Postprint of a Lightweight Fruit Load Estimation Model for Edge Computing Devices

Authors: Xia Xue, Chai Xiujuan, Zhang Ning, Zhou Shuo, Sun Qixin, Sun Tan

Date: 2023-08-14T00:00:00+00:00

Abstract

[Purpose/Significance] Fruit load is an important indicator for fruit tree cultivation management. Traditional manual sampling methods for estimating fruit load on trees are not only time-consuming and labor-intensive, but also prone to significant errors. This study proposes a lightweight model for edge computing devices to achieve automatic estimation of on-tree citrus fruit load from videos. [Method] The model employs a CSPDarkNet53+PAFPN structure as the feature extraction network to achieve faster inference speed and lower model complexity. During fruit tracking, the Byte algorithm is introduced to improve the data association strategy of FairMOT for predictive tracking of citrus in videos, thereby enhancing the accuracy of fruit load estimation. [Results and Discussion] Performance testing of the model on the edge computing device NVIDIA Jetson AGX demonstrated that the proposed model achieved an Average Estimating Precision (AEP) of 91.61% and processing speed of 14.76 Frames Per Second (FPS) for citrus fruit load estimation. The coefficient of determination R2 between model-estimated values and manually measured ground truth was 0.9858, with a Root Mean Square Error (RMSE) of 4.1713. The model parameters, computational cost (Floating Point Operations, FLOPs), and model size were 5.01 M, 36.44 G, and 70.20 MB, respectively, exhibiting superior fruit load estimation performance and lower model complexity compared with alternative models. [Conclusion] The experimental results demonstrate the effectiveness of the proposed model for citrus fruit load estimation on edge computing devices. The remote monitoring system for orchard fruit load developed based on this algorithmic model can meet the requirements for estimating fruit load on trees while the orchard mobile platform is in motion. This study can provide technical support for automatic monitoring and analysis of orchard productivity.

Full Text

A Lightweight Fruit Load Estimation Model for Edge Computing Equipment

XIA Xue[1], **CHAI Xiujuan**[1], **ZHANG Ning**[1], ***ZHOU Shuo***[1], ***SUN Qixin***[1], ***SUN Tan***[2] [1]Agricultural Information Institute, Chinese Academy of Agricultural Sciences/Key Laboratory of Agricultural Big Data, Ministry of Agriculture and Rural Affairs, Beijing 100081, China; [2]Chinese Academy of Agricultural Sciences, Beijing 100081, China

Abstract:
[**Objective**] Fruit load estimation is a critical indicator for orchard management. Traditional manual sampling methods are not only labor-intensive and time-consuming but also prone to significant errors. This study proposes a lightweight model for edge computing devices to automatically estimate on-tree citrus fruit load from videos. [**Methods**] The model adopts the FairMOT structure with CSPDarkNet53+PAFPN as the feature extraction network, achieving faster inference speed and lower model complexity. A Byte data association strategy is introduced during fruit tracking to predict and track citrus fruits in videos, thereby improving fruit load estimation accuracy. [**Results and Discussion**] Performance tests on edge computing devices demonstrate that the proposed model achieves an Average Estimating Precision (AEP) of 91.61% and processing speed of 14.76 frames per second (FPS). The coefficient of determination (R^2) between model estimates and manually measured ground truth is 0.9858, with a Root Mean Square Error (RMSE) of 4.1713. The model parameters, computational cost (FLOPs), and model size are 5.01 M, 36.44 G, and 70.20 MB, respectively, showing superior fruit load estimation performance and lower complexity compared to alternative models. [**Conclusions**] The results validate the effectiveness of the proposed model for citrus fruit load estimation on edge computing devices. The orchard fruit load remote monitoring system developed based on this algorithm can meet the requirements for fruit load estimation on mobile orchard platforms, providing technical support for automated orchard productivity monitoring.

