agriculture

MDPI

Article

Research on Apple Detection and Tracking Count in Complex Scenes Based on the Improved YOLOv7-Tiny-PDE

Dongxuan Cao [1,†], Wei Luo [1,2,3,†], Ruiyin Tang [1,2,3], Yuyan Liu [1,2,3,*], Jiasen Zhao [1], Xuqing Li [1,2,3] and Lihua Yuan [1,2,3]

[1] North China Institute of Aerospace Engineering, Langfang 065000, China; caodx@stumail.nciae.edu.cn (D.C.); luowei@radi.ac.cn (W.L.); tangry@nciae.edu.cn (R.T.); zjs@stumail.nciae.edu.cn (J.Z.); meililixuqing@163.com (X.L.); ylh20070901@163.com (L.Y.)
[2] Aerospace Remote Sensing Information Processing and Application Collaborative Innovation Center of Hebei Province, Langfang 065000, China
[3] National Joint Engineering Research Center of Space Remote Sensing Information Application Technology, Langfang 065000, China
* Correspondence: lyy@nciae.edu.cn; Tel.: +86-133-9316-3958
† These authors contributed equally to this work.

Abstract: Accurately detecting apple fruit can crucially assist in estimating the fruit yield in apple orchards in complex scenarios. In such environments, the factors of density, leaf occlusion, and fruit overlap can affect the detection and counting accuracy. This paper proposes an improved YOLOv7-Tiny-PDE network model based on the YOLOv7-Tiny model to detect and count apples from data collected by drones, considering various occlusion and lighting conditions. First, within the backbone network, we replaced the simplified efficient layer aggregation network (ELAN) with partial convolution (PConv), reducing the network parameters and computational redundancy while maintaining the detection accuracy. Second, in the neck network, we used a dynamic detection head to replace the original detection head, effectively suppressing the background interference and capturing the background information more comprehensively, thus enhancing the detection accuracy for occluded targets and improving the fruit feature extraction. To further optimize the model, we replaced the boundary box loss function from CIOU to EIOU. For fruit counting across video frames in complex occlusion scenes, we integrated the improved model with the DeepSort tracking algorithm based on Kalman filtering and motion trajectory prediction with a cascading matching algorithm. According to experimental results, compared with the baseline YOLOv7-Tiny, the improved model reduced the total parameters by 22.2% and computation complexity by 18.3%. Additionally, in data testing, the *p*-value improved by 0.5%; the R-value rose by 2.7%; the mAP and F1 scores rose by 4% and 1.7%, respectively; and the MOTA value improved by 2%. The improved model is more lightweight and can preserve a high detection accuracy well, and hence, it can be applied to detection and counting tasks in complex orchards and provides a new solution for fruit yield estimation using lightweight devices.

Keywords: smart orchard; improved YOLOv7-Tiny; occlusion; DyHead; EIoU; DeepSort; lightweight

Academic Editor: Yongchao Tian

Received: 20 January 2025
Revised: 10 February 2025
Accepted: 21 February 2025
Published: 24 February 2025

Citation: Cao, D.; Luo, W.; Tang, R.; Liu, Y.; Zhao, J.; Li, X.; Yuan, L. Research on Apple Detection and Tracking Count in Complex Scenes Based on the Improved YOLOv7-Tiny-PDE. *Agriculture* **2025**, *15*, 483. https://doi.org/10.3390/agriculture15050483

1. Introduction

In the era of Agriculture 4.0, precision orchard production achieves intelligent and refined management through smart technologies and decision support systems. The Internet of Things (IoT), drones, sensors, and satellite remote sensing technologies can

monitor soil moisture, climate changes, pests and diseases, and fruit growth well, providing accurate data collection in real time. Big data analytics and cloud computing platforms efficiently process these data, assisting farmers in production dynamic analyses and trend forecasting. Control systems, such as smart irrigation, automated fertilization, and precision spraying, perform refined operations based on real-time data, significantly improving the resource utilization efficiency and crop yields. Furthermore, decision support systems based on big data and artificial intelligence help farmers optimize resource allocation, reduce costs, and enhance sustainability by analyzing historical data, weather forecasts, and crop growth models. Precision orchard production not only enhances productivity but also promotes agricultural ecological protection, driving the development of efficient, green, and intelligent agriculture. Rural revitalization has been vigorously promoted and small agriculture has been rapidly developing, and against such a backdrop, precise yield estimation during the fruit tree growth cycle and the intelligent management of orchards have become key research topics. Apple, as a common fruit with an extremely large growth area and sales, is a significant focus. As reported by the United Nations Food and Agriculture Organization, China is the largest apple producer worldwide, with a planting area of 2,088,080 hectares and an annual output of 45.97 million tons in 2021, accounting for over 50% of the global apple production. However, China's apple industry faces challenges, such as labor shortages, high labor intensity, and low picking efficiency, necessitating the automation and intelligent transformation of the industry. The use of flying inspection robots and artificial intelligence for the regular visual inspections of apple orchards is becoming increasingly important, with machine-vision-based apple detection, tracking, and counting methods emerging as key technologies. As machine vision, robotics, and artificial intelligence technologies continue to evolve, related research has gradually been applied to apple pruning, yield estimation, and harvesting, and research into intelligent fruit-picking technologies is gaining momentum [1–4]. However, a high fruit growth density in apple orchards, particularly in non-structured environments with low-stem-density planting [5], results in fruit often being obscured by branches and leaves, leading to significant overlapping and occlusion. This complicates the target detection and recognition, reduces the accuracy, and presents a challenge that must be addressed in intelligent management.

