Hindawi
Complexity
Volume 2023, Article ID 9872756, 1 page
https://doi.org/10.1155/2023/9872756

Retraction

Retracted: Self-Correction Ship Tracking and Counting with Variable Time Window Based on YOLOv3

Complexity

Received 19 December 2023; Accepted 19 December 2023; Published 20 December 2023

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Manipulated or compromised peer review

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] C. Liu and J. Li, "Self-Correction Ship Tracking and Counting with Variable Time Window Based on YOLOv3," *Complexity*, vol. 2021, Article ID 2889115, 9 pages, 2021.

Hindawi
Complexity
Volume 2021, Article ID 2889115, 9 pages
https://doi.org/10.1155/2021/2889115

Research Article

Self-Correction Ship Tracking and Counting with Variable Time Window Based on YOLOv3

Chun Liu and Jian Li

Hubei University of Technology School of Computer Science, Wuhan 430068, China

Correspondence should be addressed to Chun Liu; liuchun@hbut.edu.cn

Received 8 April 2021; Revised 11 May 2021; Accepted 21 May 2021; Published 1 June 2021

Academic Editor: Zhihan Lv

Automatic ship detection, recognition, and counting are crucial for intelligent maritime surveillance, timely ocean rescue, and computer-aided decision-making. YOLOv3 pretraining model is used for model training with sample images for ship detection. The ship detection model is built by adjusting and optimizing parameters. Combining the target HSV color histogram features and LBP local features' target, object recognition and selection are realized by using the deep learning model due to its efficiency in extracting object characteristics. Since tracking targets are subject to drift and jitter, a self-correction network that composites both direction judgment based on regression and target counting method with variable time windows is designed, which better realizes automatic detection, tracking, and self-correction of moving object numbers in water. The method in this paper shows stability and robustness, applicable to the automatic analysis of waterway videos and statistics extraction.

1. Introduction

Target tracking has great demand in computer vision research in recent years. Traditional target tracking algorithms generally use the target's color, texture, contour, gradient histogram, Haar, SIFT, SURF, and other single features to represent the target in the process of target appearance modeling [1]. Traditional target tracking methods include mean shift algorithm [2] and Lucas–Kanade algorithm [3]. The mean shift algorithm is to continuously iterate the candidate target frame along the vector direction with the greatest similarity to the template and converge to the real position of the target, but this method does not respond well to the scale changes and rapid movement of the target [4]. Lucas–Kanade algorithm analyzes the changes of the pixel gray value over time in the video to get optical flow information. Correlation between adjacent sequences is calculated to detect the movement of the target. Lucas–Kanade algorithm assumes that the brightness of the target is constant during movement, and the displacement of the target in adjacent frames is small [5]. Therefore, this method is only suitable for scenes where the background and illumination do not change significantly, which has major limitations in this regard.

In response to the above concerns, a method based on correlation filtering was first proposed, namely, the Minimum Output Sum of Squared Error (MOSSE) [6]. It can distinguish the background and the target effectively by a discriminative filter generated by calculating the minimum mean square error of the output result. In the tracking process, the response image of the search area after the filter action is used to locate the target. The larger the value of the response image, the more the connection between the image and the target located. More improvements were made afterwards, for example, selecting relevant search area based on the size and location of the target [7], but these algorithms are closely related to the search area setting. If the area scale is too small or too large, it may cause target loss or tracking drift [8].

Great breakthrough has been made in tracking algorithms with the application of deep learning in tracking algorithms. The first method to apply deep learning to tracking tasks is to use a deep target tracking framework that creatively combines offline training with online fine-tuning

[9]. In 2013, Wu et al. [10] used auxiliary image data to pretrain the deep learning model, then tracked and fine-tuned online, and proposed the DLT target tracking algorithm. In 2015, the GOTURN algorithm proposed by Held et al. [11] realized the end-to-end deep learning target tracking model for the first time. The authors used a large amount of data offline to train and track unseen category samples and compared the target information of the previous frame with the search area of the current frame. The convolution features are output to the fully connected layer to regress and predict the target position. The 2016 VOT champion MDNet directly uses the pretrained CNN parameters to track the video and fine-tunes during tracking. During the tracking process, the model is periodically updated online to adapt to the changes in the target and scene, and the effect is significantly improved [12]. In the same year, Luca Bertinetto et al. [13] proposed SimFC, a target tracking algorithm based on deep learning, to track targets through a fully convolutional twin network. Leal-Taixé et al. [14] developed a new method of data correlation in pedestrian tracking that uses the tracking-by-detection method under the deep learning framework by two-stage learning and matching. In 2017, Kang et al. [15] proposed a time-series convolutional neural network structure for target detection in video streams and combined the POI target tracking algorithm in the task to improve the overall detection accuracy in the video. In addition, Zhu et al. [16] proposed the deep feature flow (DFF) algorithm for target recognition in the video, using only the convolutional neural network on the key video frames in the video and then using the flow field to convert the depth feature map transmitted to other frames. Later, Yang and Chan [17] applied the residual network to target tracking and proposed the CREST algorithm to perform residual learning by detecting the difference between the extracted convolution features and the real data of the target object. In 2018, the SA-Siam target tracking algorithm proposed by He et al. [18] improved the network structure on the basis of SiamFC and adopted the double twin network structure to improve the model's discriminative performance in target tracking. In 2019, Voigtlaender et al. [19] introduced a twin two-stage full-image redetection architecture using tracklet dynamic programming algorithm on the basis of Faster R-CNN. The proposed tracklet dynamic programming algorithm can play its role considering the overlap of ship imaging and lost images.