Keywords: precision horticulture; fruit load estimation; edge computing; deep learning; multi-object tracking; lightweight model

---

Fruit load estimation is a crucial indicator for evaluating orchard productivity, enabling growers to precisely understand orchard production status and rationally arrange cultivation management, harvesting, storage, and marketing activities [1]. Traditional methods for estimating on-tree fruit load rely primarily on manual sampling, such as randomly selecting a fixed percentage (5% or 10%) of trees for fruit counting to infer total orchard yield. However, this prolonged outdoor work is not only time-consuming and labor-intensive but also susceptible to counting errors due to fatigue or other disturbances. Therefore,

automatic fruit load estimation is essential for modern orchard production.

Machine vision, as an important domain of artificial intelligence applications, has been widely employed in smart orchard research due to its low cost and high efficiency. Early studies on fruit load estimation focused on traditional image processing methods based on handcrafted features, including texture features [2, 3], color features [4, 5], and shape features [6, 7]. In recent years, deep learning techniques have been extensively investigated for fruit load estimation. Sa et al. [8] pioneered the use of deep learning for fruit detection and load estimation. Chen et al. [9] employed fully convolutional networks to extract candidate blobs and used convolutional neural network-based regression models to estimate fruit count per blob. Bargoti and Underwood [10] proposed a fruit load estimation method based on Faster R-CNN with transfer learning. Häni et al. [11] developed an end-to-end system combining U-Net and Faster R-CNN to estimate apple counts from clusters, achieving 0.978 accuracy. Li et al. [12] utilized YOLOv5 for apple detection and combined it with a yield fitting network for tree yield prediction. Kestur et al. [13] specifically designed a MangoNet model for mango fruit load estimation.

However, most existing methods estimate fruit load from static images and cannot perform dynamic estimation from videos. To address this limitation, Tracking-by-Detection (TBD) methods have been introduced for video-based fruit load estimation. Gao et al. [14] proposed a method based on YOLOv4-tiny and Kalman filtering for Fuji apple detection and counting in videos. Similarly, Wang et al. [15] used Kalman filters for mango motion tracking to estimate mango counts in videos. TBD methods adopt a two-stage strategy: first performing object detection, then feeding detection results into another model for data association-based multi-object tracking. This leads to low algorithm efficiency, slow processing speed [16, 17], and the requirement for high-performance workstations, making them unsuitable for edge computing devices with limited hardware resources in practical orchard production.

With the rapid development of multi-object tracking technology, Joint Detection and Embedding (JDE) methods have become mainstream. JDE methods employ an end-to-end single-stage strategy that integrates detection and tracking into one framework to simultaneously perform object detection and identity re-identification (ReID) tasks, avoiding the complexity of multi-stage processing [18]. Zhang et al. [19] proposed the FairMOT framework based on CenterNet and JDE, which considers both object position and ReID features during tracking and reduces inference time through shared computation. Zhang et al. [20] proposed the ByteTrack algorithm, which incorporates low-confidence detection boxes into the matching process alongside high-confidence boxes to mine more true targets, improving both inference accuracy and speed. However, ByteTrack's association matching only employs motion estimation without identity ReID feature similarity calculation [21].

To achieve automatic fruit load estimation under orchard conditions, this study proposes a lightweight model for edge computing devices, using on-tree cit-

rus as the research object. The main contributions include: (1) Adopting the lightweight CSPDarkNet53+PAFPN structure as the feature extraction network to ensure feature representation capability while achieving faster inference speed and lower model complexity; (2) Introducing the Byte algorithm during the fruit tracking stage to improve FairMOT's data association strategy, designing a fruit multi-object tracking module for predictive tracking of citrus fruits to enhance estimation accuracy; (3) Developing an orchard fruit load remote monitoring system that embeds the algorithm model to enable automatic fruit load estimation.

