A MULTI-TASK CONVOLUTIONAL NEURAL NETWORK WITH SPATIAL TRANSFORM FOR PARKING SPACE DETECTION

Hoang Tran Vu and Ching-Chun Huang

Department of Electrical Engineering and Advanced Institute of Manufacturing with High-tech Innovations, National Chung Cheng University, Taiwan

ABSTRACT

Vacant parking space detection is a challenging vision task due to outdoor lighting variation and perspective distortion. Previous methods found on camera geometry and projection matrix to select space image region for status classification. By utilizing suitable hand-crafted features, outdoor lighting variation and perspective distortion could be well handled. However, if also considering parking displacement, non-unified car size, and inter-object occlusion, we find the problem becomes more troublesome. To overcome these problems, we propose a deep learning framework to infer the parking status with two contributions. First, we integrate a convolutional spatial transformer network (STN) to crop the local image area adaptively according to car size and parking displacement. Second, in order to solve inter-object occlusion problems, we group 3 neighboring spaces as a unit. A multi-task loss function is designed to consider the status estimation of the target space and its two neighbors jointly. With the loss function, we could force our network to learn occlusion patterns while estimating space status. The results show our system can reduce the error detection rate and thereby increase system accuracy.

Index Terms—Spatial transformer network, Parking space detection, CNN, Deep learning.

1. INTRODUCTION

Many vision-based parking space management systems have been discussed and developed recently. Formulated as a classification problem, parking space detection could be performed by learning a classifier that captures the variation of space appearances in a supervised manner. In general, the whole procedure consists of a feature extraction step and a classifier training step. To represent image content using discriminative features, hand-crafted extraction methods are often used such as: HOG [1-3], Texture features [4-7], Haar-like feature [8], etc. Feature fusion is another strategy to improve feature representation and boost system accuracy [9]. Later on, these features are used to train a classifier whose objective is to minimize the misclassification error on the training dataset and keep the generality for other testing samples. Many learning approaches has been utilized for parking space detection such as: support vector machine (SVM) [4, 6-8, 10-11], fuzzy c-means (FCM) classifier [12], logistic regression [2-3], and so on.

However, hand-crafted features may not be flexible and should be custom-made for each case. To be general, people turn to rely on deep learning method for feature extraction and object classification automatically and simultaneously nowadays. It has been also shown that deep networks, such as convolutional neural networks (CNNs), could achieve better performance in various domains [13]. However, to most of our knowledge, only one recent study [14] used CNNs for parking space detection. In this work, Sepehr Valipour *et al.* proposed to manually crop image patches of interest and train a standard patch-based CNN network for status classification. With CNNs, the extracted features are robust to lighting variation. However, in the case of inter-object occlusion and perspective distortion, as shown in Fig. 1, space status classification becomes a non-trivial problem. Without modification, a standard CNN could not work for all situation.

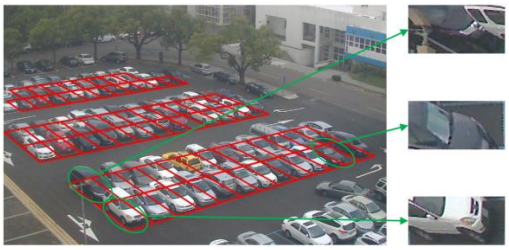


Figure 1. Inter-object occlusion and perspective distortions

Figure 2. Vehicle size and parking displacement problems

To constitute a robust space management system, we should take multiple issues into account including lighting variation, perspective distortion, inter-object occlusion, different vehicle size, and parking displacement. Most of the previous works focus on solving the lighting and distortion problem. A few researchers aim to discuss the effects from

This work was partially supported by Ministry of Science and Technology of Taiwan under Grant No. 105-2622-E-194-008-CC3.

occlusion. However, rare studies probe the influence from vehicle size and parking displacement. As shown in Fig. 2, the non-unified vehicle size and the uncontrollable parking displacement greatly complicate the classification problem.