However, interferences such as weather conditions, illumination, ship type, and overlapping of ships greatly impact the accuracy of general target tracking algorithms, which cause frequent repeated counting. In order to solve these problems mentioned above, this paper proposes a self-correcting ship target tracking and counting method with variable time window based on YOLOv3. First, the network structure is trained through Darknet-53 to obtain a ship detection model in waterway or underwater transportation scenarios. Target feature extraction, screening, and matching were performed by the overall HSV histogram and the local LBP histogram. A preliminary target tracking was completed before integrating the variable time window model composed of multiframe regression to determine the direction and statistical counting. It is applicable to jitter generated during smooth count. This algorithm can not only suit real-time monitoring and intelligent early warning of navigation facilities but also help in decision-making for navigation facilities' maintenance.

2. YOLOv3 Target Detection Algorithm

The YOLOv3 target detection algorithm is an end-to-end target detection algorithm, which is of fast speed and high accuracy, which meets the requirements of real-time detection [20]. It uses a fully convolutional neural network based on the Darknet-53 network to extract image features. The network consists of 53 convolutional layers of 3×3 and 1×1 [21]. In response to the disappearance of gradients that may be caused by too deep layers of feature extraction, residuals are used to greatly reduce the channels of each convolution, and 3 feature images of the input with different scales are multiscale predicted to output [22]. YOLOv3 target detection algorithm is shown in Figure 1. All candidate targets come from the targets detected by this YOLOv3 target detection algorithm in the follow-up tracking.

3. Target Tracking Algorithms

Target tracking refers to a complete process of target movement direction and path. The target's position in each frame in subsequent frames is predicted with the target position of an initial frame of a video sequence given [23]. In general, a typical target tracking algorithm process is composed of a motion model, an appearance model, an observation model, and an online update mechanism [24]. Although achievement has been made in the field of target tracking, there are still many key technical problems that need to overcome. For example, if the object is blocked during tracking, the algorithm may recover recognition and counting, or the algorithm may detect it as a new object so as to re-recognize and repeat counting. For this reason, this paper combines discrimination with correction in the tracking algorithm. Count suppression and recovery are performed in a variable time window.

The process of this tracking algorithm is as shown in Table 1:

The variable time window self-correction model is divided into two parts, which are the target count based on the variable time window and the motion direction discrimination based on multiframe regression. The former one can realize counting self-correction if the ship is blocked and then counting is repeated, while the latter one can realize self-correction of the movement direction according to motion tendency.

3.1. Feature Extraction and Target Selection. During target tracking, there may be multiple targets available to be tracked. In order to be able to match only one target on the screen that is closest to the initial template determined, it is necessary to screen [25]. Therefore, HSV color histogram

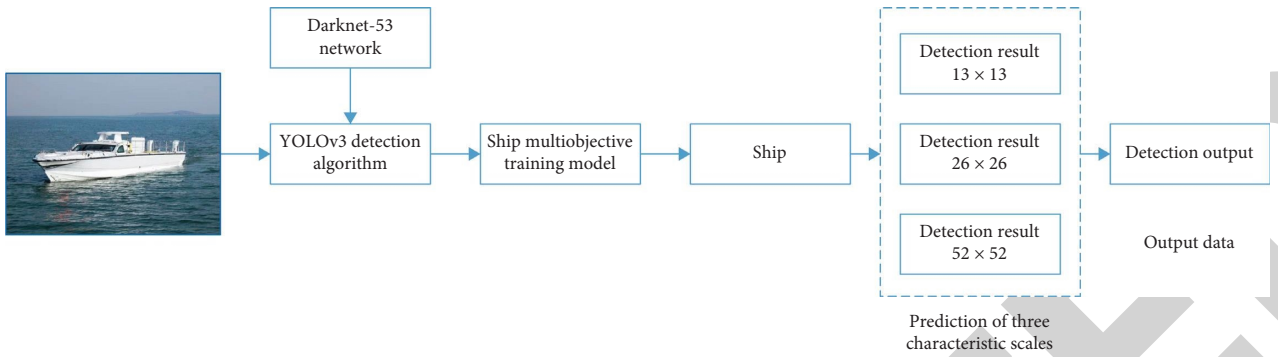


Figure 1: YOLOv3 detection algorithm flowchart.