2.1 Data Collection

The experimental orchard is located in Nalang Village, Jiangnan District, Nanning City, Guangxi Zhuang Autonomous Region (108°06 E, 22°79 N). Wo citrus cultivated at this location served as the research object, and citrus video image data were collected in late November 2019. The orchard environment is shown in Figure 1: see original paper. Researchers moved along tree rows while holding smartphones for filming, with camera distance from rows maintained at 0.5–1.0 m. Data collection was conducted under sunny and cloudy weather conditions between 8:00 and 18:00 to ensure coverage of different lighting situations.

Forty videos of trees from different orchard rows were collected and saved as *.mp4 format with a resolution of 1080$×$1920 pixels and frame rate of 24 fps. Sample images from the collected citrus videos are shown in Figure 1: see original paper. To create ground truth annotations for fruit counts, image frames were extracted from video data, yielding 2,846 citrus tree images. Fruit positions and duplicate fruits across frames were annotated. First, a custom coordinate annotation tool (Figure 2: see original paper) was used to label minimum bounding rectangles of fruits, recording the four corner coordinates of each fruit bounding box. Second, a custom fruit pairing annotation tool (Figure 2: see original paper) was used to label duplicate fruits in adjacent images, assigning identical IDs to the same fruit to form a standardized dataset.

3 Lightweight Fruit Load Estimation Model

3.1 Algorithm Overview In complex citrus orchard scenes, occlusion between branches, leaves, and fruits poses challenges for on-tree fruit load estimation. Most existing object tracking methods employ anchor-based detection [22], where misalignment between the target's actual center and the anchor box center can cause extracted identity features to misalign with the target center, reducing tracking accuracy. To mitigate this issue, anchor-free multi-object tracking models [19] have been designed to reduce anchor-related ambiguity for ReID and optimize detection inference.

Based on this analysis, this study follows the FairMOT approach, employing parallel structures for object detection and identity embedding branches. The anchor-free strategy generates bounding boxes while avoiding ambiguity from

anchor-based methods, facilitating better alignment between identity embedding features and detection target centers.

The proposed algorithm consists of two main components: (1) Detecting fruits in video frames and extracting fruit ReID features. The lightweight CSPDarknet53+PAFPN network extracts feature maps that are fed into two parallel branches—one for predicting target position information and the other for identifying target ReID features. (2) Fruit tracking and load calculation. The Byte algorithm [20] is introduced during fruit tracking, combining inter-frame fruit position information and identity features for motion trajectory prediction and identity feature similarity matching. Fruits tracked for more than five consecutive frames are counted, and the number of unique fruit IDs is output as the on-tree citrus fruit load. The algorithm processing flow is shown in [Figure 3: see original paper].

3.2 Citrus Fruit Detection and ReID Feature Extraction The fruit detection and ReID feature extraction network structure is shown in [Figure 4: see original paper], comprising a backbone network, neck network, and prediction head branches.

CSPDarknet53 is derived from Darknet53 [23] by incorporating the Cross Stage Partial Network (CSPNet) concept [24], which resolves gradient information redundancy caused by network backpropagation and effectively reduces parameters and computational cost [25, 26]. The lightweight CSPDarknet53 consumes less hardware computing resources, making it suitable for edge computing devices with limited image processing resources [27]. Therefore, CSPDarknet53 is adopted as the model backbone for feature extraction.

PAFPN consists of Feature Pyramid Network (FPN) [28] and Path Aggregation Network (PAN) [29], which builds a multi-scale feature pyramid structure to fuse deep and shallow features element-wise. This leverages both low-level features with high resolution and high-level features with rich semantic information to obtain more comprehensive feature representation [30]. Therefore, PAFPN is connected after the backbone as the neck network to achieve effective feature fusion through skip connections.

The fused features are fed into parallel branches: the detection branch and identity embedding branch. The detection branch includes three prediction heads, each performing 3$×3*convolutionand*1×$1 convolution on PAFPN output feature maps to estimate the citrus center heatmap, local offset, and object size, as shown in [Figure 5: see original paper].