In this paper, to solve the aforementioned problems, we propose a CNN-based deep learning framework to infer the parking status with two contributions. First, we integrate a convolutional spatial transformer network (STN) to crop the local image area adaptively according to vehicle size and parking displacement. Second, in order to solve inter-object occlusion problems, we group 3 neighboring spaces as a unit. A multi-task loss function is designed to consider the status estimation of the target space and its two neighbors jointly. By using an end-to-end training approach, our network can learn occlusion patterns to classify space status. For lighting variation, the extracted CNN features are robust enough to the effects. Below, we detail the proposed method.

2. THE PROPOSED METHOD

The proposed framework for parking space detection is shown in Fig. 3. The whole framework is divided into three main parts: spatial transformer network (STN), neighbor's hypotheses prediction network (NHPN), and inference layer.

2.1. Spatial Transformer network

As shown in Fig. 3, instead of extracting a specific bounding box around a single parking space, we extract a larger patch containing 3 slots (the considered slot and its two neighbors). In order to reduce the variations from perspective distortion, parking displacement, and vehicle size, a spatial transformer network (STN) [15] is introduced and trained to transform the extended patch. In our system, we apply a 2D affine transformation with 6 parameters as our geometry model. Given a testing patch, the trained STN would adaptively generate the affine model to scale, rotate and shift the patch accordingly.

Ideally, the transferred patch is supposed to be highly discriminative for status classification. To meet the goal, we integrate the STN into our CNN-based space classification framework as shown in Fig. 3. Since STN is differentiable, we could train the STN together with our CNN based on backpropagation without any extra training supervision. It is also because the objective function for network training is to optimize space classification, our STN has the flexibility to transform the input patch and determine the suitable image region useful for space classification automatically. In detail, as described in Fig. 4, our STN includes three components:

- **Localisation Network**: Two convolutional layers are used in this network. Each one is followed by a pooling layer and a ReLU nonlinear function. Later on, a fully connected layer and a final regression layer are linked to produce the parameters θ of the transformation T_θ (6 outputs in our case for a 2D affine transformation).
- **Grid generator**: The generator bases on T_θ to warp the regular grid $G=\{G_i\}$ of the output patch X_θ to the source coordinate. Here, $G=\{G_i\}$ is the collection of output pixel locations of X_θ. $G_i = (x_i^t, y_i^t)$ is the coordinate of the i^{th} sample. The pixel-wise transformation T_θ is defined in equation (1).

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = T_\theta(G_i) = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}, \quad (1)$$

where (x_i^s, y_i^s) is the corresponding source coordinate of the i^{th} location sample in the input image.

- **Sampler**: This step interpolates the output patch X_θ at the location samples $\{G_i\}$. As shown in equation (2), by referring the intensity of the input patch, a bilinear interpolation kernel can be used to estimate the output patch value X_{θ_i} at the i^{th} location (x_i^t, y_i^t).

$$X_{\theta_i} = \sum_n^H \sum_m^W X_{nm} \cdot \max(0,1-|x_i^s - m|) \cdot \max(0,1-|y_i^s - n|). \quad (2)$$

In (2), X_{nm} is the image intensity at the pixel location (m,n) in the input patch with image height H and width W.