Table 1: The process of this tracking algorithm.

Tracking algorithm
(1) Determine the tracking target in the first frame of the image, set the target as a template, and obtain its HSV and LBP feature vectors
(2) Continue target detection and obtain a number of targets to be selected
(3) Obtain the HSV and LBP features of each target to be selected, calculate their similarity rates with the tracking target in turn, and determine a candidate target with the highest similarity rate
(4) Use a variable time window self-correction model to avoid count repetition, misjudgment of the direction, and jitter

and LBP histogram are used to screen and select the object to be tracked.

The HSV features can show the changes of each component independently. It describes the proportion of different colors in the whole image and does not pay attention to the spatial position of each color. Therefore, the HSV provides an overall spatial feature that can better describe the shape of the object, especially suitable to describe objects with large deformation [26]. LBP feature is just the opposite that is more suitable to describe some local features, such as local texture and information of the spatial edge [27], because the LBP features use the gray value of the center pixel as the threshold value. Compared with its neighborhood, the obtained binary code is used to express the local texture features, and the gray scale of the LBP operator does not change with any single transformation, so the gray scale has strong robustness, not to mention that calculation of LBP features is accessible that is time-saving for detection [28]. The following formula is the extraction process of LBP features:

$$P(x_c, y_c) = \sum_{P=0}^{P-1} 2^P s(i_p - i_c), \tag{1}$$

where (x_c, y_c) indicates the center pixel; i_c is the gray value; i_p is the gray value of adjacent pixels; and s is a symbolic function, i.e.,

$$s(x) = \begin{cases} 1, x \geq 0, \\ 0, x < 0. \end{cases} \tag{2}$$

The calculation formula of HSV histogram feature similarity is as follows:

$$s(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\overline{H_1}\overline{H_2}N^2}} \sum_1 \sqrt{H_1(I)H_2(I)}}. \tag{3}$$

LBP histogram feature similarity calculation formula is as follows:

$$s(L_1, L_2) = \sqrt{1 - \frac{1}{\sqrt{\overline{L_1}\overline{L_2}M^2}} \sum_1 \sqrt{L_1(I)L_2(I)}}, \tag{4}$$

where H_1 represents the HSV feature histogram vector of the candidate target, H_2 is the HSV feature histogram vector representing the template target, and N indicates the number of bins in the histogram. $\overline{H_k} = 1/N\sum_J H_k(J)$, $H_k(J)$ means the HSV color vector statistic value of the bin whose serial number is J in H_k; L_1 and L_2 represent the LBP feature histogram vector of the candidate target and the template target, respectively; M indicates the number of bins in the histogram; $\overline{L_k} = 1/M\sum_J L_k(J)$, $L_k(J)$ means L_k the HSV color vector statistic value of the bin whose serial number is J in L_k.

Through the above calculations, histogram features of the HSV and LBP of the target to be tracked can be obtained, as well as the feature similarity between the candidate ship and the template. Similarity scores of all ships to be tracked are calculated with weight of the HSV feature set as 1 and weight of the LBP feature set as 2. Finally, the target ship is selected based on the similarity rate.

3.2. Self-Correction Ship Tracking and Counting with Variable Time Window

3.2.1. Variable Time Window Counting. After calculating HSV histogram feature similarity and LBP histogram feature similarity, each tracked ship should be recognized as one ship from beginning to end. Therefore, when each ship is detected, each different feature will be assigned a unique flag value by calculating the HSV histogram feature and the LBP histogram feature, and the count will be increased by one.

This flag value will be marked with the detection wireframe. During follow-up tracking, since ship tracked has a very similar feature to the one at the initial time, they are given the same flag value, indicating that they are the same ship, and the count will not be increased, as shown in Figure 2.

However, in the real tracking process, there may be two scenarios that may mislead the counting result. One is that when the two ships are blocked by each other, there may be a situation of "appear as one," and the two ships as a whole may show a large difference in similarity with the previous one; the second is when the ship is blocked and reappears. It may be caused by angle and light. The ship blocked also showed a large difference in similarity before and after the blocking. In both cases, it may be considered that a new ship appears, causing one count to be added.

Therefore, a variable time window counting method is designed in response to the first case. It can be concluded that if some of the ships being tracked suddenly "disappear," then the ships must be blocked. The length of this time window will be of a direct proportion to the size difference between the blocked ship and the one blocking the ship, and the speed difference will be of an inverse proportion. In the video image, smaller and slower ships often take longer time to pass the same block.

In the navigation process, the speed of each ship changes little so that the average speed of the tracked ship from appearing to disappearing can be used as the speed of the ship. The size difference between the two ships can be obtained by subtracting the length of the wireframe of the tracking ship.