The identity embedding branch distinguishes different fruit ReID features. To reduce inference time and overfitting risk, the convolution kernel channel number is adjusted from 128 to 64. The kernel extracts ReID features from the input feature map, generating an identity embedding map E $^{64}\times\times$, where W and H represent feature map width and height, and the identity feature at target center (x, y) is E , ^{64}.

3.3 Citrus Fruit Tracking with Byte Data Association Data association is critical for multi-object tracking. Most existing tracking models match only high-confidence detection boxes (scores above a threshold) while ignoring low-confidence boxes. Low-confidence detections often result from occlusion or motion blur; discarding them directly can cause heavily occluded targets to lose trajectories and trigger frequent ID switches. Considering both high- and low-confidence detection boxes for trajectory association can improve tracking continuity. Therefore, this study introduces the Byte data association algorithm from ByteTrack [20], incorporating both ReID features and position association for citrus fruit tracking.

Citrus fruit detection results serve as input to the tracking module. The module distinguishes between high- and low-confidence detection boxes, combines detection box position information with ReID features to form fused features, and employs the Byte data association matching strategy to obtain fruit target trajectories across consecutive video frames. The fruit tracking flowchart is shown in [Figure 6: see original paper].

The citrus fruit tracking procedure is as follows: (1) Let V be the input orchard video, Det the fruit detector, KF the Kalman filter, and T the video trajectory stack where each trajectory includes fruit detection box and identity information. Three thresholds are set: T g and T for detection confidence, and for tracking confidence. (2) All detection boxes and confidences from Det are classified: those above T g form D g (containing box positions and ReID features), while those above T form D (containing box positions and ReID features). (3) For each trajectory in T, KF predicts its current frame coordinates. (4) D g is first associated with all trajectories in T using Intersection over Union (IoU) and identity features to compute similarity between detection boxes and Kalman-predicted boxes, with Hungarian algorithm completing the matching. Unmatched detections are stored in D , and unmatched trajectories in T . (5) Low-confidence detections D are associated with trajectories T using the same method. Unmatched trajectories are stored in T , while unmatched low-confidence detections are directly discarded. (6) Trajectories in T are considered temporarily lost but retained in T. They are removed from both T and T if later matched successfully or if they persist for over 30 frames; otherwise, they remain in T. (7) Detections in D with confidence above that survive for more than two frames initialize new trajectories stored in T. (8) For each video frame, T outputs all detected and tracked fruit bounding boxes with corresponding IDs.

The Byte data association-based citrus fruit tracking method contributes essential fruit ReID information while considering both high- and low-confidence detection boxes, thereby maintaining tracking continuity.

3.4 Loss Functions The total loss function L is the weighted sum of L , L , and L :

L = MATH_1
L = $_1$ ($\|\hat{s}$ - s $\|_1$ + λ $\|\hat{o}$ - o $\|_1$)
L = - $_1$ $_1$ p (k)log(p (k))
L = ω_1(L + L) + ω_2L

where M is the predicted heatmap, M is the ground truth heatmap, N is the total number of targets, ŝ and s are predicted and ground truth bounding box sizes, ô and o are predicted and ground truth center offsets, p(k) is the predicted ID probability distribution for the k-th object, and ω_1 and ω_2 are learnable weight parameters for detection and identity losses.

3.5 Experimental Setup To improve training efficiency, experiments were conducted on a graphics workstation with an Intel i7-10700 (2.90 GHz) CPU, NVIDIA GeForce RTX 3080 (12 GB) GPU, and 32 GB RAM. The software environment comprised Ubuntu 20.04 LTS, CUDA 11.6, Python 3.8, and PyTorch 1.12. The trained model was deployed on an NVIDIA Jetson AGX edge computing device for performance testing. Training parameters for the citrus fruit detection model are listed in .

3.6 Evaluation Metrics For fruit detection, Precision (P), Recall (R), and F1 score (F1) evaluate model performance:

P = TP/(TP + FP)
R = TP/(TP + FN)
F1 = 2PR/(P + R)

where TP is true positives (correctly detected citrus), FP is false positives (background misdetected as citrus), and FN is false negatives (citrus misdetected as background).

