A STUDY OF PARKING-SLOT DETECTION WITH THE AID OF PIXEL-LEVEL DOMAIN ADAPTATION

Juntao Chen[1], *Lin Zhang*[1,*], *Ying Shen*[1,*], *Yong Ma*[2], *Shengjie Zhao*[1], *Yicong Zhou*[3]

[1]School of Software Engineering, Tongji University, Shanghai, China
[2]School of Computer Information Engineering, Jiangxi Normal University, China
[3]Department of Computer and Information Science, University of Macau, Macau

ABSTRACT

The self-parking system is an important component of self-driving vehicles. Such a system needs to detect and locate the parking-slots from surround-view images, and then guide the vehicle to the designated parking-slot. In the real world, the appearances and environmental conditions of parking-slots can be rich and varied. Thus, to train the parking-slot detection model, it is necessary to collect and label a huge quantity of surround-view images covering as many real cases as possible. Such a process is cumbersome and costly, and will be repeated whenever encountering an unseen parking condition that is quite different from the ones covered by existing training set. To this end, in this paper we propose an extensible pipeline, namely FakePS, to assist parking-slot detection model training by making use of synthetic data. Specifically, with FakePS, we can first build various simulated parking scenes and collect labeled surround-view images automatically. Besides, we resort to pixel-level domain adaptation strategies to enhance the realism of the synthetic images using unlabeled real images while preserving their label information. The efficacy of FakePS has been corroborated by experimental results.

***Index Terms*—** Self-parking system, parking-slot detection, learning by synthesis, domain adaptation, image realism enhancement

1. INTRODUCTION

As part of the self-parking systems, the accuracy of the parking-slot detection model is very important. In order to improve the detection accuracy, a large amount of data must be prepared during the training phase. However, the variety of parking-slots, different ground textures or lighting conditions will affect the performance of the detection model, so it is not practical to build a training set covering all situations from the beginning. Having investigated the literature, we find that in the field of parking-slot detection, although the methods are diverse, they are all limited by real-world datasets. In other words, the training data cannot cover all possible scenarios. These methods often collect certain training data and test the performance under a limited number of test samples. Once new test conditions are encountered, such as rainy days or brick roads, their performance will be compromised.

For the above reasons, whenever a new parking scene appears, it is necessary to use the experimental vehicle to collect a large number of surround-view images of the specified scene. It is even more time-consuming and laborious to manually label the collected images with parking-slot information, and finally retrain the detection model. This entire process may take more than one week. We find that although ps2.0 [1], the benchmark dataset in this field, contains 12,165

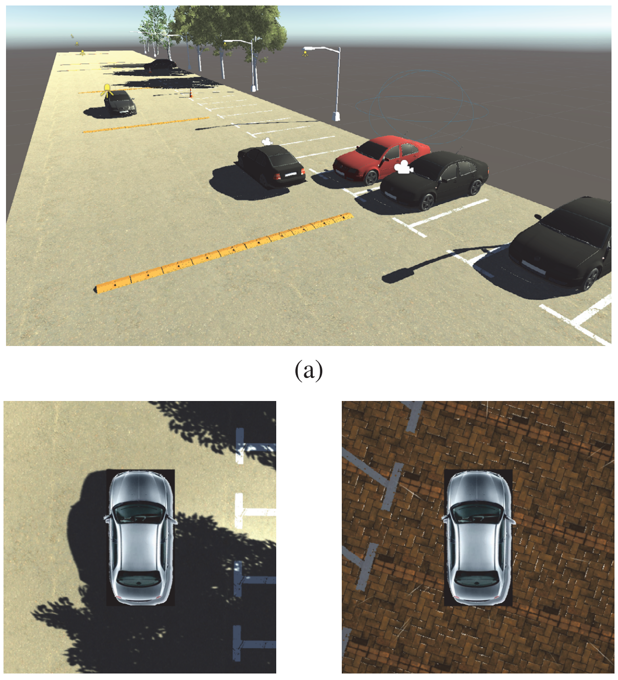

(a) (b) (c)

Fig. 1. (a) is a screenshot of the built-up Unity parking scenes. (b) and (c) are examples of surround-view images with parking-slots collected from built-up scenes.

*Corresponding authors: {cslinzhang, yingshen}@tongji.edu.cn.

