# Moving Object Detection Based on Self-adaptive Contour Extraction

Xin Shi[1,2], Tao Xue[1,2,3], and Xueqing Zhao[1,2,3](✉)

[1] School of Computer Science, Xi'an Polytechnic University, Xi'an, Shaanxi, China
[2] Shaanxi Key Laboratory of Clothing Intelligence, School of Computer Science, Xi'an Polytechnic University, Xi'an 710048, China
[3] National and Local Joint Engineering Research Center for Advanced Networking and Intelligent Information Service, Xi'an Polytechnic University, Xi'an 710048, China
zhaoxueqing@xpu.edu.cn

**Abstract.** Object detection of moving targets requires both accuracy and real-time performance. In this paper, we propose a contour extraction prior to convolutional neural network to extract more salient features and use region proposal network to generate candidate regions. Afterwards, the feature maps and proposal regions are inputed to ROI pooling layer followed with some fully connected layers to classify objects and regress bounding box. Simulation experiments show that our method is effective in improving detection accuracy by testing on the dataset with 11 categories of moving targets.

**Keywords:** Moving object detection · Contour extraction · Region proposal network

## 1 Introduction

As one of the core problems in the field of computer vision, object detection refers to the process of identifying the regions of interest, determining their categories and locating their positions [1–3]. The object detection tasks can be decomposed into two subproblems towards multiple objects; namely, objects' locating and objects' classifying [4,5]. Due to the different exteriors, shapes and postures, the interference of illumination and shades, object detection is always one of the most challenging problems in the field of computer vision. Deep learning methods, represented by neural networks [6,7], have achieved excellent performance in the field of object recognition recently, which attracts more and more researchers to devote to improving neural networks and building new computing models to deal with the problem.

Supported by the National Natural Science Foundation of China under Grant No.61806160 and Shaanxi Association for Science and Technology of Colleges and Universities Youth Talent Development Program, No. 20190112 and the Youth Innovation Team of Shaanxi Universities.

The proposal of convolutional based neural network named AlexNet [8] in 2012 has successfully demonstrated the effectiveness and potential of deep learning in the field of computer vision. Afterwards, an increasing number of deep neural networks have emerged and promoted the development of computer vision rapidly. In terms of object detection tasks, current algorithms and networks can be classified into two categories, the two-stage algorithms and the one-stage algorithms. The former ones require the generation of region proposal prior to the object detection, RCNN [9], Fast-RCNN [10] and Faster-RCNN [11] are some of the typical representations of them, these algorithms achieve relatively better accuracy. The latter ones are represented by SSD [12], RetinaNet [13] and YOLO series [14], which implement objects' locating and classifying directly in a single network so that they are capable of improving efficiency.

With regard to dynamic objects' detection in video series, it not only requires for the accuracy and precision of detection, but also expects a better capability in terms of real-time performance. However, either current one-stage algorithms or two-stage algorithms, their abilities of striking a balance between efficiency and accuracy are still less than satisfactory. Therefore, in this paper, we attempt to integrate contour extraction into deep learning based object detection neural networks with an expectation of enabling the application on dynamic objects' detection. Specifically, we extract contour information prior to the convolutional neural network (CNN for short) so as to acquire more distinct and salient feature maps. Besides, we substitute region proposal network (RPN for short) [11] for selective search (SS for short) to generate region proposal which greatly improves the generation speed of bounding box. Literature reviews have demonstrated that the replacement of RPN network has increased the speed of region proposal generation from 2 s to 10 milliseconds, which enables the requirement of end-to-end object detection. Experiment results in this paper further show that, our method effectively improves the accuracy of moving object detection in video series.

The rest of paper is organized as follows. The framework and technical details of our method are introduced in Sect. 2. Experiments and discussions are presented in Sect. 3. Lastly, Sect. 4 gives the conclusion of this paper.

## 2 Methods

In this section, we firstly introduce the network architecture of our method, followed by some core implementation technical details.