To address the challenges in fruit detection, researchers put forward various solutions by combining traditional image processing and machine learning algorithms. Traditional techniques for fruit detection typically focus on feature extraction, using color, shape, and texture, which, however, exhibit a low detection accuracy and poor generalization in complex scenarios, such as lighting variations, fruit occlusion [6], and environmental changes. For instance, Gongal et al. [7] used RGB-to-HIS conversion, histogram equalization, and threshold segmentation for apple recognition. Wang et al. [8] applied wavelet transforms to reduce lighting effects in fruit images with complex backgrounds. Chaivivatrakul et al. [9] introduced a texture analysis for detecting green fruits affected by lighting. Bulanon et al. [10] enhanced the red channel to improve apple target extraction. Some researchers also incorporated human observation or wind-blown leaves to reduce occlusion [11], and others focused on color features [12] or local textures [13] using support vector machines for detection [14]. Mai et al. [15] used the Log-Hough transform to extract apple shape features. Lin et al. [16] developed a detection algorithm that combines color, depth, and shape but struggles with occlusion. Lv et al. [17] utilized R-G color features for segmentation, while Luo et al. [18] applied k-means clustering for grape cluster detection. Si et al. [19] introduced apple segmentation using the red–green difference and template matching. Moallem et al. [20] combined morphological methods with a Mahalanobis distance classifier for stem detection. Despite progress, these traditional methods face

limitations in handling challenges, including lighting changes, complex backgrounds, and occlusion, often resulting in a low recognition accuracy and poor boundary localization. These issues highlight the need for improvements in robustness and adaptability in fruit detection systems.

Traditional fruit image detection techniques and machine learning algorithms face limitations in complex environments, making them unsuitable for practical applications. Deep convolutional neural networks (CNNs) have become the dominant approach in fruit target detection due to their robustness and generalization capabilities [21]. CNN-based detection algorithms are typically categorized into two-stage and single-stage models. Two-stage algorithms classify the generated candidate regions using a CNN to detect targets, with notable examples being Fast R-CNN and Faster R-CNN. For instance, Häni et al. [22] combined deep learning with a semi-supervised Gaussian mixture model for high-precision apple detection in natural environments. In the study by Xiong et al. [23], Faster R-CNN was employed for the detection of citrus fruits under various conditions, where it achieved detection accuracies of 77.45%, 73.53%, and 82.58% for different lighting conditions, sizes, and quantities. Peng et al. [24] improved fruit detection using SSD based on ResNet-101, while Juntao et al. [23] achieved a mean average precision (mAP) of 85.49% for citrus fruit detection using Faster R-CNN. Sun et al. [25] applied ResNet50 for Faster R-CNN to detect tomatoes. However, while two-stage algorithms perform well in unobstructed scenarios, they struggle with dense fruit and occlusion, exhibiting lower robustness and generalization in such settings. Additionally, their high computational complexity and slower detection speed limit their practicality. Thus, improving the detection accuracy under occluded conditions and enhancing real-time performance remain significant challenges for practical applications.

The YOLO (You Only Look Once) algorithm exhibits excellent real-time performance and accuracy, making it commonly used for fruit target detection and widely applied in fruit detection, yield estimation, and plant trait research. To address challenges such as occlusion in complex environments, various improvements were proposed to enhance the detection performance. For example, Chu et al. [26] developed a region-based detection model to handle severe occlusion and a high amount of overlap. In their study [27], YOLOv7 added with a small-object detection layer and lightweight convolutions presented improved citrus fruit detection accuracy. Praveen et al. [28] optimized YOLOv5 by integrating adaptive pooling and attribute enhancement, particularly enhancing apple detection in complex scenes. Li et al. [29] obtained an improved YOLOv4-Tiny-based model by combining attention mechanisms with multi-scale predictions to enhance the way occluded and small targets were recognized. Lai et al. [30] developed a YOLOv7 model for pineapple recognition, incorporating the SimAM attention module and replacing NMS with soft-NMS to boost the accuracy. Tian et al. [31] proposed YOLOv3-dense, which improved the detection accuracy for overlapping and occluded apples. Zhou et al. [32] introduced a fusion method integrating visual perception and image processing, ensuring that YOLOv7 bounding boxes for oil tea fruits matched extracted centroid points. Ji et al. [33] improved YOLOX by integrating the Shufflenetv2 attention mechanism and CBAM, enhancing apple detection. Wu et al. [34] combined YOLOv7 with an enhanced dataset, constructing the DA-YOLOv7 model for tea fruit recognition, achieving over 96% accuracy. Liu et al. [35] used CA attention and BiFPN to improve YOLOv5 with a zoom loss function. Wang et al. [36] applied variant convolutions and SE attention in YOLOv7-Tiny to improve the detection accuracy for chili peppers at different maturity stages. Zhao et al. [37] combined CSPNet and residual modules in YOLOv3 to enhance the apple detection in complex environments. Yang et al. [38] adopted an enhanced CenterNet network for the enhancement of the detection speed and accuracy for multi-apple detection in dense scenes.

Researchers developed various methods for object tracking and counting in recent years. Kuhn et al. introduced the SORT algorithm, which uses the intersection over unit (IOU) and the Hungarian algorithm to associate targets across frames in real time [39]. Henriques et al. proposed the KCF algorithm, which tracks targets by extracting features from the initial frame and applying regression methods in subsequent frames [40]. Xu Liu et al. used semantic segmentation to determine apple center positions, combined with 3D reconstruction for matching and counting; however, this method is time-consuming [41]. Stein et al. and Bargoti and Underwood adopted multi-view imaging and the Hungarian algorithm for fruit tracking [42,43]. Wojke improved the DeepSort algorithm to enhance the tracking accuracy in occluded environments and reduce the frequency of ID switching [44]. Halstead et al. applied IOU to track chili pepper fruit counting [45]. Bhattarai et al. designed a VGG16-based method to estimate apple counts from single images [46]. Nicolai Hani et al. tended to simplify the problem to apple cluster detection and classification based on the number of apples within the clusters [47]. These studies provide various methods for accurate fruit tracking and counting; however, there is still potential for improvements in real-time processing and accuracy.

The YOLO model is characterized by strong accuracy, fast recognition, and ease of deployment in object detection. However, in complex natural environments, such as those with a high density of apple fruit, leaf occlusion, and fruit overlap, existing object detection and tracking algorithms still face challenges. Specific issues include the following: (1) In complex scenarios, such as occlusion, the target detection network suffers from significant parameter redundancy and computational load. (2) In unstructured apple orchard environments, the model is incapable of extracting features of targets at varying scales well, leading to missed or misidentified target fruits and a low accuracy when dealing with occlusion, overlap, or small objects. (3) In complex scenes, the model's convergence speed is slow, and its optimization capability is inadequate. (4) There are also issues with tracking prediction mismatches across video frames due to fruit occlusion by leaves or surface changes. On these accounts, existing algorithms exhibit poor effectiveness in practical, complex environments.