Therefore, the length of the time window varies according to the size and speed of the ship, namely,

$$L = a \cdot \left| \frac{x_1 - x_2}{v_1 - v_2} \right|, \tag{5}$$

where L represents the size of the time window, a is a parameter to be determined, x_1, v_1 are the length and speed of the ship blocked, and x_2, v_2 are the length and speed of the ship blocking.

However, the second scenario is a force of nature that is out of control, so the length of the time window calculated by formula (5) can be appropriately extended to improve the fault tolerance rate.

Finally, determine whether the "new" ship disappears and the initial ship reappears within the time window calculated above. If so, then the count should be corrected, and the number of ships should be subtracted by one.

3.2.2. Multiframe Regression Direction Determination. Theoretically, when tracking a target, the direction of the movement of the target can be calculated according to the area coordinates of the same target in two adjacent frames. For example, if the coordinates of the target in the first frame are (12, 22) and the coordinates in the second frame are (14, 22), the target moves to the right side of the screen. However, in real situation, the coordinates of the target object may be jittered as the screen and the target move synchronously. If the above method is used, the object moving in the same direction is very likely to appear at (12, 22), (11, 22), and (13, 22), and a moving order of first to the left and then to the right may be concluded. The multiframe regression algorithm can handle this concern: regression analysis of the coordinates of the target object in several frames before and after the current frame should be conducted, the movement tendency of the object is obtained, and then a comprehensive judgment is made by combining the coordinate comparison of two adjacent frames. Thus, movement direction of the target object is determined.

When determining the direction of the ship's movement, the following linear relation can be established:

$$Y = A + BX + \varepsilon. \tag{6}$$

Among them, A with B is an undetermined parameter, ε is the random error, X represents the time, and Y represents the abscissa of the ship.

When the direction of the ship's movement at a certain moment is to be determined, ship coordinates of the frame at this moment and several adjacent frames before it were extracted. Positive or negative values of B, through the regression equation, can be calculated. When B is positive, it can be preliminarily determined that the ship movement direction tendency is to move to the right side of the screen, and vice versa. When the ship moves to the left, mark L on the tracking wireframe; otherwise, mark R.

Randomly select three moments from a video of the ship moving all the way to the right, and record the coordinate data of several previous frames adjacent to them (the time is 0 at this moment), as shown in Table 2.

If coordinates of the two frames before and after were calculated in the traditional way, abscissa becomes smaller and may cause wrong determination that the ship is moving to the left of the screen.

Calculating the regression equation, it shows that $Y = 2.4978 + 0.0383X$, $Y = 13.0244 + 0.045X$, and $Y = 18.78 + 0.0533X$.

Because $B = 0.0383$, 0.045, and 0.0533 are positive values, it can be determined that the ship is moving to the right of the screen at this time.

Daily practice and experimental validation found that the ship's direction should be determined based on the ship's movement tendency, not the change of two frames before and after, so this method can effectively reduce jitter and improve the accuracy of the judgment.

3.3. Improved YOLOv3 Neural Network with the Self-Correction Model. The variable time window self-tuning tracking counting model can be combined with the YOLOv3 neural network, and a correction network can be designed to realize self-correction.

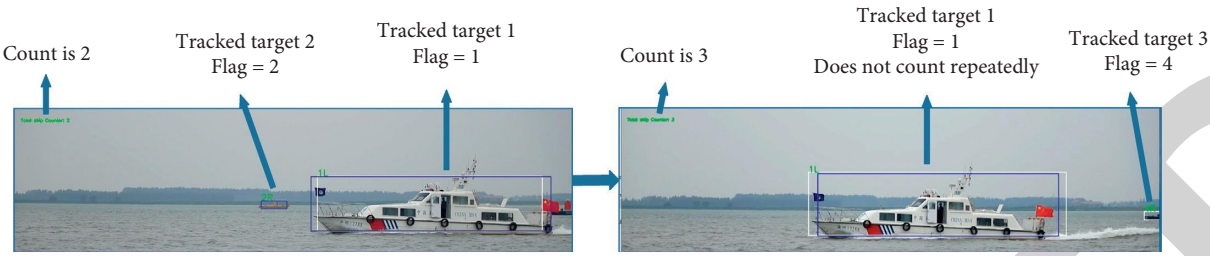


Figure 2: Ship counting principle.

Table 2: Data coordinates measured in the experiment.