### 2.1 Framework Description

Features are always the most important factors of computer vision tasks either in traditional methods or in deep learning based methods. The more salient and distinct the features are extracted, the more accurate the task is achieved. Hence, in our method, we firstly extract the contours of input images because the contour information ignores the effect of background as well as some noise interference inside the object, which is capable of protruding objects to some extent. Afterwards, the contour images are inputed to convolution neural network to generate feature maps; then comes to the RPN network to estimate and generate region proposals. ROI pooling receives the feature maps as well as the region proposals to convert proposals of different dimensions to outputs with fixed length with the purpose of adapting to successive fully connected layers. The last step composes of two branches, one is to predict the final bounding box of the objects, the other is the classification of objects. The overall flowchart of our method is illustrated as Fig. 1, the technical details of contour extraction and the main idea RPN network are introduced later.

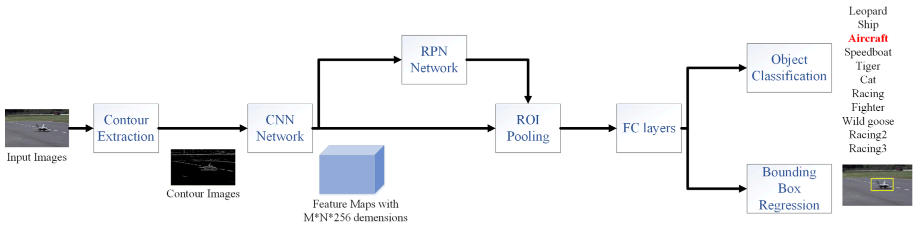


**Fig. 1.** The overall flowchart of the method in this paper.

## 2.2 Contour Extraction

Before inputting source images to deep neural network, we firstly carry on a pre-processing to the images by extracting their contour information.

As one of the most dominant edge detection algorithms, Canny [15] has been widely used with the help of its better signal to noise ratio and detection accuracy. However, some parameters in Canny require manual setting, such as variance of Gaussian filter and the two thresholds of binarization, which results in a poor adaptability and difficulty in practical application. Therefore, to improve the adaptability of Canny, we substitute self-adaptive smoothing filter for Gaussian filter and use Otsu [16] algorithm to generate the low and high threshold according to the distribution of gray-scale pixels automatically. In this way, the noises and some pseudo edges are eliminated so that the contours can be extracted with no artificial intervention. The flowchart of the self-adaptive contour extraction process is shown in Fig. 2.

To be more precise, the basic idea of self-adaptive filter lies in the iterative convolution between original image and a small average weighted template, the weighted coefficients of each pixel are changed adaptively during each iteration. The filtered image $f^{(n+1)}(x, y)$ after $n$ iterations is defined as Eq. (1),

$$f^{(n+1)}(x,y) = \frac{\sum_{i=-1}^{+1}\sum_{j=-1}^{+1} f^{(n)}(x+i,y+j)w^{(n)}(x+i,y+j)}{\sum_{i=-1}^{+1}\sum_{j=-1}^{+1} w^{(n)}(x+i,y+j)} \quad (1)$$

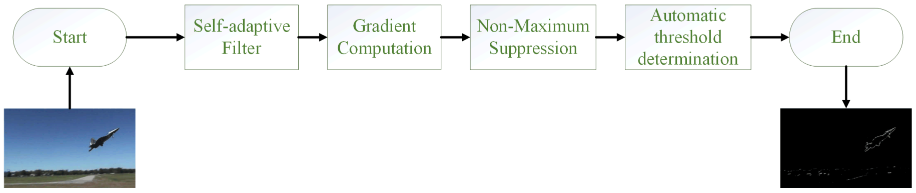


**Fig. 2.** The flowchart of the self-adaptive contour extraction.

where $w^{(n)}$ is the weighted coefficients in the $n$th iteration of each pixel defined as Eq. (2),

$$w^{(n)}(x,y) = exp[-\frac{G_x^2(x,y)+G_y^2(x,y)}{2k^2}] \tag{2}$$

$G_x^2(x,y)$ and $G_y^2(x,y)$ are gradient components as shown in Eq. (3) and Eq. (4) respectively. $k$ is a parameter to determine the size of convolutional template which is set to be 10 in this paper.