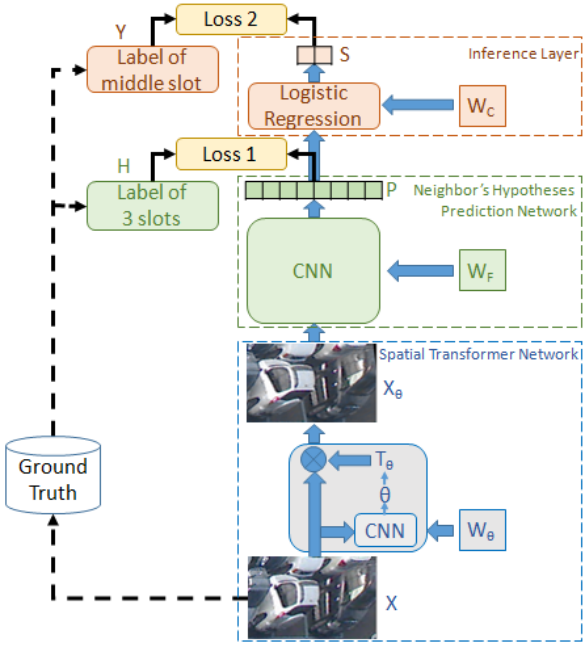


Figure 3. The proposed framework

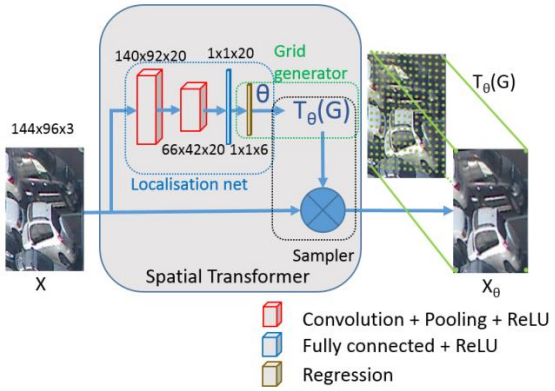


Figure 4. Spatial Transformer Network

2.2. Neighbor's Hypotheses Prediction Network

After obtaining the suitable patches from STN, we can apply CNN directly to classify the status of the considered space (the middle one). However, under the naïve idea, the STN will be trained to focus on extracting the image area around the considered space only. In the occlusion case, the loss of information from neighboring spaces would lead to severely false classification. To solve the inter-occlusion problem, we propose to design a deep network to firstly predict the status of a 3-space unit followed by an inference layer.

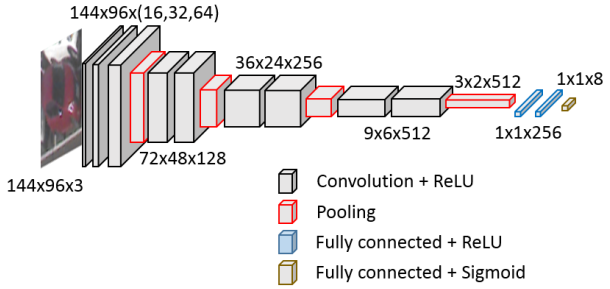


Figure 5. Neighbor's Hypotheses Prediction Network

The structure of NHPN is shown in Fig. 5, including 16 layers (9 convolutional layers, 4 pooling layers and 3 fully connected layers). It is designed with three properties:

- The network is determined by many stages. Different stages are separated by a pooling layer. With this design, we hope that the network can learn more semantic features by many convolutional layers before pooling.
- The network down-samples the input image to a small size (3x2 in our case) before applying fully connected layers for classification. This enables NHPN to learn the high-level feature patterns by a larger receptive field, and reduces the number of parameters. Hence, it also reduces the number of necessary training data.
- As the receptive field grows, we increase the number of kernels. We find that in the early layers, the network will learn the basic components such as object edges or the arrangements of edges [16]. Hence, many kernels are redundant in this stage. When the receptive field grows in the later layers, new and more complex patterns are emerging. To increase the discrimination of the network, we increase the number of kernels.