Time	Abscissa
(a)	
−8	2.1
−7	2.2
−6	2.4
−5	2.3
−4	2.3
−3	2.5
−2	2.4
−1	2.5
0	2.4
(b)	
−8	12.6
−7	12.7
−6	12.9
−5	12.8
−4	12.7
−3	12.9
−2	13.0
−1	13.1
0	12.9
(c)	
−8	18.3
−7	18.4
−6	18.5
−5	18.4
−4	18.6
−3	18.7
−2	18.9
−1	18.7
0	18.6

The correction network is a chain structure that contains 2 modules. The first module contains 3 convolution units, and the second module contains 3 fully connected layers. Then, the network structure of the YOLOv3 model is modified to integrate the self-correction network into the tracking network structure based on YOLOv3. The tracking network can be updated by processing and correcting the output result of the tracking network, as shown in Figure 3.

Results of the correction network are used to update and correct the tracking network at the same time and can be used as the network for learning in the next frame. The system generates the current average loss during training for the real-time observation. Because it is an important indicator for evaluating the effect of target detection [29], we can see the effect of training intuitively. After 400 training sessions, the average loss at this time changed very little, which shows that the training has been basically completed, and a relatively good effect can be expected. The current average loss convergence diagram is shown in Figure 4:

In this study, the scale of the best sample determined by the correction network is used as the scale of the tracking target [30], which greatly accelerates calculation.

4. Test Results

The development environment of NVIDIA 2060 GPU and PyCharm editor are used in this test. 1565 ship images were obtained by Yangtze River field shooting and internet search. VOC dataset was prepared. Images were divided into the training set, validation set, and test set at a ratio of 6 : 2 : 2 with a training set of 939 pictures, test set of 313 pictures, and verification set of 313 pictures. Darknet-53 is used to train the network structure and modify the sample classification and algorithm according to the actual situation, and the self-calibration model is added to the network structure. Before starting the training session, according to the GPU performance, batch = 64, subdivisions = 8, learning rate = 0.001, momentum coefficient is set as 0.9, and the adjusting policy of learning rate is set as steps. Then, the dataset is trained so that the system can automatically save all records and train models. There is an optimal training model that is automatically covered available to use.

After training, a ship picture is randomly selected for detection. It can be seen that the detection time is 35.723 ms, and the matching degree of "ship" tag in ship detection is 90%, as shown in Figure 5.

Video tests in water transportation and waterway scenarios are mainly used to deal with issues such as the diverse ship types, different light, and different shooting angles of view. This article adopts multiple types of surveillance and video materials with diverse test scenarios under various weather conditions to ensure that the target tracking and counting method can operate stably in multiple scenarios such as video or surveillance. The above ship detection model is used by uploading videos of ships in different