To ensure that apple fruit detection performs better in complex scenes and satisfies the requirements of lightweight edge devices, the existing model framework shall be optimized. Therefore, with the aim of achieving the accurate and fast recognition and counting of apples in complex environments, this study developed an improved YOLOv7-Tiny-PDE method based on the YOLOv7-Tiny model to handle apple fruit detection and tracking.

This study made contributions primarily from four perspectives:

(1) Partial convolution (PConv) was used to replace the simplified efficient layer aggregation network (ELAN) in the backbone network, effectively reducing the network parameters and redundant computations while maintaining the detection accuracy.

(2) The original detection head was replaced with a dynamic detection head (DyHead), which effectively suppresses background interference and captures background features more comprehensively, thereby improving the detection accuracy of occluded targets at different scales and enhancing the ability to extract fruit features.

(3) The boundary box loss function complete intersection over union (CIOU) loss was replaced with efficiency intersection over union (EIOU) loss, which minimizes the differences in width and height between the predicted and ground truth boxes, thereby accelerating the model's convergence speed and effectively improving the optimization performance.

(4) The improved model was combined with the DeepSort algorithm and further integrated with the Kalman filter (KF) state prediction algorithm, the motion-based cascade

matching algorithm, and the Hungarian algorithm to achieve the precise tracking and counting of fruit.

2. Materials and Methods

2.1. Image Data Collection

The apple image data used in this study were obtained from the Shengfengyuan apple-picking orchard (Laiwu High-Tech Zone, Jinan City, Shandong Province, coordinates: 36°14′22″ N, 117°48′30″ E). Data collection was completed at 10:00 A.M. on 2 November 2024. To strengthen the samples' representativeness, broaden the model's applicability, and ensure device compatibility, we utilized drones for the image acquisition. The drone system equipped with an Intel RealSense D435i stereo camera (Chengdu Bobei Technology Co., Ltd., Chengdu, China) served for obtaining sensing and depth data. The system is characterized by its lightweight design, wide field of view, high depth accuracy, and excellent stability. We employed various shooting conditions, including multiple occlusion scenarios, angles, and distances. During image capture, the distance between the camera and apple fruit was maintained between 0.5 and 2.0 m. The shooting positions were precisely aligned to match the apples in their natural growth environment, accounting for various real-world growth conditions, including different angles (upward and downward), lighting conditions (backlight and front light), and occlusion scenarios (leaf branch occlusion and fruit occlusion). All images were saved in the JPG format. Figure 1 illustrates the partial collection results. A total of 594 apple-related images with a resolution of 1600 × 1600 and a 10 min video segment with a resolution of 1920 × 1440 were captured.

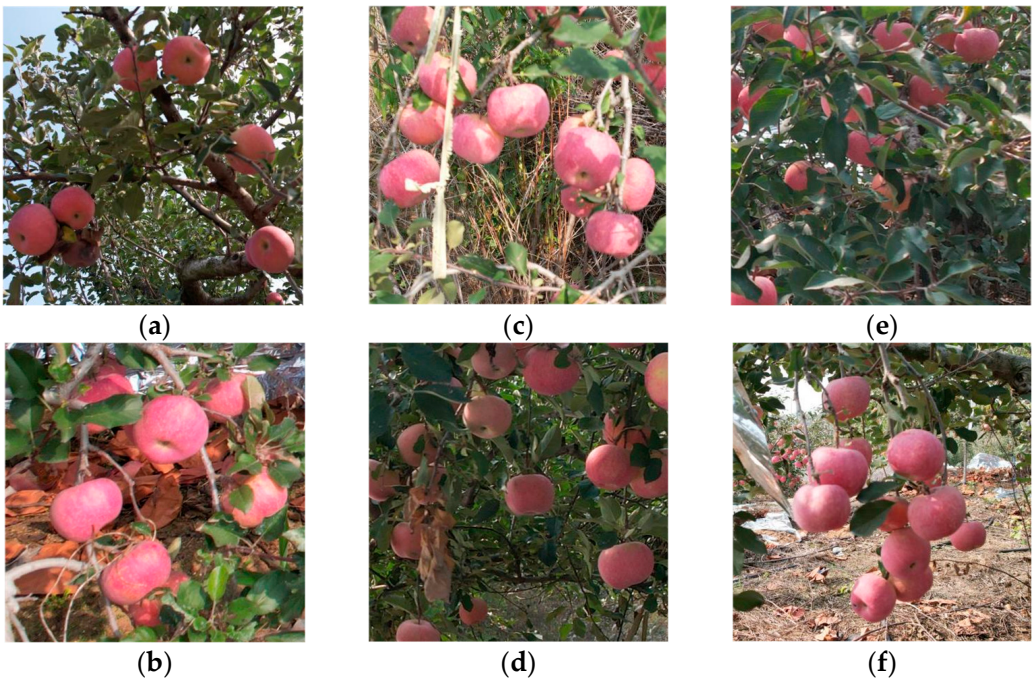

Figure 1. Apple images collected from the orchard, including the (**a**) upward view, (**b**) downward view, (**c**) frontlit, (**d**) backlit, (**e**) leaf and branch occlusions, and (**f**) fruit occlusion.

2.2. Construction of the Dataset

In real-world natural environments, due to the presence of various application scenarios and multiple interference factors, such as stems, leaves, fruits, and occlusions caused by changes in lighting conditions, it is challenging for a model to achieve full coverage. With the aim of enhancing the robustness and generalization ability of the model while avoiding overfitting caused by insufficient training data, this study selected 200 images

with occlusion features from the original dataset and used PyTorch and OpenCV for the data augmentation. To further expand the dataset, this study applied various image augmentation techniques, including brightness adjustment, Gaussian noise, blurring, geometric transformations, image composition, and mosaicing. The hardware used for the image enhancement consisted of a laptop with an AMD Ryzen 7 8745H (3.80 GHz) CPU, 16 GB RAM, and an NVIDIA GeForce RTX 4060 8 GB GPU, running 64-bit Windows 11. The software environment included CUDA 12.0, PyTorch 2.3.1, PyCharm Community 2024.3, Python 3.8.2, Numpy 1.23.5, and OpenCV 4.10.0 (Lenovo Group Limited, Beijing, China). The data augmentation process is visualized in Figure 2.

