions (indoor, outdoor, nighttime, daytime, etc.), ground conditions (strong shadow, reflective floor, brick, grass, etc.) as well as parking slot styles (different colors, with marks, double separating line, etc.).

Fig. 8 presents failure cases of the proposed method in the test images of the SNU dataset. Fig. 8(a)-(b) show false positives. In Fig. 8(a), the lower junction of the detected parking slot does not satisfy the location criterion due to the reflective floor. In Fig. 8(b), the rear part of the parking slot marking (rightmost blue line) is wrongly detected due to the shape similarity. In Fig. 8(c), the parking slot occupied by a pole is misclassified as vacant because the pole occupies only a tiny area. Fig. 8(d) shows a false negative, where the upper junction of the parking slot is heavily occluded by the parked vehicle. Among these failure cases, cases (a) and (d) can be solved by associating sequential information. Occluded or blurred junctions can be seen and correctly detected when the vehicle moves to a different position. Case (b) can be solved by adding post-processing to remove predictions at the rear of another parking slot. And case (c) can be solved by adding more training data for parking slots occupied by small-size objects. These solutions are considered in our future research directions because this paper focuses mainly on properly improving the parking slot performance of the two-stage detection scheme.

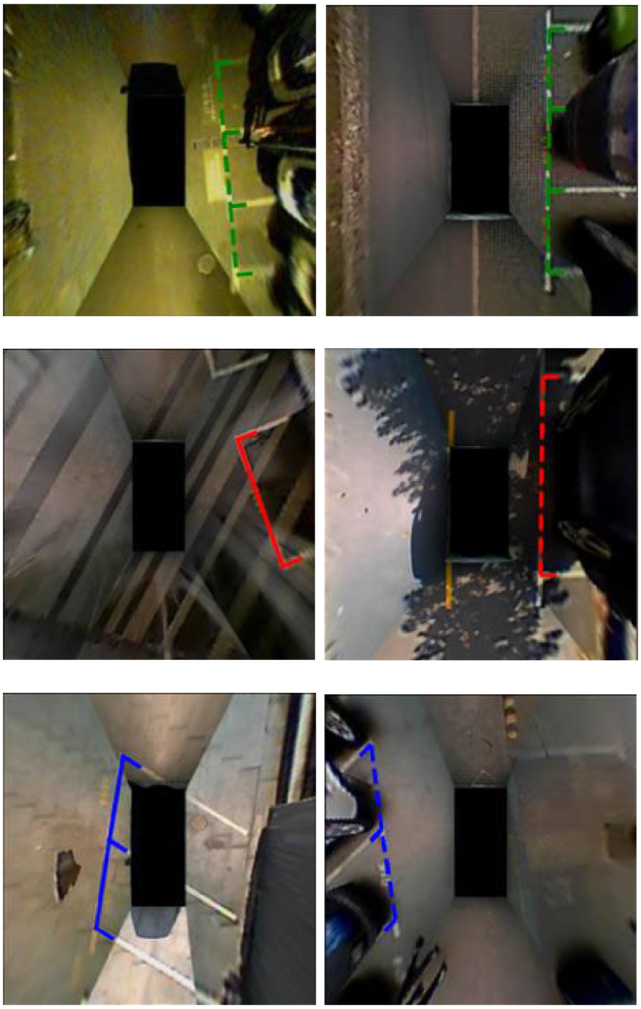
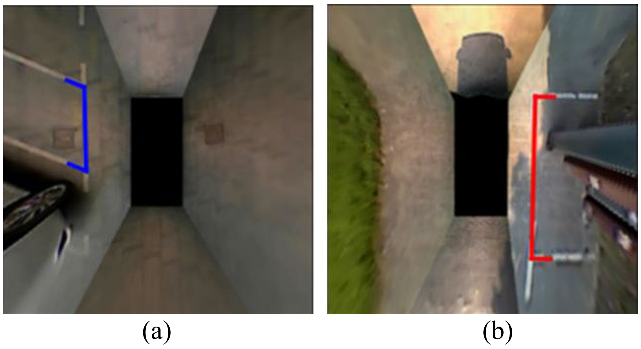


FIGURE 9. Parking slot detection results of the proposed method in the test images of the PS2.0 dataset. Green, red, and blue lines indicate perpendicular, parallel, and slanted parking slots, respectively; solid and dashed lines indicate vacant and occupied parking slots, respectively.