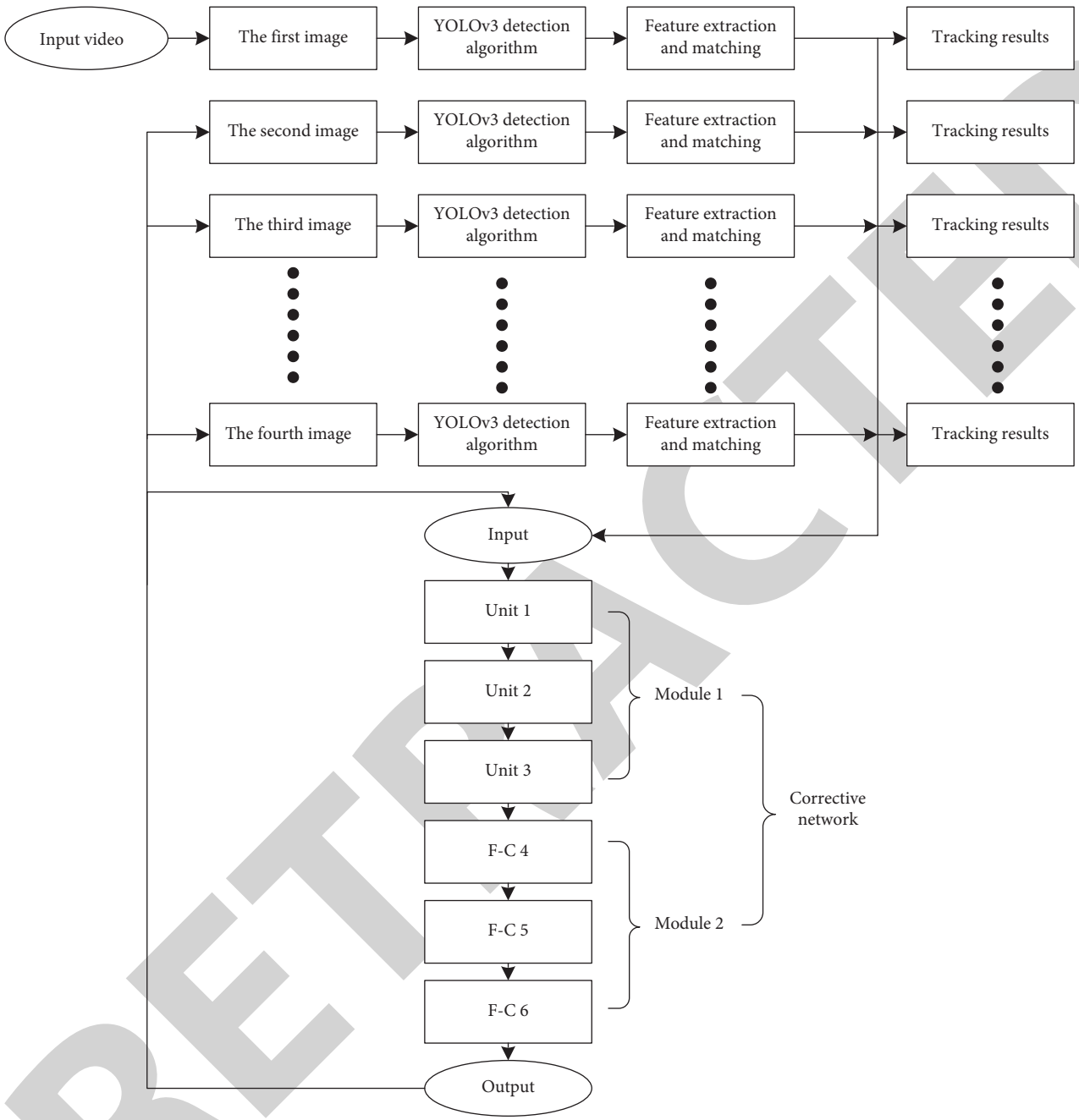


Figure 3: Integrated network structure diagram.

scenarios. Target detection and tracking counting show high accuracy. The tracking effect is as shown in Figure 6.

In order to verify whether the self-calibration counting model proposed in this paper can self-calibrate the number of navigable ships by using a variable time window, a control experiment was carried out. The experimental group is an improved network that incorporates the self-calibration model, and the control group is an unfused network, ordinary YOLOv3 network. By comparing the variable time window self-correcting tracking counting model, the counting results are calculated. Nine different videos were used in the experiment, and the tracking counting effects were counted, respectively, of whether the algorithm is applied. The experimental results are shown in Table 3

It can be seen from Table 2 that when the self-correcting counting model proposed in this article is not used, the counting is very inaccurate due to screen jitter and occlusion. In particular, the number of ships in video sequence no. 2 is 70, and the ships are repeatedly counted 18 times. After using the self-correcting counting model, the number of ships in video sequence no. 2 is counted as 54. Reference to the experimental data of other video sequences shows that the performance requirements for real-time tracking and counting of targets have been basically met.

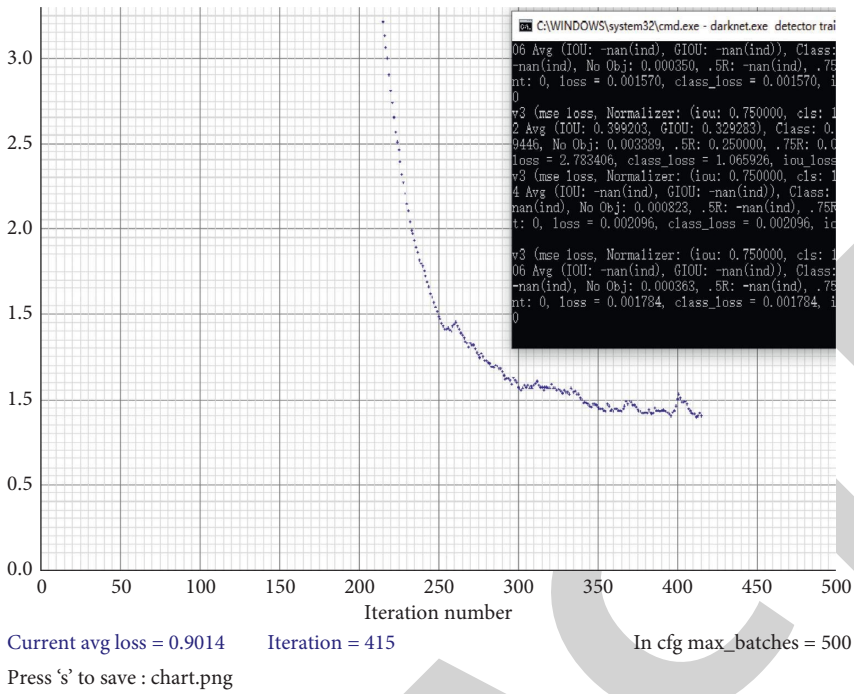


Figure 4: Current average loss convergence graph.