Figure 2. Augmented apple images: (**a**) original image; (**b**) brightness adjustment; (**c**) geometric transformation; (**d**) Gaussian noise; (**e**) blur; (**f**) image composition; (**g**) mosaic.

After performing the augmentation on the representative images and combining them with the original images, we obtained 1200 image samples, which fell into the training set, validation set, and testing set in a 7:1:2 ratio, which consisted of 840, 120, and 240 images, respectively. For the test set, we selected 100 images with a relatively sparse distribution of fruit to form the lightly occluded test set A. Figure 3 displays some sample images from test set A. Similarly, we selected 100 images with a denser distribution of fruit to form the densely occluded test set B, with sample images displayed in Figure 4.

Figure 3. Sample images of lightly occluded apples.

Figure 4. Sample images of densely occluded apples.

The open-source tool LabelImg was employed to accurately annotate the targets in the aforementioned images, precisely drawing rectangular bounding boxes around the contours of the apples. Specific to targets partially occluded by leaves, branches, or other fruit, their visible contours were labeled as accurately as possible according to manual experience. Moreover, if over 80% of an apple's surface area was occluded, in principle, the target was not annotated. The label for the target fruit was uniformly named "Apple_object." After the annotation, the data were directly saved as YOLO—format.txt label files corresponding to the images. The images were then classified into training, validation, and test sets in accordance with the final 7:1:2 ratio for model training, with the annotation results displayed in Figure 5.

Figure 5. Apple images annotated using LabelImg. The green rectangular boxes are labeled as target bounding boxes.

2.3. Improved YOLOv7-Tiny-PDE Network Detection Model

2.3.1. Principle of the Network Detection Model

The YOLOv7 series includes 7 versions, i.e., YOLOv7, YOLOv7-d6, YOLOv7-e6, YOLOv7-e6e, YOLOv7-w6, YOLOv7-Tiny, and YOLOv7-x, that differ in network depth and width. Given the requirements for deployment on embedded devices and the need to balance a lightweight design with accuracy, we selected YOLOv7-Tiny as the base model for this study.

YOLOv7 is an anchor-based, single-stage object detection algorithm, and YOLOv7-Tiny is an optimized version that retains the scaling method of the YOLOv7 composite model. It uses an ELAN [48] to replace the extended efficient layer aggregation network (E-ELAN), resulting in fewer model parameters and a faster detection speed while maintaining accuracy. Relying on such a design, YOLOv7-Tiny is particularly suitable for the real-time fruit detection and deployment on embedded devices. YOLOv7-Tiny consists of three main components, namely, a backbone, neck, and head, with its network architecture shown in Figure 6. To achieve a lightweight design, YOLOv7-Tiny reduces the number of convolutions in the ELAN, MP, and SPPCSPC modules, replacing the CBL module with regular convolutions. Despite this, YOLOv7-Tiny fails to utilize the full integration of some advantageous modules in its structure, somewhat limiting the network's feature learning capability [49].

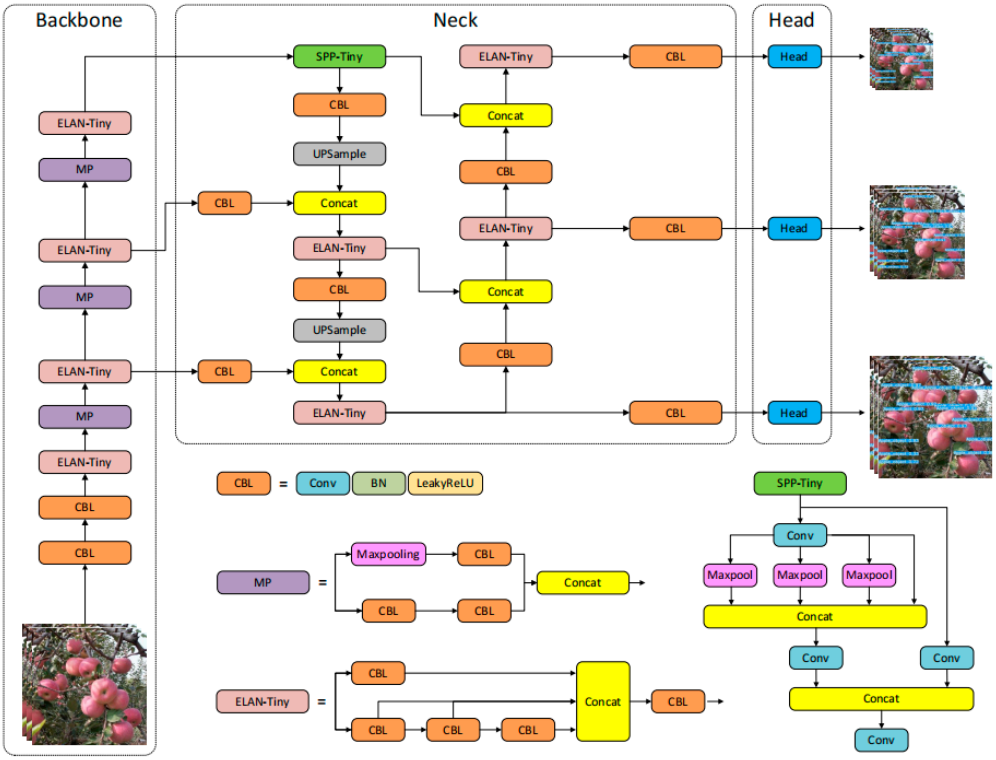

Figure 6. Structure diagram of the YOLOv7-Tiny model.

According to Figure 7 illustrating the network architecture of the improved model, despite being more streamlined than YOLOv7, YOLOv7-Tiny still contains numerous parameters and complicated model structures. For optimizing its deployment on embedded devices, we proposed several improvements to make YOLOv7-Tiny less complicated. Specifically, after using PConv to replace the CBL convolution in the original YOLOv7-Tiny backbone network, using DyHead to replace the original detection head, and substituting the CIOU with the EIOU loss function, we obtained the improved YOLOv7-Tiny-PDE model.