$$G_x(x,y) = \frac{1}{2}[f(x+1,y) - f(x,y)] \tag{3}$$

$$G_y(x,y) = \frac{1}{2}[f(x,y+1) - f(x,y)] \tag{4}$$

In terms of the threshold selection, the threshold is determined as the one which enables the maximum between-cluster variance of foreground and background. The between-cluster variance is defined as Eq. (5),

$$\sigma_B^2 = \frac{[m_G P_1(k) - m(k)]^2}{P_1(k)[1-P_1(k)]} \tag{5}$$

where $m_G$ is the average grayscale of the image defined as Eq. (6), $L$ is the largest grayscale and $p_i$ refers to the probability of grayscale $i$. $m(k)$ is the average grayscale from grayscale 0 to $k$, as shown in Eq. (7). $P_1(k)$ is the probability that a pixel is in the range of 0 and $k$, as shown in Eq. (8),

$$m_G = \sum_{i=0}^{L-1} ip_i \tag{6}$$

$$m(k) = \sum_{i=0}^{k} ip_i \tag{7}$$

$$P_1(k) = \sum_{i=0}^{k} p_i \tag{8}$$

The value of $k$ which enables $\sigma_B^2$ to be maximized is selected as the optimal high threshold $T_H$ in Canny and the low threshold $T_L$is accordingly set to be $\frac{1}{2}k$.

## 2.3 Region Proposal Network

As the main innovation of Fast-RCNN, the proposal of RPN integrates the region proposal process into the neural network architecture and realizes an end-to-end object detection model, which increases the detection efficiency by reducing the redundant region generation and integrating all computations onto GPUs. The purpose of RPN is to extract proposal regions through neural network instead of the traditional selective search based methods [17]. The input of RPN is the feature map extracted from original images by backbone convolution neural networks such as VGG16 [18], the output of RPN can be used to determine possible region proposal. The basic idea of RPN can be summarized as follows.

After feature extracting though some convolution and pooling layers, a feature map with the dimension of $M \times N \times 256$ is acquired, which is taken as the input of RPN network. To better fuse the neighborhood information so as to make the features more robust, a $3 \times 3$ convolution is applied to the feature maps.

As the feature maps are extracted through convolution and pooling, it is not hard to comprehend that each point on the feature map can be mapped to a region on original image, by adding some parameters and restrictions, RPN determines $k$ kinds of mapping for each point on feature maps and renamed points on feature maps as anchors. Therefore, two branches are followed after the $3 \times 3$ convolution layer where each branch is a fully connected layer (FC layer for short) with different output dimensions. The first branch is used to determine whether the mapped region from anchors to original image contains object or not, the output is a vector with the size of $M \times N \times 2k$ and is further reshaped into (M, N, k, 2) where the last dimension contains two numbers to show the scores with or without object. The second branch is used to determine the offset from anchors on feature map to original image, for any possibility of $k$ kinds of mappings, it has 4 coordinates which are center coordinates (x, y) and the height-width size of the rectangular mapping (h, w). Therefore, the output of the second branch is a vector with the size of $M \times N \times 4k$.

To sum up, the output of the RPN network is two vectors with the dimension of (M, N, k, 2) and (M, N, k, 4), respectively, which also means that for each feature map with the size of $M \times N$, RPN generates $M \times N \times k$ proposal regions primarily. This is still a large amount of proposal regions which is not only computational resources consuming but also inefficient. Therefore, a proposal layer is attached to further screen the candidate regions. It firstly sorts the $k$ candidate proposal regions according to their scores of having objects and selects the top $k'$ with highest scores. Secondly, it maps the $k'$ anchors to original image and compares with the ground truth by using non-maximum suppression [19] to determine the final region proposal results of RPN network.