978-1-7281-1331-9/20/$31.00 ©2020 IEEE

images, it still cannot cover all parking conditions.

In this work, we attempt to fill the aforementioned research gaps to some extent, that is, to train the model with synthetic images instead of real images. In order to obtain synthetic images, we built some scalable parking scenes in Unity, covering various weather (sunny, cloudy, rainy, etc.), pavement materials (cement, bricks, waterlogged, etc.) and parking-slot types (right-angle, slanted, etc.). A screenshot of the built scenes and some sample collected surround-view images are shown in Fig. 1. By applying unsupervised pixel-level domain adaptation technology, the realism of the synthetic images is further improved, which makes the parking-slot detection accuracy of the models for new scenes further increase. Our contributions are summarized as follows:

- We are the first to train parking-slot detection models with synthetic images instead of the expensive real ones. We propose an extensible pipeline, namely FakePS, to assist parking-slot detection model training by making use of synthetic data. The flowchart of FakePS is shown in Fig. 2. When an unseen parking condition appears, FakePS allows us to easily obtain the corresponding synthetic data and retrain the parking-slot detection model. The pipeline eliminates cumbersome data collection and labeling.
- To complement the scenes that are not covered by the existing datasets, a large number of simulated parking scenes have been built. These scenes include various weather conditions, road textures, and parking-slot types. The scenes can be combined with each other. By traversing all combinations, the number of scenes exceeds 120. We collect surround-view images in these scenes and corresponding parking-slot annotations. With these collected images, we build a synthetic parking-slot dataset that covers 17 major scenes with over 23,000 images. Besides, these scenes are extensible, and new scenes can be obtained by simple modification.
- In addition, we enhanced the realism of these images by applying pixel-level domain adaptation strategies. A customized loss function makes the image more realistic while preserving its parking-slot annotation after transformation. More importantly, the performance of the parking-slot detection model can be further improved after using the refined images compared to that of the model trained with the synthetic ones.

2. RELATED WORK

Vision-based parking-slot detection. Parking space detection methods include free-space-based ones and vision-based ones. The free-space-based approaches [2, 3, 4, 5, 6] locate the target parking position by identifying sufficient free space between adjacent vehicles. They usually depend on vehicles that have already parked, so their application is limited and not practical. In contrast, vision-based ones have been widely studied by scholars for their superior performance.

The vision-based parking-slot detection approaches can be divided into three categories, namely user interaction ones, line-based ones, and point-based ones. User interaction approaches [7] require manual operation, and they are not fully automatic. Line-based detection methods focus on finding marking-lines [8, 9, 10]. They use line fitting algorithms to detect marking-lines and then distinguish the entrance-lines and separating-lines by their geometric relations. In practice, these approaches are sensitive to other edges (such as the edges of the car) in the surround-view image, resulting in unsatisfactory performance. Point-based methods first predict the locations of marking-points and then, for each pair of marking-points, determine if they can form an entrance-line of a parking-slot. Low-level point-based approaches rely on traditional point detection methods like the Harris Corner detection algorithm [11, 12] or a boosting decision tree [13]. DeepPS is the first to predict the marking-points and their patterns with deep convolutional neural networks [1]. After that, DMPR-PS proposed a parking-slot detection method based on directional marking-point regression [14]. DeepPS and DMPR-PS have achieved better performance than previous methods. However, their training data and test data are limited to ps2.0 [1], and they do not always achieve good results in new test data.

Unsupervised image to image translation is a branch of domain adaptation. It aims to learn a model that maps one type of images to another without pair supervision and retains some relevant features. This task can be traced back to Unsupervised Image Translation by Resales *et al.* [15], who employed a Bayesian model with a prior based on a patch-based Markov random field obtained from the source image. More recently, the methods based on generative adversarial networks (GANs [16]) have become the main solution to unsupervised image to image translation problems. CycleGAN [17] and DualGAN [18] proposed a cycle consistency loss that attempts to preserve the input image after a cycle of transformation (forward and backward). There are also some methods tried to preserve pixel values [19, 20] or high level features [21] while translating images. Liu *et al.* proposed UNIT [22] to solve this problem, which is based on variational autoencoders (VAEs). Assuming that both encoders share the same latent space, UNIT enforces weight sharing between the last few layers of the encoders and between the first few layers of the generators. Some of these methods have achieved impressive image translation results, but few of them use the translation results for a specific computer vision task.