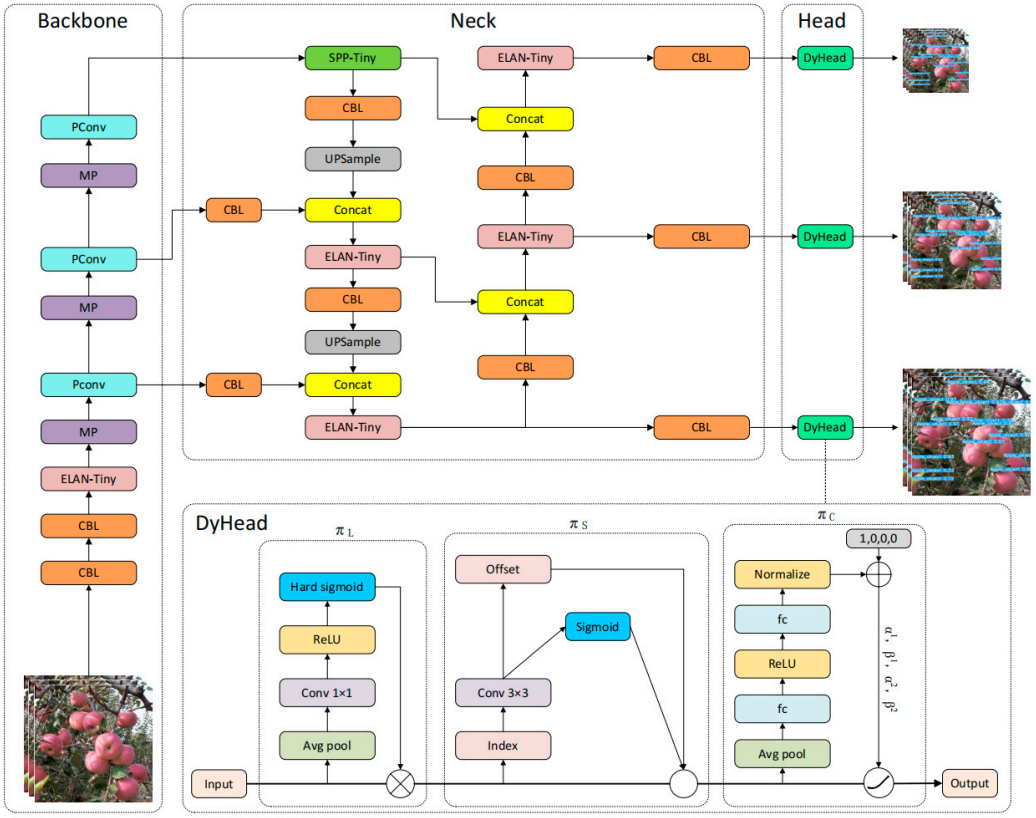

Figure 7. Structure diagram of the improved YOLOv7-Tiny-PDE model.

2.3.2. PConv Based on Fasternet

The number of floating-point operations per second (FLOPS) commonly serves as a standard to measure computational speed, and a higher value indicates stronger performance of the network hardware. However, the widely adopted architectures, such as conventional convolution, group convolution, and depthwise separable convolution, often face the challenge of lower FLOPS, primarily resulting from frequent memory access. When there are more floating-point operations (FLOPs), the model's computation becomes more complex; hence, the index usually serves to assess the model complexity. The original YOLOv7-Tiny backbone network relies on massive regular convolutions to extract features, resulting in more parameters and a larger computational load. Although there are fewer parameters and FLOPs after the kernel optimization, the frequency of memory accesses rises with the network width, which partially offsets the loss in accuracy.

PConv significantly reduces the number of FLOPs through performing computations on a limited number of channels. Compared with traditional convolutions, PConv reduces the memory access requirements, which is particularly beneficial for devices with input/output (I/O) limitations. Although PConv only processes some input channels, the retained channels continue to function in the subsequent pointwise convolution (PWConv) layers, ensuring that feature information flows across all channels. Therefore, this study opted to use PConv to replace the standard convolutions in the backbone network, with the aim to weaken the network complexity, as well as preserve the feature extraction capabilities. Figure 8c illustrates the operational mechanism of PConv.

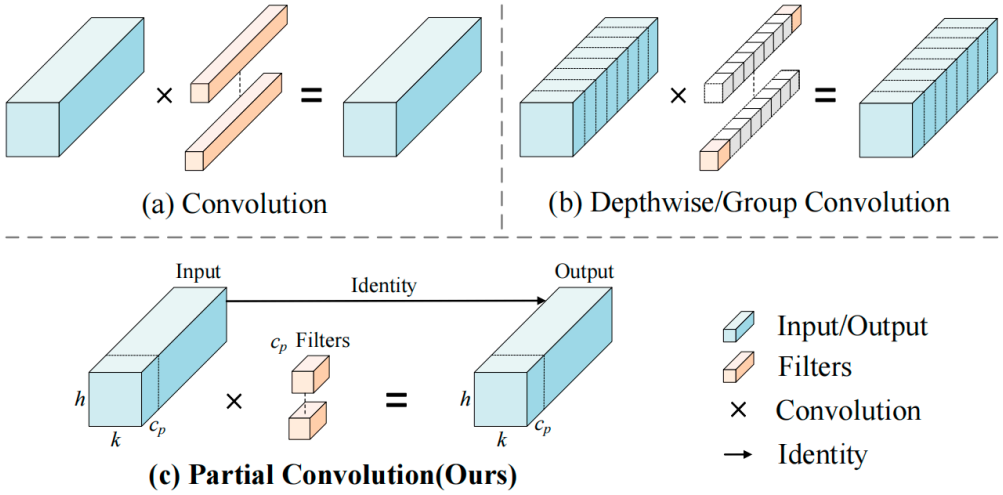

Figure 8. Convolution architectures: (**a**) conventional convolution; (**b**) depthwise/grouped convolution; (**c**) partial convolution.

As shown in Figure 8c, h and w denote the feature map's height and width, respectively; c refers to the total number of channels; and c_p denotes the number of channels actually used. The size of the convolution kernel is $k \times k$. Unlike conventional convolution operations, PConv performs spatial feature extraction only on selected input channels while keeping the other channels unchanged. When dealing with continuous or conventional memory accesses, the first or last continuous channels represent the entire feature map and are used for the associated computations. Furthermore, the input data and feature maps typically have the same number of channels.