NHPN would produce 8 outputs, which are the scores of the status hypotheses of a 3-space unit, {(0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,0,1), (1,1,0), (1,1,1)}. As shown in Fig. 5, we use a sigmoid function for the final layer in order to have probability outputs corresponding to the 8 status hypotheses. Here, we introduce the first object function L_1 for training. The training goal is to learn the NHPN parameters W_F and the STN parameters W_θ to minimize the cross-entropy loss function L_1 in equation (3).

$$L_1(W_F, W_\theta, X_t, H) = -\frac{1}{8N}\sum_{n=1}^{N}\sum_{k=1}^{8}[h_n^k log p_n^k + (1-h_n^k)\log(1-p_n^k)]. \quad (3)$$

In (3), $\boldsymbol{X_t}$ is the input training set and $\boldsymbol{H}$ is the corresponding status labels. $\boldsymbol{N}$ is the sample number. $h_n^k \in [0,1]$ is the label of the k^{th} hypothesis of the n^{th} sample, where k=1~8. If the n^{th} sample belongs to hypothesis k, h_n^k would be 1, otherwise it would be 0. p_n^k is the status probability output of the NHPN. It is calculated by equation (4).

$$p = G_{W_F}\left(G_{W_\theta}(X)\right) = G_{W_F}(X_\theta), \quad (4)$$

where, X and X_θ are the input patch and the transformed patch by STN. $G_{W_F}(.)$ and $G_{W_\theta}(.)$ represent the functions of NHPN and STN.

2.3. Inference layer

In order to infer the status of the middle space, we build a 2-class logistic regression model on the top of NHPN. The input consists of the probabilities $p^k|_{k=1\sim8}$ of the 8 hypothesis and an extra bias node p^0=1. The output has two nodes, S^1 and S^0, indicating the occupied and vacant probabilities of the middle space. To train the model parameter W_C, as defined in (5), we introduce the second objective function L_2.

$$L_2(W_C, P, Y) = -\frac{1}{N}\sum_{n=1}^{N} y_n \log(S_n^1) + (1-y_n)\log(S_n^0). \quad (5)$$

$\boldsymbol{P}$ is the input training set and $\boldsymbol{Y}$ is the corresponding status labels. $y_n \in [0,1]$ is the label of the middle space of the n^{th} 3-space unit. $\boldsymbol{N}$ is the sample number. With trained W_C, S_n^1 and S_n^0 could be calculated by

$$S_n^i = \frac{\exp(p_n \cdot W_C^i)}{\sum_{k=0}^{1}\exp(p_n \cdot W_C^k)}, \quad (6)$$

where, p_n=($p_n^0, p_n^1 \cdots, p_n^8$) and i=0~1.

The joint objective function for the whole network (Fig. 3) can be formulated as $L = L_2 + \lambda \cdot L_1$. λ is the weight for L_1, which is determined empirically. Back-propagation method is used to train the whole network and determine the STN parameter (W_θ), the NHPN parameter (W_F), and the logistic regression parameters (W_C) simultaneously.

3. RESULTS AND DISCUSSIONS

In our experiment, we evaluate our system in an outdoor parking lot with a challenging viewing angle as shown in Figure 6. The training and testing datasets are collected within 1 month. Training images are captured and stored for every minute within the first 15 days. Totally, 8277 images are used. Each image has 71 parking spaces. The 525 testing images are captured and stored for every 20 minutes within the other 15 days. These datasets, results, and real-time demonstration can be found in our website [17]. We use Caffe [18] package for implementation. The network was trained using a learning rate of 0.001 for the first 40k iterations and 0.01 for the next 5k. The input patches are warped to a fixed size of 144x96.

In the experiments, we compare our method with CNN-based methods named as CNN_1, CNN_2, $CNN\text{-}STN_1$, and $CNN\text{-}STN_2$. CNN_1 is our baseline system. The input patch only contains the considered space. Instead, the input of CNN_2 covers 3 spaces. Both networks are trained using only the loss function $L_2(.)$. $CNN\text{-}STN_1$ uses STN for patch

transformation and uses $L_1(.)$ for training. CNN-STN$_2$ has similar network like CNN-STN$_1$ but trained by $L_2(.)$. Besides CNN-based methods, we also compare with Huang's work [3]. We evaluate our system by false positive rate (FPR), false negative rate (FNR), and accuracy (ACC). The evaluation results are listed in Table I and the Receiver Operating Characteristic (ROC) curves are shown in Fig. 7.