3. PARKING SCENES SIMULATION AND DATA COLLECTION

Before building the scenes in Unity, we need to prove the feasibility of this solution. As we know, the input to the

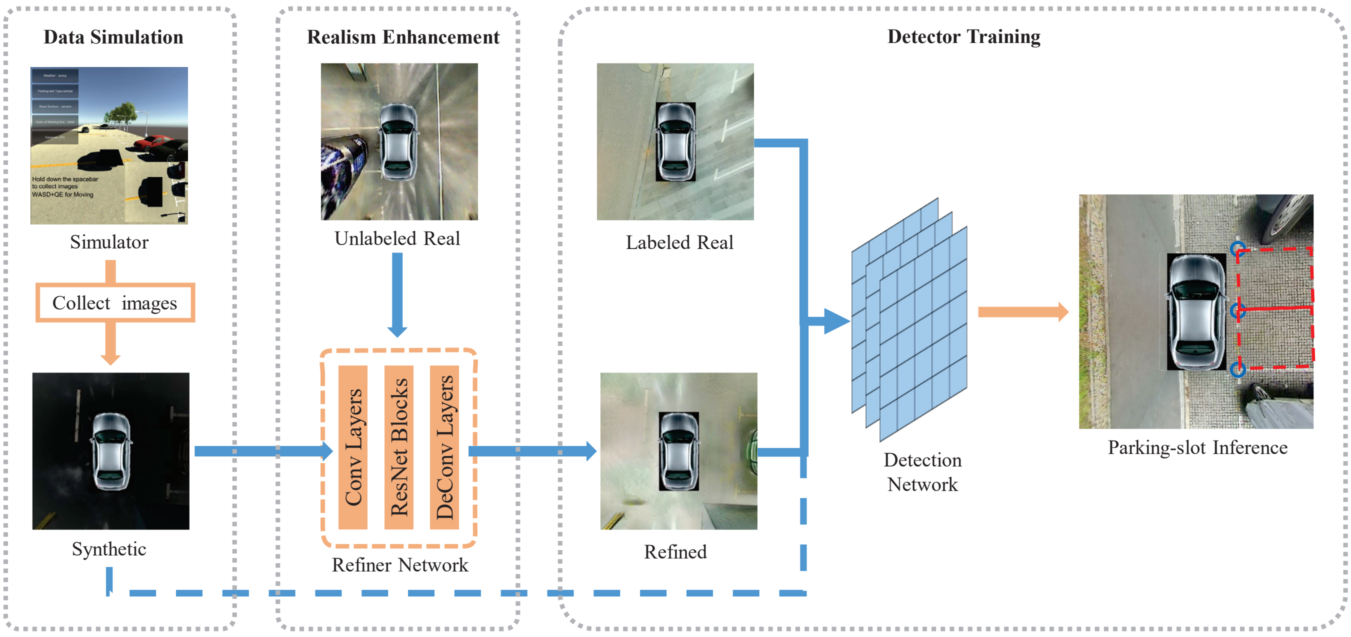


Fig. 2. The flowchart of FakePS. We first get the synthetic images of the specific built-up scenes, and then use the refiner network to enhance the realism of the synthetic images. The detector is finally trained with the mixed dataset that includes real images and synthetic (or refined) ones.

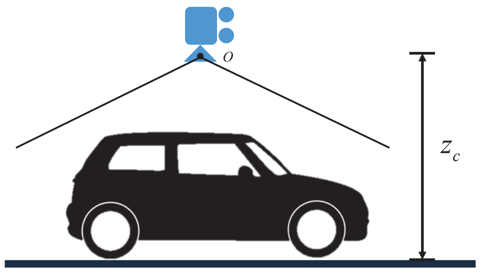


Fig. 3. Simplified surround-view image acquisition model, in which the surround-view images are directly captured by a wide-angle virtual camera.

parking-slot detection model is a surround-view image. The purpose of generating a surround-view image is to get a similarity transformation from the ground to the image, which means the transformation preserves the similarity ratio of objects [23]. Thus the parking-slot detected in the image can be conveniently found in the real world according to the similarity ratio.