Under general conditions, the formulas below give the FLOP and memory access equations of PConv:

$$h \times w \times k^2 \times c_p^2, \tag{1}$$

$$h \times w \times 2c_p + k^2 \times c_p^2 \approx h \times w \times 2c_p, \tag{2}$$

Upon inspecting Equations (1) and (2), it becomes evident that when the partial rate is set to one-fourth, the number of FLOPs of PConv amounts to merely one-sixteenth of that of standard convolution (Conv). Simultaneously, the number of memory accesses for PConv is decreased to one-fourth of that of conventional convolution.

2.3.3. The "Dynamic" Target Detection Head DyHead

In object detection tasks, as the classification and localization exhibit complexity, the detection head shall adapt to features of different scales in order to effectively identify multi-scale objects. Dai et al. [50] proposed a DyHead, which significantly enhances the object detection performance by introducing an attention mechanism between feature layers, spatial locations, and output channels. Specifically, the DyHead can perceive the scale variations of objects at multiple levels, focus on key information spatially, and optimize features according to task requirements. This design improves the model's adaptability to various complex scenarios, demonstrating stronger robustness and accuracy when handling objects at varying scales and locations.

DyHead consists primarily of three attention mechanisms: π_L, π_S, and π_C (Figure 9). The DyHead module is realized by the combination of these three attention mechanisms. However, it is important to note that the stacking of the three attention mechanisms represents a single block, while the actual detection head is formed by stacking multiple such blocks. First, the input is processed through the π_L module, which uses a hard sigmoid activation function, ReLU activation, 1×1 convolution, and average pooling to extract local

features. Next, the input enters the π_S module, where the offset information is processed through a sigmoid activation function, followed by a 3×3 convolution operation and indexing for spatial feature extraction. Finally, the input passes through the π_C module, where the normalization is applied, followed by a fully connected (FC) layer and ReLU activation to generate the final output, which includes several parameters ($\alpha 1$, $\beta 1$, $\alpha 2$, and $\beta 2$) used for subsequent feature fusion and model optimization. This structure enables multi-level feature extraction and fusion through the collaborative operation of multiple modules.

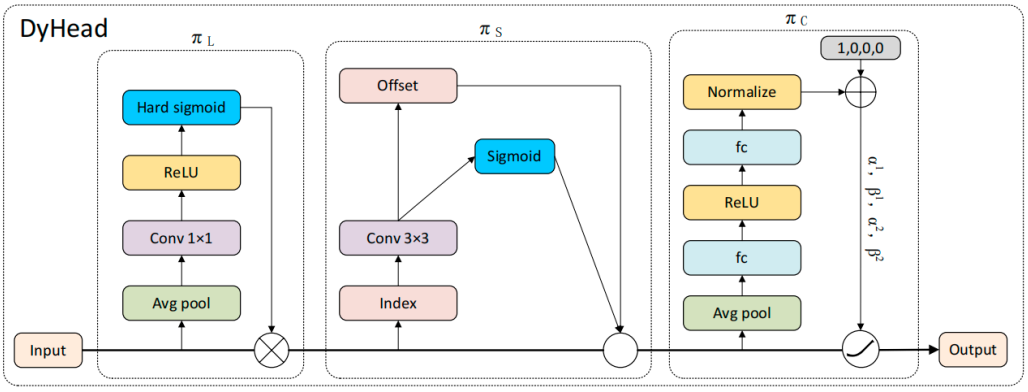

Figure 9. Structure of DyHead.

The scale-aware attention module dynamically integrates multi-scale information by taking into account the semantic importance of features at varying scales. The spatial-aware attention module achieves attention sparsification through deformable convolutions, and it performs multi-level feature fusion across varying spatial locations in regions with similar object features, enhancing the richness of feature representation. The task-aware attention module conducts average pooling for the elimination of feature dimensions, followed by two FC layers and normalization functions, which dynamically control the activation state (ON/OFF) of the feature channels. This enables joint learning and the generalization of the targets, thus optimizing the task-aware attention.

2.3.4. EIoU Loss Function

The original YOLOv7-Tiny network employs the CIoU loss function, extending the distance intersection over union (DIoU) loss by further incorporating the size loss of the detection box, including both the width and height discrepancies. This enhancement improves the alignment between the predicted and practical truth bounding boxes. The calculation formulas are presented as follows:

$$CIoU\_LOSS = 1 - CIoU, \tag{3}$$

$$CIoU = IoU - \frac{d_O^2}{d_C^2} - \alpha v, \tag{4}$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w^{p}}{h^{p}}\right)^2, \tag{5}$$

$$\alpha = \frac{v}{(1 - IoU) + v\prime}, \tag{6}$$

In these formulas, d_O represents the Euclidean distance between the center points of the predicted and ground truth bounding boxes, while d_C denotes the Euclidean distance of the diagonal of the smallest enclosing box covering both the predicted and practical truth boxes. The parameter v measures the consistency of the aspect ratio, where w^{gt} and h^{gt}, along with w^{p} and h^{p}, refer to the width and height of the practical and predicted boxes, respectively.

Therefore, the CIoU loss function considers the overlapping area, the center point distance, and the aspect ratio in the bounding box regression. Nevertheless, the difference in the aspect ratio (v) fails to accurately reflect the true variations between the width, height, and confidence, which limits the model's optimization efficiency to some extent. On this account, this study introduced the EIoU loss function to replace the CIoU. Based on the CIoU, the EIoU decomposes the aspect ratio factors by separately calculating the width and height differences between the predicted and practical truth boxes, accordingly optimizing the bounding box shape more precisely. The width–height loss is accompanied by directly reduced differences in width and height; as a result, the convergence becomes faster, and the localization accuracy is improved. The formula below interprets the specific calculation steps:

$$L_{EIoU} = L_{IoU} + L_{dis} + L_{asp} = 1 - \text{IoU} + \frac{\rho^2(b, b^{gt})}{c^2} + \frac{\rho^2(w, w^{gt})}{C_w^2} + \frac{\rho^2(h, h^{gt})}{C_h^2}, \quad (7)$$

where C_w and C_h denote the width and height of the minimum enclosing rectangle between the predicted and practical truth bounding boxes, respectively; ρ denotes the Euclidean distance between two points.