## 3 Simulation Experiments

### 3.1 Dataset and Environmental Settings

We construct the dataset with 8650 images by extracting frames from 11 video series with moving object, the size of each image is $640 \times 320$ pixels, Fig. 3 shows some examples of dataset.

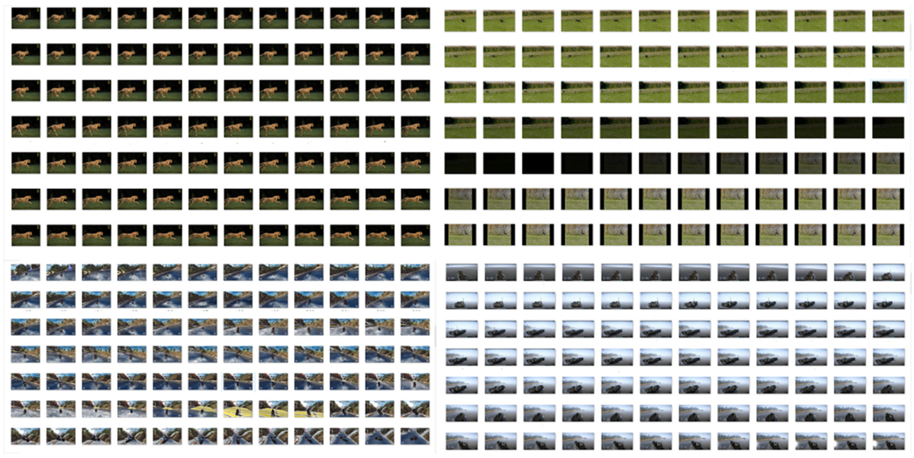


**Fig. 3.** Examples of the dataset.

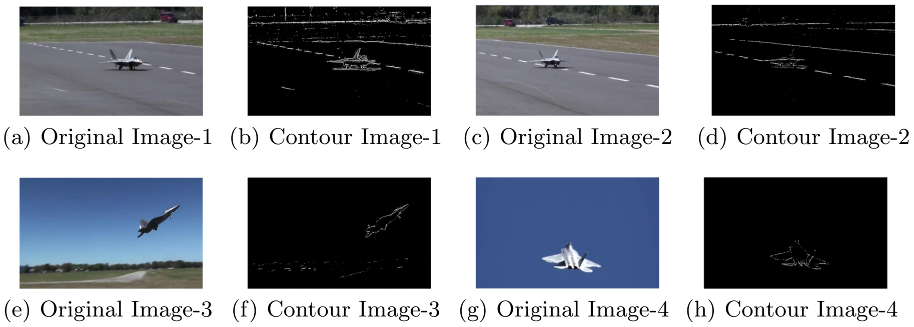
Our experiments are executed on the basis of Window 10 operation system with Intel(R) Core(TM) i5-6500 CPU with 4 GB RAM. The program is written in MATLAB under MATLAB R2018a. The parameters of the simulation experiments are shown in Table 1.

**Table 1.** The parameter settings in simulation experiments.

| Parameter names | Parameter values |
|---|---|
| Mini BatchSize | 5 |
| Initial LearnRate | 0.001 |
| MaxEpochs | 100 |
| Verbose frequency | 200 |

### 3.2 Results and Analysis

Figure 4 shows some examples after contour extraction, it is obviously that the objects in extracted images are salient and distinct.


(a) Original Image-1 (b) Contour Image-1 (c) Original Image-2 (d) Contour Image-2

(e) Original Image-3 (f) Contour Image-3 (g) Original Image-4 (h) Contour Image-4

**Fig. 4.** Some examples of original images and results after contour extraction. (a)(c)(e)(f) are examples of original images, (b)(d)(f)(h) are corresponding contour images.