To be more specific, we construct a 3D Cartesian coordinate system based on the right-hand rule, whose XY-plane coincides with the ground plane and the positive Z-axis points up. This coordinate system is called the world coordinate system, abbreviated as WCS. Based on the above description, Suppose there is a point P on the ground plane. After projection, its image in the surround-view is P'. The coordinates of P in the WCS are $(X, Y, 0)^\top$ while the coordinates of P' in the image coordinate system (ICS) are $(u, v)^\top$. The relationship between $(X, Y, 0)^\top$ and $(u, v)^\top$ can be expressed as:

$$(u, v)^\top = (\frac{X}{s}, \frac{Y}{s})^\top \tag{1}$$

where s represents the similarity ratio.

Fig. 3 illustrates a simplified model of how to collect surround-view images in this work, where O represents the optical center of the virtual camera and Z_c represents the distance from the optical center to the ground. In this model, surround-view images are taken directly by a vertically downward wide-angle camera. However, in practice, the surround-view image is usually formed by stitching images captured by four car-mounted cameras. Then the question arises, whether the "surround-view image" captured by a wide-angle camera overhead also preserves the similarity ratio?

The homogeneous coordinates of P in WCS are denoted by P_w. After projection, the homogeneous coordinates of corresponding point P'' in the "surround-view" ICS are denoted by P_{uv}. According to the pinhole camera model [24], P_{uv} is given by:

$$ZP_{uv} = Z \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = KTP_w \tag{2}$$

where P_w equals $(X, Y, 0, 1)^\top$, T is the camera extrinsic matrix, K is the camera intrinsic matrix and Z is the normalization coefficient. Note that the latter formula implies a conversion from homogeneous to non-homogeneous coordinates.

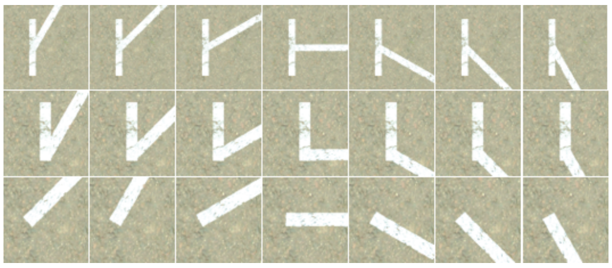


Fig. 4. Virtual marking-lines. The first row is **T**-shaped marking-lines, the second row is **L**-shaped ones, and the third row is **I**-shaped ones.

Since the virtual camera is very accurate, T and K satisfy:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & Z_c \\ 0 & 0 & 0 & 1 \end{bmatrix}, K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

where Z_c is the height of the optical center O mentioned before, and f is the focal length of the camera. Substituting expressions for T and K into Eq. 2, we obtain:

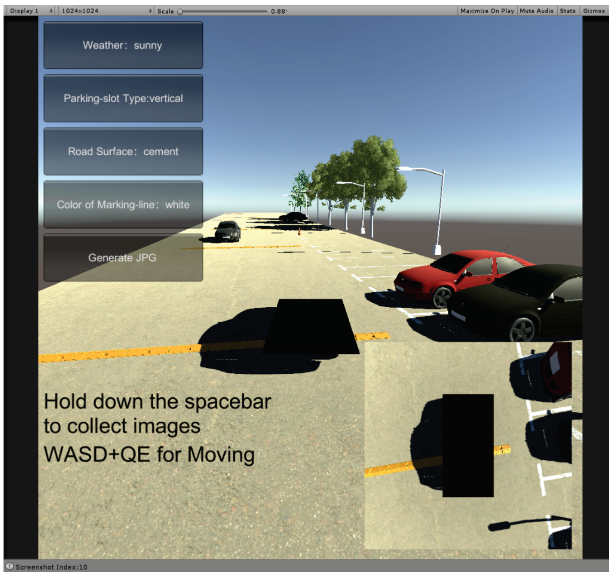
$$P_{uv} = [u', v', 1]^\top = [\frac{fX}{Z_c}, \frac{fY}{Z_c}, 1]^\top \quad (4)$$

In the end, we get a simple form, where $\frac{Z_c}{f}$ is the similarity ratio, which means that capturing surround-view images by a wide-angle camera is feasible. We can adjust the height or focal length of the camera to achieve the same effect as the real surround-view image.

