Research On Cone Bucket Target Detection Based On Improved Faster R-CNN Deep Network

Qinghe Liu¹*, Weihao Chen¹, Lijun Zhao¹ and Conggan Ma¹

¹School Of Automotive Engineering, Harbin Institute of Technology(Weihai), Weihai, Shandong, 264200, China

*Corresponding author. e-mail: qingheliu@hitwh.edu.cn

Abstract. To solve the problem of detecting the cone bucket target on both sides of the FSAC racing car, a new target detection method based on Faster R-CNN is proposed. The method uses a deep residual network as a feature extraction network to fully extract sample data features. In the Region Proposal Network (RPN), the anchor boxes are divided into finer parts to enhance the detection effect on the small target cones. Optimize the Non-maximum Suppression (NMS) algorithm to extract the proposal box and reduce the missed detection rate of the adjacent cone bucket target. The target detection network model is obtained by training on the self-made cone bucket dataset. The experimental results show that the improved algorithm based on Faster R-CNN is robust to the detection of cone bucket targets.

1. Introduction
As one of the important methods for the perception of the environment of driverless cars, visual sensing is to obtain the surrounding environmental information through machine vision, and filter the information through image processing to complete the perception of the surrounding environment. Target detection is to simulate the human visual mechanism, obtain the surrounding environment information image by the visual sensor and then use the computer to calculate the target in the region of interest, so that we can determine the category and location of the target. Driverless cars need to identify the information of pedestrians, traffic signs, traffic lights and obstacles in the real scene environment, and make the next judgment and processing according to the acquired information. The importance of target detection is self-evident and is of great significance for the development and advancement of driverless cars.

The Region-based Convolutional Neutral Network (R-CNN) algorithm is rapidly developing and has been widely used in the field of target detection. R-CNN [1] combines the region proposals extraction method with the convolutional neural network. The first method uses the selective search algorithm [2] instead of the sliding window to segment the image, which greatly improves the efficiency of region proposals extraction. The network structure of SPPNet [3] introduces the method of spatial pyramid pooling, which effectively solves the problem about fixed input size in R-CNN and improves the speed and accuracy of detection. Fast R-CNN [4] draws on the idea of spatial pyramid pooling in SPPNet and proposes Region of Interest pooling (ROI pooling) to improve the problem of feature extraction repetitive calculation in R-CNN, however it still uses Selective Search algorithm to candidate region proposals extraction, which takes a lot of time. In response to this problem, Faster RCNN [5] proposes Region Proposal Network (RPN) to extract region proposals, and combines RPN network and Fast R-CNN through sharing convolutional layers to further improve detection performance and achieve end-to-end detection.
In this paper, by studying and improving the Faster R-CNN algorithm, the idea of deep convolutional neural network is used to solve the problem of small cone bucket target recognition and localization on the sides of FSAC racing car, which provides an efficient method for the detection of small targets.

2. Improve the Faster R-CNN network model

Faster R-CNN can be regarded as a combination of Fast R-CNN and RPN network. The overall detection framework includes the following four aspects:

- **Feature extraction:** As a region-based target detection algorithm, it is first necessary to perform feature extraction of images through a convolution layer, which is used for subsequent RPN layers and fully connected layers;
- **RPN:** The RPN network is used to generate anchor boxes. This layer uses the softmax function to determine whether the anchor boxes contain objects, that is, to distinguish whether it belongs to the foreground or background. And then use the bounding box regression to correct the position of the anchors;
- **ROI pooling:** According to the acquired feature map and anchor boxes, The region proposals are extracted after synthesizing the information, and the subsequent fully connected layer is used to determine the target category;
- **Classification and Regression:** The extracted region proposals is used to multi-class classification, and the bounding box regression is used again to correct the position of the region proposals to obtain the final precise position.