D. PERFORMANCE ON THE PS2.0 DATASET

Table 10 shows the comparison of the parking slot detection performances on the PS2.0 dataset. For the PS2.0 dataset, three more methods have been added for the comparison because more papers shared their codes and detection results, unlike the newly opened SNU dataset. In Table 10, the proposed method shows a slightly higher parking slot detection performance than the others. Note that the performance gaps



(a) (b)

FIGURE 10. Failure cases of the proposed method in the test images of the PS2.0 dataset. (a) shows a false negative, (b) shows an incorrect occupancy classification.

TABLE 10. Comparison of parking SLOT detection performances on the PS2.0 dataset.

Method	Loose criteria (12 pixels, 10 degrees)		Tight criteria (6 pixels, 5 degrees)	
	Recall	Precision	Recall	Precision
Proposed method	**99.77%**	**99.77%**	**99.54%**	**99.54%**
Method in [23]	99.77%	99.77%	99.45%	99.45%
Method in [40]	94.43%	95.22%	73.35%	73.97%
VPS [15]	99.31%	99.40%	99.22%	99.17%
DMPR [18]	93.13%	96.51%	92.34%	95.70%
DeepPS [17]	98.99%	99.63%	97.88%	98.51%

TABLE 11. Comparison of type and occupancy classification performances on the PS2.0 dataset.

Method	Type classification rate	Occupancy classification rate
Proposed method	100%	99.49%
Method in [23]	100%	99.31%
Method in [40]	100%	99.40%
VPS [15]	100%	98.54%
DMPR [18]	N/A	98.33%
DeepPS [17]	N/A	N/A
Proposed method	N/A	N/A

on the PS2.0 dataset are not as apparent as on the SNU dataset because almost all methods have already reached very high detection performances on this dataset. This is mainly due to the similarity between the training and test images of the PS2.0 dataset. This similarity makes it hard to be used to compare the performances of different methods. Table 11 compares the type and occupancy classification performances on the PS2.0 dataset. It also shows that the proposed method gives a slightly higher parking slot classification performance than the others. The previous methods with no ability for type or occupancy classification are masked as N/A.

Fig. 9 illustrates the parking slot detection results in various parking situations contained in the test images of the PS2.0 dataset. It also shows that the proposed method can properly handle the various situations included in the PS2.0 dataset. Fig. 10 presents failure cases of the proposed method on the PS2.0 dataset. Fig. 10(a) includes a false negative where the lower parking slot is undetected because one of its junctions is severely blurred. In Fig. 10(b), the occupied parking slot is misclassified as vacant.

V. CONCLUSION

This paper proposes a novel highly specialized two-stage parking slot detection method using the region-specific multi-scale feature extraction. The proposed method finds parking slot entrances as region proposals in the first stage and extracts region-specific features from multi-scale feature maps to precisely predict positions and properties of parking slots in the second stage. This method was quantitatively evaluated using two large-scale public parking slot detection datasets and outperformed previous methods in terms of both detection performance and positioning accuracy. This result revealed that the two-stage approach is superior to the one-stage approach if it is adequately specialized, the same as the case of general object detection. In the future, we are planning to optimize the network using filter pruning and weight quantization to implement it in real-time embedded systems. In addition, we are trying to improve the performance by integrating sequential detection results and adding task-specific post-processing and rarer parking slot cases.

REFERENCES

[1] H. Banzhaf, D. Nienhüser, S. Knoop, and J. M. Zöllner, "The future of parking: A survey on automated valet parking with an outlook on high density parking," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1827–1834, doi: 10.1109/IVS.2017.7995971.

[2] M. Khalid, K. Wang, N. Aslam, Y. Cao, N. Ahmad, and M. K. Khan, "From smart parking towards autonomous valet parking: A survey, challenges and future works," *J. Netw. Comput. Appl.*, vol. 175, Feb. 2021, Art. no. 102935, doi: 10.1016/j.jnca.2020.102935.