To create the synthetic dataset, there are three main steps:

Firstly, create all assets we need, such as various types of road texture maps, various objects (speed bumps, traffic cones, trees, street lights, etc.) that may appear in parking scenes, and of course, car models. Among these assets, the model and texture of the marking-line are particularly important since they are the patterns directly learned by the parking-slot detector. As shown in Fig. 4, one of the benefits of synthesizing marking-lines by computer is that we can adjust the size and angle of the marking-lines arbitrarily.

Secondly, use these assets to build each scene. By adjusting the intensity and orientation of the light source, the strength of the shadows, the intensity of the environment lighting, we can simulate scenes of the sunny, cloudy and night. By changing texture maps, we can simulate various road surfaces. Furthermore, by applying a customized shader and particle system, scenes of flooded road or rainy condition can also be simulated. Don't forget to put some objects in the scene to make it more realistic. For example, in a real parking lot, there should be some parked vehicles and shadows cast by trees. The final number of scenes can be more than 120 kinds based on the combination of weather conditions, pavement materials, parking-slot types, etc.



Weather: sunny

Parking-slot Type:vertical

Road Surface: cement

Color of Marking-line: white

Generate JPG

Hold down the spacebar
to collect images
WASD+QE for Moving

Fig. 5. User interface of the Unity project.

Thirdly, capture surround-view images automatically using scripts. Fig. 5 is the user interface of this Unity project. As shown in the figure, we can switch freely between scenes through the four buttons in the upper left. The lower right corner is a thumbnail of the surround-view image that can be collected at the current position. Through the world coordinates of marking-points and the matrices of the camera, we can determine whether each marking-point can be seen by the wide-angle camera. If the answer is yes, record the parking-slot information and generate the label file.

Two things need to be noticed here. First, the virtual experimental vehicle is not rendered while its shadow is rendered. This is because the vehicle is located directly below the camera as is shown in Fig. 3. Due to perspective, it will occupy a large area in the center of surround-view, which is against the intention. Secondly, the height of the camera must be slightly lower than the height of the car, which is about 1.2 meters. Eq. 4 shows that by adjusting $\frac{Z_c}{f}$, the similarity ratio can be the same as that of the real surround-view image. However, since not all objects are on the ground plane, such as traffic cones and other vehicles, there is a parallax between the collected surround-view image and the real one. For the above reason, strictly, we need to place the virtual camera close to the real car-mounted cameras to reduce parallax. Therefore, we choose a fixed camera height $\hat{Z}_c$ (1.2 meters) and adjust the camera focal length f to obtain the required similarity ratio.

4. REALISM ENHANCEMENT WITH PARKING-SLOT CONSTANCY

Although synthetic images can be directly used to train detection models, there is still a gap between the distribution of the synthetic images and the real ones. To bridge the gap, a

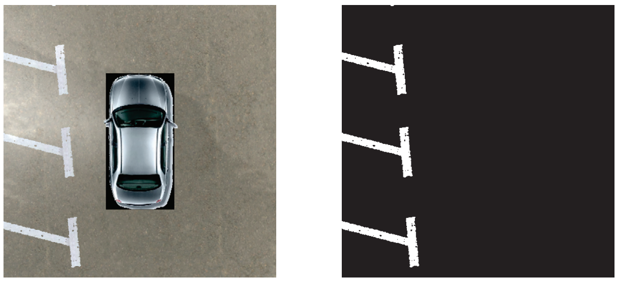

(a) (b)

Fig. 6. (a) is a sample of surround-view image, and (b) is the corresponding mask.

parking-slot constancy approach is proposed, which can add realism to the synthetic images. The parking-slot information of the image must be strictly unchanged after image translation so that the output image can still be directly used to train the detection network.

Since the semantic information of each object in Unity is clear, we can simply get the mask of the marking-lines in each surround-view image. Specifically, while rendering each surround-view image, we render a binary image containing only the marking-lines as the mask as is shown in Fig. 6.