2.1. Feature extraction

For the feature extraction part, the basic network such as VGG and GoogleNet is usually used for feature extraction. Whether the feature extraction is sufficient or not is directly related to whether the neural network can learn the nature of the data, which has a great impact on the final effect of the model. In computer vision, the "rank" of features increases with the depth of the network. Studies have shown that the depth of the network is an important factor in achieving good results. However, vanishing gradient and exploding gradient become the deep network obstacle of training, which leads to failure of convergence.

In order to extract sufficient features to solve the above mentioned problems, the convolutional layer of the feature extraction part uses a deep residual network. It allows the network to deepen as much as possible, which introduces a completely new structure as shown in Figure 1.

![Deep residual network](image)

Figure 1. Deep residual network.

It is difficult for deep convolutional layers to fit a complex non-linear function $H(x)$, but it is easier for neural network to approach zero than to approach a nonlinear function, so we consider fitting a residual $F(x) = H(x) - x$, the way to achieve this is to direct the input to the output. From another point of view, it is the fusion of low-level features and high-level features. In the unit residual model, its output can be defined as:

$$y = F(x, \{ W_i \}) + x$$

(1)
In the formula, $x$ is the input of the model and $F(x, \{W_i\})$ represents the mapping to be fitted by the network. When you need to change the input and output dimensions (such as changing the number of channels), you can make a linear transformation $W_s$ to $x$ in the shortcut, as shown in the following equation:

$$y = F(x, \{W_i\}) + W_s x$$  \hspace{1cm} (2)

Using the unit residual model, you can construct a deep residual network with any number of layers, and design a suitable network structure according to the task requirements. The relative depth of the network generally depends on the complexity of the task. We use a 50-layer deep residual network as the feature extraction layer.

### 2.2. Region Proposal Network

The RPN network is a Fully Convolutional Network (FCN), and its network structure is shown in Figure 2. The output of conv4 is taken as the input of the RPN network, and the $n \times n$ sliding window is used to process and fuse the surrounding spatial information on the feature map, then we can get the 256-dimensional feature vector. Each feature point corresponds to $k$ kinds of anchor boxes as the initial detection boxes. Each anchor needs to distinguish between foreground and background, so each classification layer (cls) transformed from 256-dimensional feature vector has 2k outputs. Each anchor corresponds to 4 offsets $(x, y, w, h)$, so the regression layer (res) has 4k outputs. For a feature map of size $W \times H$, the center of each anchor coincides with the center of the sliding window, and its size is determined by the scale and ratio relative to the reference box, for a total of $k \times W \times H$ anchors.

![Figure 2. RPN network structure.](image)

The basic size of anchor is $16 \times 16$, and the size of each anchor (if the ratio is 1, the size of the anchor is $256 \times 256$) is much larger than the area of the cone bucket target. The proportion of the target in the anchor is very small, which increases the positioning error. The anchor is wrongly identified as a positive sample, which makes the model difficult to converge.

In this paper, according to the size of the cone bucket target in the training set, we improve the scale and ratios of anchor, the specific parameters are shown in Table 1. The traditional Faster R-CNN removes the small region proposals and the detection of small targets affects its performance. By adding smaller scales and introducing smaller ratios, the accuracy of small targets detection is improved.

| anchor     | number |
|------------|--------|
| base_size  | 16*16  | 1     |
| Original ratios | [0.5,1,2] | 3     |
| Original scale     | [8,16,32] | 3     |
| New ratios   | [0.3,0.4,0.5,1,1.5,2] | 6     |
| New scale    | [4,8,12,16] | 4     |

The total loss function of the RPN network can be defined as
\[
L(p_i(t_i)) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)
\]

(3)

Where \(i\) represents the \(i\)th anchor. \(p_i^* = 1\) represents that the anchor is a positive sample, and if it is a negative sample, then \(p_i^* = 0\). \(t_i^*\) represents the coordinate offset of the prediction box relative to the anchor. \(t_i^*\) represents the offset of the true box relative to the anchor. \(L_{cls}\) represents classification loss and is defined as

\[
L_{cls}(p_i, p_i^*) = -\log[p_i^*(1 - p_i) + (1 - p_i^*)p_i]
\]

(4)

\(L_{reg}\) represents the regression loss and is defined as

\[
L_{reg}(t_i, t_i^*) = R(t_i - t_i^*)
\]

(5)

Where \(R\) represents the \(\text{Smooth}_{L1}\) loss function and is defined as

\[
\text{Smooth}_{L1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}
\]

(6)

The \(N_{cls}\), \(N_{reg}\) and \(\lambda\) parameters are used for normalization of balanced classification and bounding box regression.