[3] J. K. Suhr and H. G. Jung, "Survey of target parking position designation for automatic parking systems," *Int. J. Automot. Technol.*, vol. 24, no. 1, pp. 287–303, Feb. 2023, doi: 10.1007/s12239-023-0025-6.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[5] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[7] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–16.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 386–397, Feb. 2020, doi: 10.1109/TPAMI.2018.2844175.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[10] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.

[11] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. Mark Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0_2.

[14] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.

[15] W. Li, L. Cao, L. Yan, C. Li, X. Feng, and P. Zhao, "Vacant parking slot detection in the around view image based on deep learning," *Sensors*, vol. 20, no. 7, p. 2138, Apr. 2020, doi: 10.3390/s20072138.

[16] A. Zinelli, L. Musto, and F. Pizzati, "A deep-learning approach for parking slot detection on surround-view images," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 683–688, doi: 10.1109/IVS.2019.8813777.

[17] L. Zhang, J. Huang, X. Li, and L. Xiong, "Vision-based parking-slot detection: A DCNN-based approach and a large-scale benchmark dataset," *IEEE Trans. Image Process.*, vol. 27, no. 11, pp. 5350–5364, Nov. 2018, doi: 10.1109/TIP.2018.2857407.

[18] J. Huang, L. Zhang, Y. Shen, H. Zhang, S. Zhao, and Y. Yang, "DMPR-PS: A novel approach for parking-slot detection using directional marking-point regression," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2019, pp. 212–217, doi: 10.1109/ICME.2019.00045.

[19] C. Jang and M. Sunwoo, "Semantic segmentation-based parking space detection with standalone around view monitoring system," *Mach. Vis. Appl.*, vol. 30, no. 2, pp. 309–319, Mar. 2019, doi: 10.1007/s00138-018-0986-z.

[20] S. Jiang, H. Jiang, S. Ma, and Z. Jiang, "Detection of parking slots based on mask R-CNN," *Appl. Sci.*, vol. 10, no. 12, p. 4295, Jun. 2020, doi: 10.3390/app10124295.

[21] C. Min, J. Xu, L. Xiao, D. Zhao, Y. Nie, and B. Dai, "Attentional graph neural network for parking-slot detection," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3445–3450, Apr. 2021, doi: 10.1109/LRA.2021.3064270.

[22] W. Li, H. Cao, J. Liao, J. Xia, L. Cao, and A. Knoll, "Parking slot detection on around-view images using DCNN," *Frontiers Neurorobotics*, vol. 14, p. 46, Jul. 2020, doi: 10.3389/fnbot.2020.00046.

[23] J. K. Suhr and H. G. Jung, "End-to-end trainable one-stage parking slot detection integrating global and local information," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 4570–4582, May 2022, doi: 10.1109/TITS.2020.3046039.

[24] H. Gi Jung, D. S. Kim, P. J. Yoon, and J. Kim, "Parking slot markings recognition for automatic parking assist system," in *Proc. IEEE Intell. Vehicles Symp.*, 2006, pp. 106–113, doi: 10.1109/IVS.2006.1689613.

[25] K. Hamada, Z. Hu, M. Fan, and H. Chen, "Surround view based parking lot detection and tracking," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2015, pp. 1106–1111, doi: 10.1109/IVS.2015.7225832.

[26] C. Wang, H. Zhang, M. Yang, X. Wang, L. Ye, and C. Guo, "Automatic parking based on a bird's eye view vision system," *Adv. Mech. Eng.*, vol. 6, Jan. 2014, Art. no. 847406, doi: 10.1155/2014/847406.

[27] S. Kim, J. Kim, M. Ra, and W.-Y. Kim, "Vacant parking slot recognition method for practical autonomous valet parking system using around view image," *Symmetry*, vol. 12, no. 10, p. 1725, Oct. 2020, doi: 10.3390/sym12101725.

[28] X. Du and K. K. Tan, "Autonomous reverse parking system based on robust path generation and improved sliding mode control," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1225–1237, Jun. 2015, doi: 10.1109/TITS.2014.2354423.