2.4. DeepSort Multi-Object Tracking Algorithm

2.4.1. The Principle of DeepSort

The traditional SORT (Simple Online and Realtime Tracking) algorithm, as the predecessor of the DeepSort algorithm, is characterized by its simplicity and efficiency. This algorithm utilizes the IoU metric for target association and relies on two core components, the KF and the Hungarian algorithm, to reduce ID-switching issues. Specifically, the KF encompasses the steps of prediction and update, with the former one estimating the current frame's (t) motion state (position and velocity) based on the previous frame's ($t - 1$) target motion state, while the latter one combines the current frame's detected target with the predicted results, using linear weighting for refinement. The Hungarian algorithm is responsible for matching the KF's predicted frames with the IoU-based detection results and assigning a unique ID to each successfully matched target, effectively solving the linear assignment problem for precise object tracking. However, in practical applications, when objects overlap or become occluded, the algorithm may experience ID switching during the matching process between detected and predicted frames. This can reduce the tracking accuracy and ultimately affect the final counting results.

To address the aforementioned issues, DeepSort integrates a SORT-framework-based re-identification (ReID) network model to extract appearance features from the detection boxes, and it employs a matching cascade strategy, effectively reducing the frequency of target ID switches. Specifically, DeepSort uses the Mahalanobis distance $d^{(1)}$ and the cosine distance $d^{(2)}$ as the cost matrices to evaluate the association between the prediction and detection results, as shown in Equations (8) and (9). These matrices are then used for target matching. If the matching fails, the target, as a new object, will be assigned a new ID; if successful, the target retains its original ID. The DeepSort algorithm significantly decreases the frequent switching of target IDs in complex occlusion scenarios, thereby enhancing the multi-object tracking and counting accuracy.

$$d_{(i,j)}^{(1)} = (d_j - y_i)^T S_i^{-1} (d_j - y_i), \quad (8)$$

where d_j denotes the j-th detection box, y_i represents the state vector of the i-th detection box, and S_i denotes the standard deviation matrix of the i motion trajectories.

$$d_{(i,j)}^{(2)} = min\left\{1 - r_j^T \, r_k^{(i)} \;\middle|\; r_k^{(i)} \in R_i\right\}, \tag{9}$$

In this equation, $d_{(i,j)}$ is the minimum cosine distance, r_j is the feature vector of the detection box, r_k is a feature vector that achieves a successful association over the next 100 frames, and R_i denotes the set of appearance features of the target.

2.4.2. The DeepSort Algorithm Combined with the Improved YOLOv7-Tiny-PDE Model

The flowchart in Figure 10 illustrates the application process of the improved YOLOv7-Tiny model combined with the KF prediction and object-matching algorithm in video tracking. First, video data are input, and boundary detection and feature detection are performed. Then, the target position is updated using KF prediction. If a successful match occurs, then the system updates the KF based on the matched detection results and outputs the result. If the match fails, then the Hungarian algorithm is used for target matching. Unmatched targets are further processed using a "matching cascade" method. Matched targets are marked as confirmed and undergo KF updates. For unconfirmed targets, if their age exceeds the maximum time threshold (max_age), they are deleted; otherwise, tracking and updates continue. Additionally, when a new target is detected, a new tracking trajectory is generated and updated.

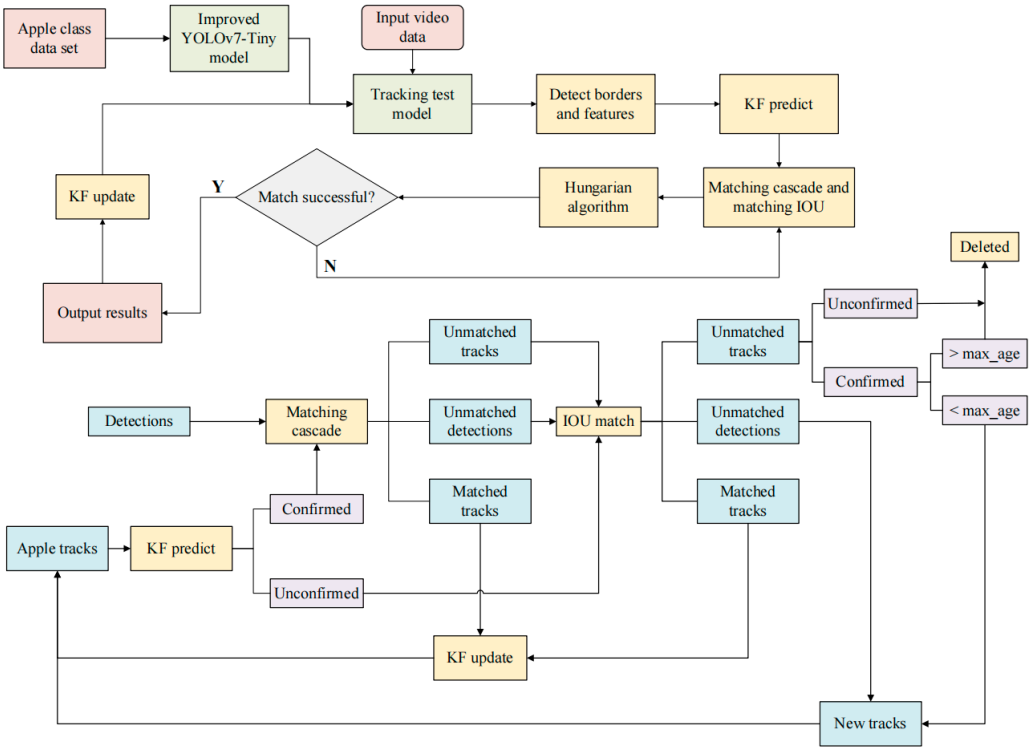

Figure 10. Flowchart of the improved YOLOv7-Tiny-PDE combined with DeepSort.

2.5. Model Training and Testing

2.5.1. Testing Environment and Parameter Settings

The network model here was constructed using the deep learning framework PyTorch, and the program was run on a computer (Lenovo Group Limited, Beijing, China) with the Windows 11 64-bit operating system. The hardware configuration for the training and testing included an AMD Ryzen 7 8745H processor with Radeon 780M Graphics, a clock speed of 3.80 GHz, 16 GB of RAM, and an 8 GB NVIDIA GeForce RTX 4060 Laptop

GPU. The software environment included CUDA version 12.0, cuDNN version 8.7, Numpy version 1.23.5, OpenCV version 4.10.0, and Python version 3.8.2. Other environmental configurations were consistent with the above setup.