We adopt the architecture for our realism enhancement networks from Zhu *et al.* [17] who have shown impressive results for unpaired image translation. Our goal is to learn a mapping function G from synthetic image domain X to real image domain Y. Assuming the set of masks of surround-view images is M, the objective of parking-slot constancy can be expressed as:

$$\mathcal{L}_{const}(G, X, M) = \mathbb{E}[\|(G(x) - x) \odot M_x\|_1] \tag{5}$$

where $\odot$ denotes the entrywise product, $x \in X$ and M_x represents the mask of x, which is a binary matrix. This objective function drives the pixel value of the generated image $G(x)$ in the mask area to be consistent with x. Assuming F is the mapping function from Y to X and D_X, D_Y are the discriminator for F and G, respectively, the full objective can be expressed as:

$$\begin{aligned}\mathcal{L}(G, F) = &\mathcal{L}_{GAN}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{GAN}(F, D_X, Y, X) \\ &+ \lambda_1 \mathcal{L}_{cyc}(G, F) \\ &+ \lambda_2 \mathcal{L}_{const}(G, X, M)\end{aligned} \tag{6}$$

where $\mathcal{L}_{GAN}$ and $\mathcal{L}_{cyc}$ are the traditional GAN loss and cycle consistency loss, respectively; λ_1 and λ_2 control the relative weights of loss terms. The details of $\mathcal{L}_{GAN}$ and $\mathcal{L}_{cyc}$ can be found in [17].

With the advantage of the parking-slot constancy loss, we can get images with improved realism while retaining its annotation completely.

Table 1. Synthetic image experimental results.

method	training set	precision	recall
DeepPS [1]	ps2.0	97.10%	90.48%
	ps2.0 + synthetic data	97.14%	92.87%
DMPR-PS [14]	ps2.0	97.06 %	92.29%
	ps2.0 + synthetic data	**97.22%**	**94.92%**

Table 2. Refined image experimental results.

method	training set	precision	recall
DeepPS [1]	ps2.0	95.10%	89.87%
	ps2.0 + synthetic data	95.17%	93.27%
	ps2.0 + refined data	95.25%	94.19%
DMPR-PS [14]	ps2.0	95.03%	93.77%
	ps2.0 + synthetic data	95.08%	96.44%
	ps2.0 + refined data	**95.33%**	**97.39%**

5. EXPERIMENTAL RESULTS

To evaluate the efficacy of the pipeline FakePS, we used the benchmark dataset ps2.0 established by Zhang *et al.* [1]. It is the largest dataset in the field of vision-based parking-slot detection, comprising 12,165 surround-view images collected from typical indoor and outdoor parking sites. We trained two representative detection models, DeepPS and DMPR-PS, with additional synthetic or refined images, and observed whether their performance was improved. The architecture of DeepPS and DMPR-PS can be found in [1, 14].

Synthetic image experiments. After careful analysis of ps2.0, we found that there are few brick-paved scenes in ps2.0. To this end, we collected 2,954 synthetic surround-view images from the brick-paved scenes as the training set. We trained the networks on ps2.0 and ps2.0 combined with synthetic data, respectively. Except for the training set, all training settings were consistent with the original DMPR-PS. We used Adam optimizer with 10^{-4} as the initial learning rate. We trained the networks on Nvidia Titan Xp with a batch size of 24 for 12 epochs. Besides, we carefully labeled over 400 surround-view images of brick-paved streets and combined these images with the original ps2.0 test set as the new test set. We adjusted the probability threshold to make the precision of the two models higher than 97%. After we trained the models with the synthetic images, the recall of DeepPS and DMPR-PS increased by 2.39% and 2.63%, respectively. This indicates that after using synthetic images, the performance of DeepPS and DMPR-PS on new scenes can be improved.

Refined image experiments. We removed indoor images from ps2.0, for a total of 2,209 images and then trained the detector on the left images. It can be expected that the performance of the detector is relatively poor because the model has never seen an indoor parking-slot during the training phase. Our goal is to use synthetic images and refined images to improve the performance of the detection model. Just like the flowchart in Fig. 2, firstly, we collected 2,190 synthetic surround-view images from the indoor scenes. Then we collected 2,477 real surround-view images with experimental cars. These images did not need to be manually labeled or even include parking-slots, as long as they were collected from indoor scenes. Finally, we trained DeepPS and DMPR-PS on ps2.0, ps2.0 combined with synthetic data and ps2.0 combined with refined data, respectively. Similarly, except for the training set, all training settings were consistent with the original DMPR-PS. The final models were evaluated on the original test set of ps2.0. The experimental results in Table 2 show that, by training with the extra refined images, the type I errors (when the parking-slot is true but is rejected) of DeepPS and DMPR-PS were reduced by 42.65% and 58.11% with the precision over 95%.