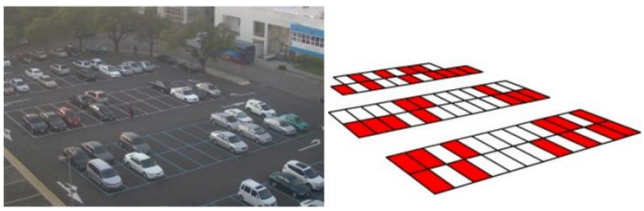


Figure 6. Testing environment and detection result

According to Table I, the proposed method gives better performance because of the introduction of NHPN, STN, and joint training. When the neighboring information is introduced (CNN$_2$), the performance is boosted nearly 2% compared with CNN$_1$. Once STN is added, the performance goes to 99.01%. With joint training, we can achieve the final ACC, 99.25%. Note that CNN-STN$_2$ is trained with $L_2(.)$, so the STN tends to focus just on the middle slot. In contrast, both of the proposed method and CNN-STN$_1$ consider the neighboring information by using loss function $L_1(.)$. Hence, STN keeps the information of all three spaces. However, CNN-STN$_1$ tries to crop a 3-space unit as snug as possible. With the effect of $L_2(.)$, the proposed method would select a wider view of a 3-space unit.

Table II shows some cases to illustrate the effects of NHPN, STN, and joint training. In case (a), we find STN can help to extract the discriminative patch for correct status detection even if the middle car shifts from the right position. Case (b) gives an example under lighting variation. Case (c) shows the detection under low image contrast. An occlusion situation is presented in Case (d). We find CNN-STN$_1$ and our method can handle these cases well. Instead, CNN-STN$_2$ focuses mainly on the target space without considering the neighbors and thereby produces false detection. Moreover, compared with CNN-STN$_1$, our method introduces a multi-task loss function for training and allows the network to consider the target and neighbor spaces simultaneously. This property makes the network more robust such as in Case (e).

Table I. Performance Comparison

	ACC	FPR	FNR
Huang's work [3]	98.44%	0.0128	0.0173
CNN$_1$	96.78%	0.0666	0.0136
CNN$_2$	98.71%	0.0129	0.0129
CNN-STN$_1$	99.01%	0.0057	0.0124
CNN-STN$_2$	98.98%	0.0057	0.0129
Proposed method	**99.25%**	**0.0029**	**0.0103**

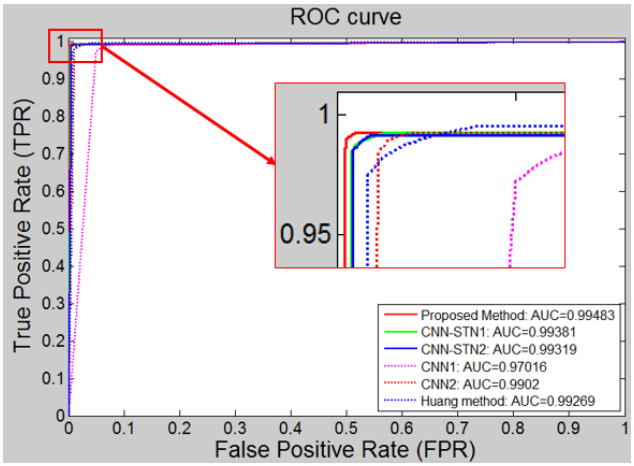
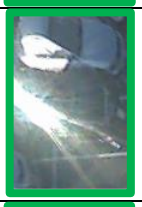
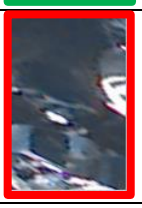
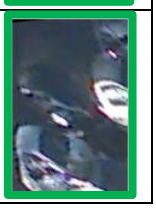


Figure 7. The ROC curve comparison

Table II. Transformed Patches from different methods and the detection results. Red boxes indicate false detection and green boxes mean correct detection.