For fruit load estimation, Average Estimating Precision (AEP) evaluates algorithm accuracy:

AEP = (1/n) $_1^n$ (S /G) × 100%

where S is algorithm-estimated citrus count, G is manually verified actual count, and n is the number of test videos.

Additionally, coefficient of determination R^2 and Root Mean Square Error (RMSE) evaluate the correlation and error between algorithm-estimated and manually measured fruit counts. Higher R^2 indicates better correlation, while lower RMSE indicates smaller error.

4 Results and Discussion

4.1.1 Comparison of Citrus Fruit Detection Performance Across Models To validate the proposed model, comparative experiments were conducted using ResNet34 and HRNet18 (original FairMOT backbones) and Faster R-CNN as benchmarks, evaluating detection performance and model complexity on the

same test dataset of 211 citrus tree images. Detection results are shown in [Figure 7: see original paper].

The proposed model with CSPDarkNet53+PAFPN achieved 83.6% precision, 89.2% recall, and 86.3% F1 score, outperforming FairMOT(ResNet34), FairMOT(HRNet18), and Faster R-CNN. Faster R-CNN achieved only 65.1% precision, 69.0% recall, and 67.0% F1. The CSPDarkNet53+PAFPN structure better detects fruit locations, laying a foundation for accurate fruit load estimation. Although dense foliage causes occlusion and potential detection failures, the same fruit appears across multiple video frames, allowing detection in subsequent frames if missed in one frame, thus mitigating impact on load estimation accuracy.

4.1.2 Model Complexity Comparison Complexity comparison results are shown in . The proposed model's parameters, FLOPs, and size are 5.01 M, 36.44 G, and 70.2 MB, respectively. Its parameter count is 20.19% of FairMOT(ResNet34) and 41.51% of FairMOT(HRNet18). FLOPs are 78.31% and 87.63% lower than the ResNet34 and HRNet18 versions, respectively. Model size is 23.96% of FairMOT(ResNet34) and 45.00% of FairMOT(HRNet18). Compared with Faster R-CNN, the proposed model shows clear advantages across all complexity metrics, proving its suitability for resource-constrained edge computing devices and mobile orchard platforms.

4.1.3 Ablation Study of Lightweight Backbone Networks To validate CSPDarkNet53's effectiveness and PAFPN's impact, an ablation study compared EfficientNet-Lite with CSPDarkNet53. Results are shown in .

CSPDarkNet53 outperformed EfficientNet-Lite in both detection performance and complexity. Without PAFPN, CSPDarkNet53 achieved 83.3% precision, 88.7% recall, and 85.9% F1, with 4.93 M parameters, 35.42 G FLOPs, and 69.1 MB size. Adding PAFPN improved all detection metrics while maintaining similar model size, demonstrating PAFPN's positive impact on performance.

4.2.1 Comparison of Fruit Load Estimation Performance Across Models To validate the proposed model for fruit load estimation and enable practical orchard applications, an orchard fruit load remote monitoring system was developed by embedding the algorithm model for automatic estimation. The system server runs on edge computing devices, while the monitoring client runs on ordinary computers to receive and display results. The system operation is shown in [Figure 8: see original paper].

Ten test videos containing ten citrus trees were randomly selected, with ground truth counts of 39, 70, 125, 14, 81, 15, 47, 32, 27, and 63 fruits. Performance comparison of different feature extraction networks and tracking strategies is shown in .

Integrating the Byte algorithm into FairMOT improved tracking strategy and enhanced estimation accuracy. With the improved tracking strategy, models using ResNet34, HRNet18, and CSPDarkNet53+PAFPN achieved AEPs of 86.98%, 91.53%, and 91.61%, with processing speeds of 6.05, 3.14, and 14.76 f/s, respectively. The proposed CSPDarkNet53+PAFPN structure with improved tracking demonstrated superior performance, maintaining high accuracy while achieving 2.4× and 4.7× the speed of comparison models, validating its effectiveness for citrus fruit load estimation on edge devices.