These experimental results suggest that our solution can alleviate the problem of insufficient training data in some scenes to a certain extent. It can not only improve the performance of detection models in new scenes but also avoid the pain of cumbersome data collection and annotation.

6. CONCLUSION

In this paper, a pipeline named FakePS is proposed to improve the performance of parking-slot detection models in new scenes not covered by the real training set. This pipeline allows us to collect a large number of synthetic surround-view images in the scenes of Unity. The synthetic images are then refined with the unlabeled real images while the parking-slots are preserved by our proposed parking-slot constancy loss. The experimental results show that FakePS is effective. We plan to study the effect of the synthetic dataset in complex parking scenes in future work.

7. ACKNOWLEDGMENT

This research was funded in part by the National Natural Science Foundation of China under grants 61672380, 61973235, 61972285, and 61936014 and in part by the Natural Science Foundation of Shanghai under grant 19ZR1461300.

8. REFERENCES

[1] 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. IP*, vol. 27, no. 11, pp. 5350–5364, 2018.

[2] W. J. Park, B. S. Kim, D. E. Seo, D. S. Kim, and K. H. Lee, "Parking space detection using ultrasonic sensor in parking assistance system," in *IEEE Intell. Veh. Symp.*, 2008, pp. 1039–1044.

[3] S. H. Jeong, C. G. Choi, J. N. Oh, P. J. Yoon, B. S. Kim, M. Kim, and K. H. Lee, "Low cost design of parallel parking assist system based on an ultrasonic sensor," *Int. J. Autom. Technol.*, vol. 11, no. 3, pp. 409–416, 2010.

[4] J. Zhou, L. E. Navarro-Serment, and M. Hebert, "Detection of parking spots using 2D range data," in *IEEE Int. Conf. Intell. Transp. Syst.*, 2012, pp. 1280–1287.

[5] R. Dubé, M. Hahn, M. Schütz, J. Dickmann, and D. Gingras, "Detection of parked vehicles from a radar based occupancy grid," in *IEEE Intell. Veh. Symp.*, 2014, pp. 1415–1420.

[6] A. Loeffler, J. Ronczka, and T. Fechner, "Parking lot measurement with 24 GHz short range automotive radar," in *IEEE Int. Radar Symp.*, 2015, pp. 137–142.

[7] H. G. Jung, Y. H. Lee, and J. Kim, "Uniform user interface for semiautomatic parking slot marking recognition," *IEEE Trans. Veh. Technol.*, vol. 59, no. 2, pp. 616–626, 2010.

[8] K. Hamada, Z. Hu, M. Fan, and H. Chen, "Surround view based parking lot detection and tracking," in *IEEE Intell. Veh. Symp.*, 2015, pp. 1106–1111.

[9] J. K. Suhr and H. G. Jung, "A universal vacant parking slot recognition system using sensors mounted on off-the-shelf vehicles," *Sensors*, vol. 18, no. 4, pp. 1213, 2018.

[10] 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, 2016.

[11] J. K. Suhr and H. G. Jung, "Full-automatic recognition of various parking slot markings using a hierarchical tree structure," *Opt. Eng.*, vol. 52, no. 3, pp. 1 – 15, 2013.

[12] 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, 2014.

[13] 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 *IEEE ICME*, 2017, pp. 649–654.

[14] 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 *IEEE ICME*, 2019, pp. 212–217.

[15] R. Resales, K. Achan, and B. J. Frey, "Unsupervised image translation," in *IEEE ICCV*, 2003, pp. 472–478.

[16] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, pp. 2672–2680. 2014.

[17] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE ICCV*, 2017, pp. 2223–2232.

[18] Z. Yi, H. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised dual learning for image-to-image translation," in *IEEE ICCV*, 2017, pp. 2849–2857.

[19] S. Benaim and L. Wolf, "One-sided unsupervised domain mapping," in *NIPS*, pp. 752–762. 2017.

[20] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *IEEE CVPR*, 2017, pp. 2107–2116.

[21] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," in *ICLR*, 2017.

[22] M. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *NIPS*, pp. 700–708. 2017.

[23] D. W. Henderson and D. Taimina, *Experiencing geometry: Euclidean and non-Euclidean with history*, Prentice Hall, 2005.

[24] D. A. Forsyth and J. Ponce, *Computer vision: A modern approach*, Prentice Hall, 2002.