	Input	CNN-STN$_1$	CNN-STN$_2$	Our method
(a)				
(b)				
(c)				
(d)				
(e)				

4. REFERENCES

[1] C.C. Huang, Y.S. Tai, and S.J. Wang, "Vacant Parking Space Detection Based On Plane-Based Bayesian Hierarchical Framework," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 23, Issue 9, pp. 1598–1610, Sept. 2013.

[2] C.C. Huang, Hoang Tran Vu, "A Multi-Layer Discriminative Framework For Parking Space Detection," *IEEE International Workshop on Machine Learning for Signal Processing*, Sept. 2015.

[3] C.C. Huang, Hoang Tran Vu, "Vacant Parking Space Detection based on a Multi-layer Inference Framework," *IEEE Transactions on Circuit and Systems for Video Technology*, May 2016.

[4] Paulo Almeida, Luiz S. Oliveira, Eunelson Silva Jr., Alceu Britto Jr., and Alessandro Koerich, "Parking Space Detection using Textural Descriptors," *in IEEE International Conference on Systems, Man, and Cybetnetics*, 2013.

[5] R.J. López-Sastre, P. Gil Jimenez, F.J. Acevedo, and S. Maldonado Bascon, "Computer Algebra Algorithms Applied to Computer Vision in a Parking Management System," *IEEE International Symposium on Industrial Electronics*, 2007.

[6] Wand Lixia and Jiang Dalin, "A method of Parking space detection based on image segmentation and LBP*," International Conference on Multimedia Information Networking and Security*, pp. 229-232, Nov. 2012.

[7] Diana Delibaltov, Wencheng Wu, Robert P. Loce, and Edgar A. Bernal , "Parking Lot Occupancy Determination from Lamp-post Camera Images," *in Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems*, 6-9 October, 2013.

[8] M. Tschentscher and M. Neuhausen, "Video-based parking-space detection," *in Proceedings of the Forum Bauinformatik*, 2012, pp. 159-166.

[9] N.True, "Vacant parking space detection in static image," *Projects in Vision & Learning*, University of California, 2007.

[10] Harish Bhaskar, Naoufel Werghi, and Saeed AL-Mansoori, "Rectangular Empty Parking Space Detection using SIFT based Classification," *in Proceedings of the Sixth International Conference on Computer Vision Theory and Applications*, 5-7 March, 2011.

[11] Q. Wu, C. C. Huang, S. Y. Wang, W. C. Chiu, and T. H. Chen, "Robust Parking Space Detection Considering Inter-Space Correlation," *IEEE International Conference on Multimedia and Expo*, pp. 659-662, 2007.

[12] H. Ichihashi, T. Katada, M. Fujiyoshi, A. Notsu, and K. Honda, "Improvement in the performance of camera based vehicle detector for parking lot," *in Proc. IEEE Int. Conf. Fuzzy Syst.*, pp. 1950–1956, 2010.

[13] Li Wang and Dennis Sng, "Deep Learning Algorithms with Applications to Video Analytics for A Smart City: A Survey," *arXiv:1512.03131 [cs.CV]*, Dec. 2015.

[14] S. Valipour, M. Siam, E. Stroulia, and M. Jagersand, "Parking Stall Vacancy Indicator System Based on Deep Convolutional Neural Networks," *arXiv:1606.09367 [cs.CV]*, June 2016.

[15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu, "Spatial transformer networks," *in Advances in Neural Information Processing Systems*, pages 2017‑2025, 2015.

[16] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature 521,* pp. 436–444, 2015.

[17] C. C. Huang. (2015). *Huang's Projects*. [Online]. Available at http://acm.ee.ccu.edu.tw:2017.

[18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding," *In Proceedings of the ACM International Conference on Multimedia*, pages 675-678, 2014.