4.2.2 Quantitative Numerical Fitting Analysis of Citrus Fruit Load Estimation Quantitative fitting analysis between algorithm-estimated and manually measured values further evaluated performance. Fitting results are shown in [Figure 9: see original paper]. The proposed model's RMSE is 4.1713, which is 47.61% and 22.94% lower than FairMOT(ResNet34) and FairMOT(HRNet18), respectively. The R^2 between estimated and ground truth values is 0.9858, outperforming comparison models and indicating excellent correlation.

5 Conclusion

This study proposes a lightweight model for edge computing devices to automatically estimate on-tree citrus fruit load from videos. The model employs CSPDarkNet53+PAFPN as the feature extraction network to achieve faster inference and lower complexity while maintaining feature representation capability. The Byte algorithm improves FairMOT's data association strategy during fruit tracking, enabling predictive tracking of citrus fruits to enhance estimation accuracy. Experimental results demonstrate 91.61% AEP and 14.76 f/s processing speed, with R^2 of 0.9858 and RMSE of 4.1713. The model's parameters, FLOPs, and size are 5.01 M, 36.44 G, and 70.2 MB, respectively, proving superior performance and lower complexity than alternatives. The orchard fruit load remote monitoring system based on this algorithm meets the requirements for mobile platform-based fruit load monitoring, providing technical support for automated orchard productivity analysis. Future research will enrich data resources, further improve model performance, and explore more efficient methods for additional fruit varieties.

Conflict of Interest Statement: The authors declare no conflicts of interest.

References: [1] FENG A J, ZHOU J F, VORIES E D, et al. Yield estimation in cotton using UAV-based multi-sensor imagery[J]. Biosystems engineering, 2020, 193: 101-114. [2] KURTULMUS F, LEE W S, VARDAR A. Green citrus detection using 'eigenfruit', color and circular Gabor texture features under natural outdoor conditions[J]. Computers and electronics in agriculture, 2011, 78(2): 140-149. [3] QURESHI W S, PAYNE A, WALSH K B, et al. Machine vision for counting fruit on mango tree canopies[J]. Precision agriculture, 2017, 18(2): 224-244. [4] ZHOU R, DAMEROW L, SUN Y R, et al. Using colour features of cv. 'Gala' apple fruits in an orchard in image processing to predict