[29] S. Lee and S. Seo, "Available parking slot recognition based on slot context analysis," *IET Intell. Transp. Syst.*, vol. 10, no. 9, pp. 594–604, Nov. 2016, doi: 10.1049/iet-its.2015.0226.

[30] W. Zong and Q. Chen, "A robust method for detecting parking areas in both indoor and outdoor environments," *Sensors*, vol. 18, no. 6, p. 1903, Jun. 2018, doi: 10.3390/s18061903.

[31] J. Suhr and H. Jung, "A universal vacant parking slot recognition system using sensors mounted on off-the-shelf vehicles," *Sensors*, vol. 18, no. 4, p. 1213, Apr. 2018, doi: 10.3390/s18041213.

[32] J. Chen and C. Hsu, "A visual method tor the detection of available parking slots," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2017, pp. 2980–2985, doi: 10.1109/SMC.2017.8123081.

[33] J. K. Suhr and H. G. Jung, "Automatic parking space detection and tracking for underground and indoor environments," *IEEE Trans. Ind. Electron.*, vol. 63, no. 9, pp. 5687–5698, Sep. 2016, doi: 10.1109/TIE.2016.2558480.

[34] H. Gi Jung, Y. Hee Lee, and J. Kim, "Uniform user interface for semi-automatic parking slot marking recognition," *IEEE Trans. Veh. Technol.*, vol. 59, no. 2, pp. 616–626, Feb. 2010, doi: 10.1109/TVT.2009.2034860.

[35] J. K. Suhr and H. G. Jung, "Fully-automatic recognition of various parking slot markings in around view monitor (AVM) image sequences," in *Proc. 15th Int. IEEE Conf. Intell. Transp. Syst.*, Sep. 2012, pp. 1294–1299, doi: 10.1109/ITSC.2012.6338615.

[36] J. K. Suhr and H. G. Jung, "Sensor fusion-based vacant parking slot detection and tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 21–36, Feb. 2014, doi: 10.1109/TITS.2013.2272100.

[37] L. Li, L. Zhang, X. Li, X. Liu, Y. Shen, and L. Xiong, "Vision-based parking-slot detection: A benchmark and a learning-based approach," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2017, pp. 649–654, doi: 10.1109/ICME.2017.8019419.

[38] S. Houben, M. Komar, A. Hohm, S. Lüke, M. Neuhausen, and M. Schlipsing, "On-vehicle video-based parking lot recognition with fisheye optics," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 7–12, doi: 10.1109/ITSC.2013.6728595.

[39] Z. Zhong, L. Sun, and Q. Huo, "An anchor-free region proposal network for faster R-CNN-based text detection approaches," *Int. J. Document Anal. Recognit.*, vol. 22, no. 3, pp. 315–327, Sep. 2019, doi: 10.1007/s10032-019-00335-y.

[40] H. Do and J. Y. Choi, "Context-based parking slot detection with a realistic dataset," *IEEE Access*, vol. 8, pp. 171551–171559, 2020, doi: 10.1109/ACCESS.2020.3024668.

[41] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2261–2269, doi: 10.1109/CVPR.2017.243.

[42] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128837–128868, 2019, doi: 10.1109/ACCESS.2019.2939201.

[43] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, Feb. 2020, doi: 10.1007/s11263-019-01247-4.

[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

QUANG HUY BUI received the B.S. degree in mechatronics engineering from the Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam, in 2019. He is currently pursuing the Ph.D. degree with the Department of Intelligent Mechatronics Engineering, Sejong University, Seoul, South Korea. His research interests include computer vision and deep learning, with a focus on applications for autonomous vehicles.

JAE KYU SUHR (Member, IEEE) received the B.S. degree in electronic engineering from Inha University, Incheon, South Korea, in 2005, and the M.S. and Ph.D. degrees in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2007 and 2011, respectively.

From 2011 to 2016, he was with the Automotive Research Center, Hanyang University, Seoul. From 2016 to 2017, he was with the Korea National University of Transportation, Chungju, South Korea. He is currently an Associate Professor with the Department of Intelligent Mechatronics Engineering, Sejong University, Seoul. His research interests include computer vision, image analysis, pattern recognition, and sensor fusion for intelligent and autonomous vehicles.

• • •