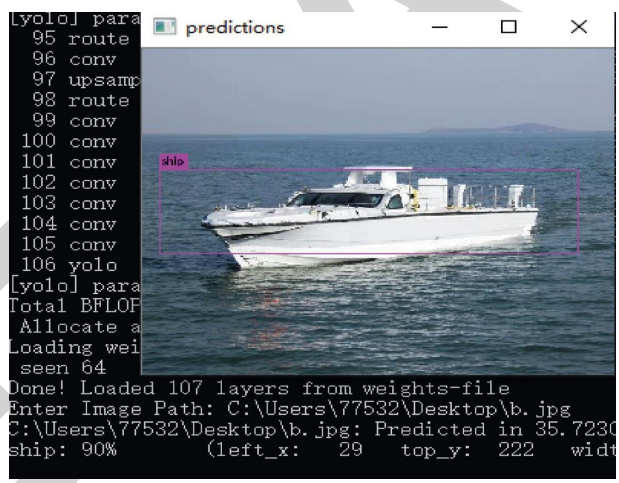
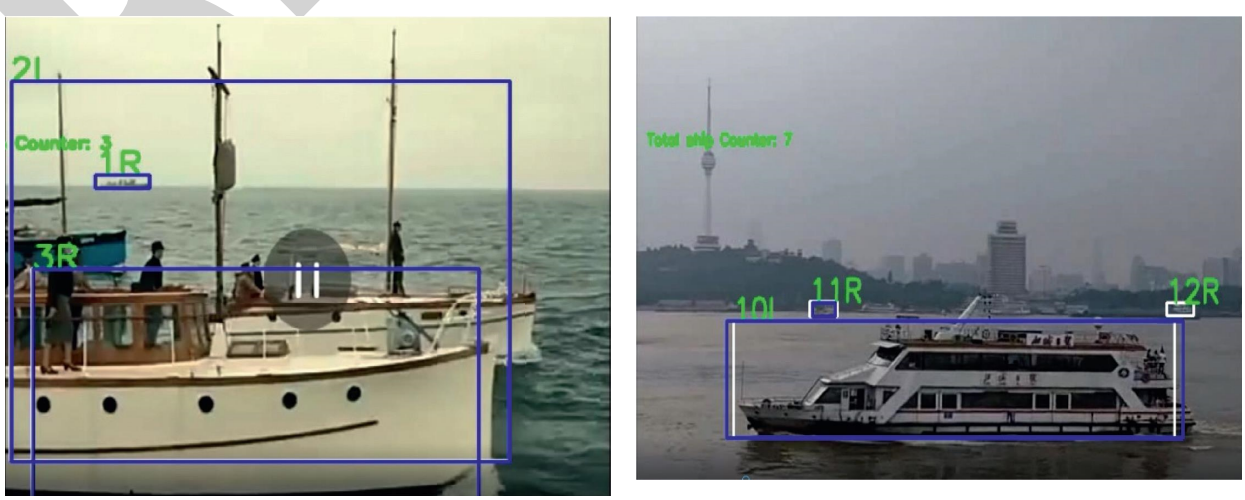


Figure 5: Ship target detection effect.



(a) (b)

Figure 6: Ship target tracking effect.

Table 3: Comparison of counting effects before and after applying the self-correction algorithm.

Video serial number	Actual number of ships	Original count	Count after improvement
1	12	21	12
2	52	70	54
3	24	30	26
4	5	6	5
5	17	20	18
6	8	10	8
7	30	36	32
8	22	30	23
9	20	25	20

5. Conclusion

In this paper, an algorithm of ship target tracking and self-correction counting in the waterway scene based on YOLOv3 is proposed. It consists of system architecture, dataset, model training, feature extraction, target tracking, direction discrimination and counting, detection results, and result analysis. Analysis of the test results of a large number of video materials shows that the ship target tracking and counting system described in this article can achieve stable counting despite complex scenarios and weather conditions. However, under the condition that two similar ships appear and overlap each other, the counting error still cannot be prevented, which needs follow-up research for optimization.

Data Availability

All the data included in this study are available from the corresponding author upon request.

Disclosure

The funders had no role in the design of the study, in the collection, analyses, or interpretation of the data, in the writing of the manuscript, or in the decision to publish the results.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This study was supported by the National Major Special Project "R&D and Application of Decision Support System for Rights Protection and Law Enforcement of Island Reefs and Marine Structures" (no. 2017YFC1405400).

References

[1] K. H. Zhang, J. Q. Fan, and Q. S. Liu, "Advances on visual object tracking in past decade," *Computer Science*, vol. 48, no. 3, pp. 40–49, 2021.

[2] K. Du, Y. F. Ju, Y. L. Jin, G. Li, Y. Li, and S. Qian, "Object tracking based on improved mean shift and SIFT," in *Proceedings of the 2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, pp. 2716–2719, IEEE, Yichang, China, April 2012.

[3] J. B. Shi and C. Tomasi, "Good features to track," in *1994 Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1063–6919, IEEE, Seattle, WA, USA, June 1994.

[4] G. Johncy and A. A. Felise, "An efficient power theft detection using mean-shift clustering and deep learning in smart grid," *IOP Conference Series: Materials Science and Engineering*, vol. 983, no. 1, 2020.