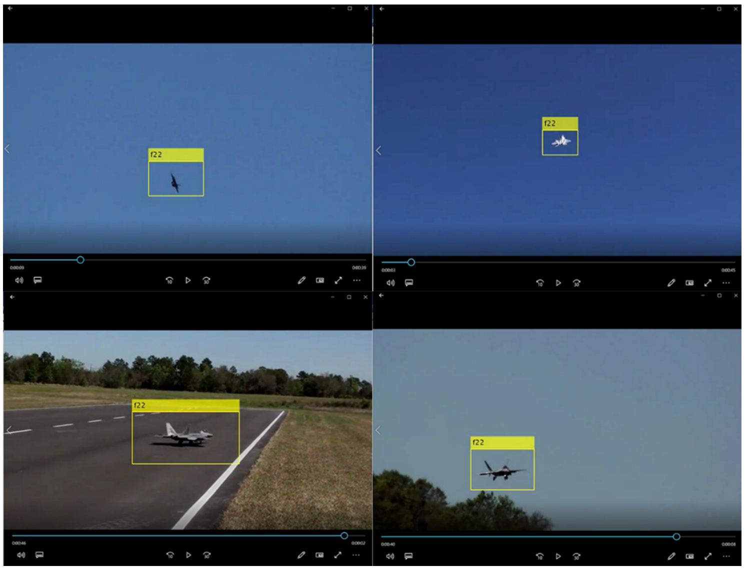


**Fig. 5.** Examples of the object detection results.

Figure 5 is the video screenshot of the testing videos, it can be seen that the moving targets have been detected and recognized successfully.

To validate the effectiveness of our method, we further compare it with Faster-RCNN on the dataset with 11 categories in terms of detection accuracy which is defined as F1-score in Eq. (9),

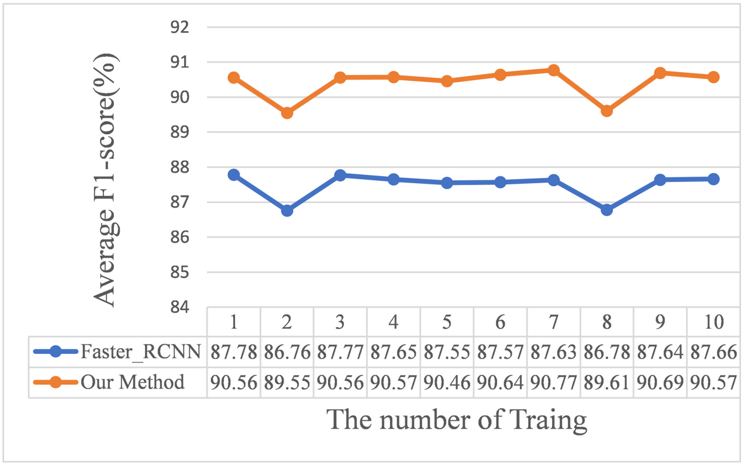
$$F1 - score = \frac{2 * P * R}{P + R} \tag{9}$$

where $P$ means precision rate defined as $P = TP/(TP + FP)$ and $R$ represents recall rate defined as $R = TP/(TP + FN)$. $TP, FP, FN$ show the relationship between predicted result and real result are true-positive, false-positive and false-negative, respectively. Table 2 illustrates the comparison results of our method and Faster-RCNN. It can be seen that the F1-scores of different categories increase by an average of 2 percent, the average accuracy rate is about 88.08% with some categories exceeding 90%. To further analyze the relationship between the content of videos and detection accuracy, we find that the accuracy rate is inversely proportional to the moving speed of objects. For example, the videos with slow moving objects such as cat and wild goose reach the higher accuracy while videos with leopard and speedboat get relatively lower accuracy.