yield[J]. Precision agriculture, 2012, 13(5): 568-580. [5] ANNAMALAI P, LEE W S. Citrus yield mapping system using machine vision[C]//2003 ASAE Annual Meeting. St. Joseph, MI, USA: American Society of Agricultural and Biological Engineers, 2003: 1. [6] STAJNKO D, RAKUN J, BLANKE M. Modelling apple fruit yield using image analysis for fruit colour, shape and texture[J]. European journal of horticultural science, 2009, 74(6): 260-267. [7] DORJ U O, LEE M, YUN S S. An yield estimation in citrus orchards via fruit detection and counting using image processing[J]. Computers and electronics in agriculture, 2017, 140: 103-112. [8] SA I, GE Z Y, DAYOUB F, et al. DeepFruits: A fruit detection system using deep neural networks[J]. Sensors, 2016, 16(8): 1222. [9] CHEN S W, SHIVAKUMAR S S, DCUNHA S, et al. Counting apples and oranges with deep learning: A data-driven approach[J]. IEEE robotics and automation letters, 2017, 2(2): 781-788. [10] BARGOTI S, UNDERWOOD J. Deep fruit detection in orchards[C]//2017 IEEE International Conference on Robotics and Automation (ICRA). Piscataway, NJ, USA: IEEE, 2017: 3626-3633. [11] HÄNI N, ROY P, ISLER V. A comparative study of fruit detection and counting methods for yield mapping in apple orchards[J]. Journal of field robotics, 2020, 37(2): 201-217. [12] LI Z J, YANG S H, SHI D S, et al. Yield estimation method of apple tree based on improved lightweight YOLOv5[J]. Smart agriculture, 2021, 3(2): 100-114. [13] KESTUR R, MEDURI A, NARASIPURA O. MangoNet: A deep semantic segmentation architecture for a method to detect and count mangoes in an open orchard[J]. Engineering applications of artificial intelligence, 2019, 77: 73-81. [14] GAO F F, WU Z C, SUO R, et al. Apple detection and counting using real-time video based on deep learning and object tracking[J]. Transactions of the Chinese society of agricultural engineering, 2021, 37(21): 217-224. [15] WANG Z L, WALSH K, KOIRALA A. Mango fruit load estimation using a video based MangoYOLO-kalman filter-hungarian algorithm method[J]. Sensors, 2019, 19(12): 2742. [16] LUO W H, XING J L, MILAN A, et al. Multiple object tracking: A literature review[J]. Artificial intelligence, 2021, 293: 103448. [17] RAKAI L, SONG H S, SUN S J, et al. Data association in multiple object tracking: A survey of recent techniques[J]. Expert systems with applications, 2022, 192: 116300. [18] TU S Q, TANG Y J, LI C J, et al. Behavior recognition and tracking of group-housed pigs based on improved ByteTrack algorithm[J]. Transactions of the Chinese society for agricultural machinery, 2022, 53(12): 264-272. [19] ZHANG Y F, WANG C Y, WANG X G, et al. FairMOT: On the fairness of detection and re-identification in multiple object tracking[J]. International journal of computer vision, 2021, 129(11): 3069-3087. [20] ZHANG Y F, SUN P Z, JIANG Y, et al. ByteTrack: Multi-object tracking by associating every detection box[C]//European Conference on Computer Vision. Berlin, Germany: Springer, 2022: 1-21. [21] WU H. Pedestrian multi-target tracking method based on YOLOX and person re-identification[J]. Automation & instrumentation, 2023, 38(3): 59-62, 67. [22] OUYANG W L, WANG X G, ZENG X Y, et al. DeepID-Net: Deformable deep convolutional neural networks for object detection[C]//2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway, NJ, USA: IEEE, 2015: 2403-2412. [23] REDMON J, FARHADI A. YOLOv3: An incremental improve-

ment[EB/OL]. arXiv: 1804.02767, 2018. [24] WANG C Y, MARK LIAO H Y, WU Y H, et al. CSPNet: A new backbone that can enhance learning capability of CNN[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Piscataway, NJ, USA: IEEE, 2020: 1571-1580. [25] WEI J, LI Z Q, XU E Y, et al. Research on hedge recognition based on DA2-YOLOv4 algorithm[J]. Journal of Chinese agricultural mechanization, 2022, 43(9): 122-130. [26] WANG C Y, BOCHKOVSKIY A, LIAO H Y M. Scaled-YOLOv4: Scaling cross stage partial network[C]//2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway, NJ, USA: IEEE, 2021: 13024-13033. [27] GÜNEY E, BAYILMIŞ C, ÇAKAN B. An implementation of real-time traffic signs and road objects detection based on mobile GPU platforms[J]. IEEE access, 2022, 10: 86191-86203. [28] LIN T Y, DOLLÁR P, GIRSHICK R, et al. Feature pyramid networks for object detection[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Piscataway, NJ, USA: IEEE, 2017: 936-944. [29] LIU S, QI L, QIN H F, et al. Path aggregation network for instance segmentation[C]//2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway, NJ, USA: IEEE, 2018: 8759-8768. [30] SUN Z Q, CHEN B C, CUI X B, et al. Strip steel surface defect detection by YOLOv5 algorithm fusing frequency domain attention mechanism and decoupled head[J]. Journal of computer applications, 2023, 43(1): 242-249.

Note: Figure translations are in progress. See original paper for figures.