3. Experiment and analysis

3.1. Test environment configuration and data set

Experiment with the Tensorflow deep learning framework in the Linux environment. The specific configuration is: Ubuntu16.04, GEFORCE GTX10180Ti, NVIDIA cuDNN 6.0 and CUDA 8.0. We use a self-made cone bucket dataset. In order to enable deep convolutional neural networks to learn more rich features and enhance the robustness and generalization capabilities of the model. We collected 9949 samples from different scenes, weather and environments. According to the format of the VOC2007 data set, the ratio of the training-validation set (trainval) and the test set (test) is divided into 7:3. After 70,000 iterations of training, the final loss curve is shown in Figure 3. On the whole, after about 50,000 iterations of training, the network model tends to converge on the training set and the validation set, and the loss value is stable at around 0.5. It shows that the network model can fit the sample data well.

![Figure 3. loss curve.](image-url)
3.2. Comparative analysis of detection results

The original Faster R-CNN is referred to as the original network, and the improved Faster R-CNN is referred to as the improved network. **Figure 4** and **Figure 5** corresponds to the comparison of the detection results between the original network and the improved network in simulated track scene.

![Figure 4. Original network.](image)

![Figure 5. Improved network.](image)

In the simulated track scene, the original network can only detect the cone bucket target closer, the smaller cone bucket target in the far distance can not be detected, however, the improved network can detect the targets far away. In terms of the accuracy of detection, both classification and localization ensure very high accuracy, only two red cone buckets with a high degree of coincidence on the right side are mistakenly detected as one red cone bucket, the results show that the improved network has stronger feature extraction ability and the use of multi-scale anchors can improve the detection of small targets.

4. Conclusion

In this paper, the feature extraction layer of the network is replaced by a deep residual network based on FasterR-CNN, and we re-select the anchor in the RPN. The self-made cone bucket data set is used for training, then we get the improved Faster R-CNN detection model. The method uses the deep residual network to perform shared feature extraction, which enhances the feature extraction ability of the network and improves the detection accuracy of the model on the cone bucket target. The re-select anchor can identify smaller cone bucket targets and reduce the missed detection rate of the model.

It is concluded from the experimental results and analysis that the improved method proposed in this paper improves the detection accuracy of cone bucket targets and has strong robustness. However, the model is time-consuming in the process of extracting features. Under the premise of ensuring the detection effect, how to improve the detection efficiency still needs further study.
Acknowledgement
This paper is funded by Science and Technology Development Plan of Weihai.

References
[1] Girshick R, Donahue J, Darrell T, et al. (2014). Rich feature hierarchies for accurate object
detection and semantic segmentation In: Proceedings of 2014 IEEE Conference on Computer
Vision and Pattern Recognition. Columbus. pp. 580-587.
[2] Uijlings J R R, van de Sande K E A, Gevers T, et al. (2013) Selective search for object recognition. 
International journal of computer vision., 104(2): 51–59.
[3] He K, Zhang X, Ren S, et al. (2015) Spatial Pyramid Pooling in Deep Convolutional Networks
for Visual Recognition. IEEE Transactions on Pattern Analysis & Machine Intelligence.,
37(9): 1904–1916.
[4] Girshick R. (2015). Fast R-CNN In: Proceedings of 2015 IEEE International Conference on
Computer Vision. Santiago. pp. 1440-1448.
[5] Ren S, Girshick R, Girshick R, et al. (2015) Faster R-CNN: Towards Real-Time Object Detection
with Region Proposal Networks. IEEE Transactions on Pattern Analysis & Machine
Intelligence., 39(6): 1137–1149.