**Table 2.** The $F1 - score$ comparison results of our method and Faster-RCNN in terms of 11 categories.

| Category name | Result of faster-RCNN (%) | Result of our method (%) |
|---|---|---|
| Leopard | 81.75 | 84.66 |
| Ship | 86.79 | 89.45 |
| Aircraft | 87.23 | 89.26 |
| Speedboat | 82.75 | 85.57 |
| Tiger | 83.81 | 86.66 |
| Cat | 89.45 | 91.64 |
| Racing | 87.67 | 89.47 |
| Fighter | 87.83 | 90.71 |
| Wild goose | 86.68 | 90.39 |
| Racing2 | 85.44 | 88.57 |
| Racing3 | 87.16 | 90.20 |

Moreover, we shuffle the dataset for ten times and divide the training and testing set in a ratio of 8:2 to further make the comparison between our method and Faster-RCNN in terms of average F1-scores, Fig. 6 shows the comparison results. The average accuracy of our method is about 2.5 percents higher than that of Faster-RCNN, which validates that our method is effective for the accuracy improvement.



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Faster_RCNN | 87.78 | 86.76 | 87.77 | 87.65 | 87.55 | 87.57 | 87.63 | 86.78 | 87.64 | 87.66 |
| Our Method | 90.56 | 89.55 | 90.56 | 90.57 | 90.46 | 90.64 | 90.77 | 89.61 | 90.69 | 90.57 |

**Fig. 6.** The comparison results of average F1-score between our method and Faster-RCNN for 10 iterations of training.

## 4 Conclusion

To balance between efficiency and accuracy of moving object detection, in this paper, we integrate contour extraction into deep learning based object detection neural network. Specifically, we extract contour information prior to the feature map generation of convolutional neural network (CNN for short) so as to acquire more distinct and salient feature maps. Besides, we use RPN network to generate region proposal which enables the requirement of end-to-end object detection. The simulation experiments show that our method increases the average F1-score by 2.5% than that of Faster-RCNN in terms of moving object detection by testing on the dataset with 11 categories. Our future work will concentrate on the improving the detection accuracy of fast moving objects and high resolution videos.

## References

1. Borji, A., Cheng, M.-M., Hou, Q., Jiang, H., Li, J.: Salient object detection: a survey. Comput. Visual Media **5**(2), 117–150 (2019). https://doi.org/10.1007/s41095-019-0149-9
2. Xiao, Y., et al.: A review of object detection based on deep learning. Multimedia Tools Appl. **79**(33), 23729–23791 (2020). https://doi.org/10.1007/s11042-020-08976-6
3. arXiv:1905.05055. https://arxiv.org/abs/1905.05055. Accessed 16 May 2019
4. Everingham, M., Eslami, S., Gool, L.V., et al.: The pascal visual object classes challenge: a retrospective. Int. J. Comput. Vis. **111**(1), 98–136 (2015)
5. Redmon, J., et al.: You only look once: unified, real-time object detection. In: Computer Vision & Pattern Recognition IEEE (2016)

6. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. Neural Comput. **29**(9), 2352–2449 (2017)
7. Gu, J., et al.: Recent advances in convolutional neural networks. Pattern Recogn. **77**, 354–377 (2015)
8. Krizhevsky, A., Sutskever, I., Hinton, G. E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, 1097–1105(2012)
9. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation, In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, pp. 580–587 (2014)
10. Girshick, R. : Fast R-CNN. In: Computer Science (2015)
11. Ren, S., He, K., Girshick, R., et al.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2017)
12. Liu, W., et al.: SSD: single shot MultiBox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
13. Lin, T.Y., Goyal, P., Girshick, R., et al.: Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. **42**(2), 318–327 (2017)
14. Redmon, J., Farhadi, A.: YOLOv3: an incremental improvement. arXiv e-prints(2018)
15. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) **8**(6), 679–698 (1986)
16. Otsu, N.: A thresholding selection method from gray-level histogram. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (2007)
17. Syed, H.: Selective search for object recognition. Int. J. Comput. Vis. **104**(2), 154–171 (2013)
18. Simonyan, K., Zisserman A.: Very deep convolutional networks for large-scale image recognition. Computer Science (2014)
19. Neubeck, A., Gool, L. V.: Efficient non-maximum suppression. In: International Conference on Pattern Recognition, Hong Kong, pp. 850–855. IEEE Computer Society (2006)