[5] Y. Ahmine, G. Caron, E. M. Mouaddiba, and F. Chouireb, "Adaptive lucas-kanade tracking," *Image and Vision Computing*, vol. 88, pp. 1–8, 2019.

[6] D. S. Bolme, J. R. Beveride, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2544–2550, IEEE, San Francisco, CA, USA, June 2010.

[7] Q. Zhu, *Research on Minimum Squared Error Based Face Feature Extraction and Classification Algorithms*, Harbin Institute of Technology, Harbin, China, 2014.

[8] Y. M. Tang, Y. F. Liu, and H. Huang, "Survey of single-target visual tracking algorithms," *Measurement & Control Technology*, vol. 39, no. 8, pp. 21–34, 2020.

[9] N. Y. Wang and D. Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 1, pp. 809–817, Lake Tahoe, CA, USA, December 2013.

[10] Y. Wu, J. Lim, and M. H. Yang, "Online object tracking: A benchmark," in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, Portland, OR, USA, June 2013.

[11] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 FPS with deep regression networks," 2016, http://arxiv.org/ads/1604.01802.

[12] M. P. Venugopal, D. Mishra, and G. R. K. S. Subrahmanyam, "Computationally efficient deep tracker: guided MDNet," in *Proceedings of the 2017 Twenty-third National Conference on Communications (NCC)*, pp. 1–6, Chennai, India, March 2017.

[13] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: complementary learners for real-time tracking," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Las Vegas, NV, USA, June 2016.

[14] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, "Learning by tracking: siamese CNN for robust target association," 2016, https://arxiv.org/abs/1604.07866.

[15] K. Kang, W. Ouyang, H. Li, and X. Wang, "Object detection from video tubelets with convolutional neural networks," 2016, https://arxiv.org/abs/1604.04053.

[16] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," 2016, https://arxiv.org/abs/1611.07715.
[17] T. Yang and A. B. Chan, "Recurrent filter learning for visual tracking," in *Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 2010–2019, Venice, Italy, October 2017.
[18] A. F. He, C. Luo, X. M. Tian, and W. Zeng, "A twofold siamese network for real-time object tracking," 2018, https://arxiv.org/abs/1802.08817.
[19] P. Voigtlaender, J. Luiten, P. H. S. Torr, and B. Leibe, "Siam R-CNN: visual tracking by re-detection," 2019, https://arxiv.org/abs/1911.12836.
[20] X. M. Jiang, F. G. Xiang, M. H. Lv, W. Wang, Z. Zhang, and Y. Yu, "YOLOv3_slim for face mask recognition," in *Proceedings of the International Conference on Artificial Intelligence and Cloud Computing (ICAICC) 2020*, vol. 1771, pp. 18–20, Suzhou, China, December 2020.
[21] Z. Q. Zhang, Y. Y. Liu, C. C. Pan et al., "Application of improved YOLOv3 in aircraft recognition of remote sensing images," *Electronics Optics & Control*, vol. 26, no. 4, pp. 28–32, 2019.
[22] S. S. Padmanabula, R. C. Puvvada, V. Sistla et al., "Object detection using stacked YOLOv3," *IIETA*, vol. 25, no. 5, 2020.
[23] L. Meng and X. Yang, "A survey of object tracking algorithms," *Acta Automatica Sinica*, vol. 45, no. 7, pp. 1244–1260, 2019.
[24] A. J. BoltesMaik and R. A. Katharina, "A hybrid tracking system of full-body motion inside crowds," *Sensors*, vol. 21, no. 6, 2021.
[25] J. Tian and K. J. Wang, "Superpixel object tracking with adaptive compact feature," *Journal of Image and Graphics*, vol. 22, no. 10, pp. 1409–1417, 2017.
[26] W. Bi, W. G. Huang, Y. P. Zhang et al., "Object detection based on salient contour of image," *Acta Electronica Sinica*, vol. 45, no. 8, pp. 1902–1910, 2017.
[27] W. Ling and L. Sheng, "Comparative analysis and application of LBP face image recognition algorithms," *International Journal of Communication Systems*, vol. 34, no. 2, 2021.
[28] P. P. Ji, K. Chen, Z. Liu et al., "Random ferns with HOG-LBP feature and online multi-instance learning for complex tracking," *Journal of Ningbo University (Natural Science & Engineering Edition)*, vol. 28, no. 4, pp. 42–47, 2015.
[29] M. Gao, *Research on Object Tracking and Trajectory Prediction in Complex Traffic Environment Based on Deep Learning*, Jilin University, Changchun, China, 2020.
[30] T. Zhang and L. Zhang, "An object detection algorithm based on multi-scale feature fusion [J/OL]," *Laser & Optoelectronics Progress*, pp. 1–11, 2020.