2.5.2. Evaluation Metrics

We focused on evaluating the modified YOLOv7-Tiny-PDE model using the metrics of precision (P, %), recall (R, %), mean average precision (mAP), and $F1$ score, thereby verifying whether the proposed improvements were effective. The relevant calculation formulas are as follows:

$$P = \frac{TP}{TP + FP} \times 100\%, \tag{10}$$

$$R = \frac{TP}{TP + FN} \times 100\%, \tag{11}$$

$$AP = \int_0^1 P(R)dR, \tag{12}$$

$$mAP = \frac{1}{M}\sum_{k=1}^{M} AP(k) \times 100\%, \tag{13}$$

$$F1 = \frac{P \times R \times 2}{P + R}, \tag{14}$$

Based on the relationship between the actual class of a sample and the class predicted by the model, samples can be categorized into four types: true positive (TP), i.e., the cases where the sample is positive and correctly predicted as positive; false positive (FP), i.e., the cases where the sample is negative but incorrectly predicted as positive; false negative (FN), i.e., the cases where the sample is positive but incorrectly predicted as negative; and true negative (TN), i.e., the cases where the sample is negative and correctly predicted as negative. Here, k is the current class under consideration, and M denotes the total number of classes.

In Equations (10) and (11), precision P means the proportion of correctly predicted positive samples among all the samples predicted as positive. A higher P value indicates a greater proportion, resulting in a lower false positive rate. The recall R, however, is a metric measuring the proportion of actual positive samples under the correct identification of the model. A higher R value signifies a stronger detection ability of the model against positive samples, leading to a lower false negative rate.

The mAP is the average of the average precision (AP) values across all categories, positively reflecting the model's overall detection performance across different classes. The $F1$ score, however, is the harmonic mean of P and R, serving as a key metric for evaluating the model's overall recognition capability.

In multi-object tracking tasks, the multi-object tracking accuracy ($MOTA$) metric usually serves for assessing the performance of DeepSort. This metric comprehensively reflects the occurrence of false positives, missed detections, and ID switches during the tracking process.

The calculation formula for $MOTA$ is as follows:

$$MOTA = \frac{\sum_{i=1}^{n}\left(T_{P_i} - F_{P_i} - IDSW_i\right)}{\sum_{i=1}^{n}\left(T_{P_i} + F_{N_i}\right)}, \tag{15}$$

where T_{P_i} represents the number of tracks correctly associated with the true targets in the i-th frame; F_{P_i} indicates the number of tracks falsely reported in the i-th frame; F_{N_i} is the number of true targets in the i-th frame that are not tracked; and $IDSW_i$ represents the

number of identity switches that occur in the *i*-th frame, where the same true target is incorrectly assigned different IDs.

3. Results and Analysis

3.1. Comparison of Detection and Counting Results of Varying Detection Models

In natural apple orchards, apple fruit are often partially obscured due to mutual overlapping or occlusion by leaves and branches, leading to the loss of some contour information, making it more difficult to achieve fruit detection. As the degree of occlusion intensifies, much more fruit contour information will be lost, and the size information of the occluded regions will be reduced, which further complicates the detection task. Therefore, analyzing the model's detection accuracy under varying occlusion conditions is crucial. With the purpose of enhancing the model's performance in apple fruit detection, we selected several lightweight models, namely, MobileNetv2; ShuffleNetv2; and the YOLO series models YOLOv8n, YOLOv8s, YOLOv9s, YOLOv10s, and YOLOv11n. We also used mainstream object detection networks from the YOLOv7 series, namely, YOLOv7, YOLOv7x, and YOLOv7-Tiny. Additionally, we included the newly improved YOLOv7-Tiny-PDE, for a total of 11 models for comparative testing. During the training, all models used the same training dataset and maintained consistent hyperparameters. After the training, the best-performing network weights were selected for testing and evaluated on the same test set. The test set consisted of 420 apple images that covered various complex scenarios, and the comprehensive evaluation results are shown in Table 1.

Table 1. Detection results comparison between different models.

Model	Parameters	GFLOPs	P/%	R/%	mAP@0.5/%	mAP@0.95/%	F1 Measure
MobileNetv2	4,757,846	10.2	95.50	93.10	96.40	86.50	0.942
ShuffleNetv2	5,518,362	10.6	96.00	94.30	95.60	84.30	0.951
YOLOv8n	3,005,843	8.1	96.50	93.50	97.60	87.30	0.949
YOLOv8s	11,176,233	28.8	95.30	92.80	97.40	83.00	0.940
YOLOv9s	7,167,475	26.7	96.30	96.10	97.80	88.20	0.961
YOLOv10s	8,035,734	24.4	96.80	95.10	97.70	89.10	0.959
YOLO11n	2,582,347	6.6	96.70	93.70	97.80	86.00	0.951
YOLOv7	36,481,772	103.2	95.70	94.80	94.50	84.00	0.952
YOLOv7x	70,782,444	188.0	97.40	94.10	94.20	83.90	0.957
YOLOv7-Tiny	6,007,596	13.1	96.70	94.10	94.10	81.20	0.954
YOLOv7-Tiny-PDE	4,676,516	10.7	97.20	96.60	97.90	82.90	0.969

According to the experimental results, the YOLOv7-Tiny-PDE model could balance the lightweight design and detection performance well. Specifically, the model had a low parameter count (4.68M) and computational complexity (10.7 GFLOPs), ranking just behind the lightweight models, such as MobileNetv2, ShuffleNetv2, and YOLOv8n. At the same time, these values were significantly lower than those of the high-performance models, such as YOLOv9s and YOLOv10s, demonstrating its advantages in computational and storage resources. While maintaining a low computational complexity, YOLOv7-Tiny-PDE exhibited an mAP@0.5 of 97.90%, ranking first among all the compared models, as well as significantly outperforming the other models, such as ShuffleNetv2 (95.60%) and YOLOv7x (94.20%), thereby proving its superior detection capability. Additionally, YOLOv7-Tiny-PDE achieved the highest values for the recall rate (R = 96.60%) and F1 score (0.969), indicating that it maintained a high detection accuracy while also achieving a low false-negative rate. Although YOLOv9s and YOLOv10s slightly outperformed in the two mAP metrics, their GFLOP values were 26.7 and 24.4, respectively, which were significantly higher than the computational complexity of YOLOv7-Tiny-PDE. Compared with the

original YOLOv7-Tiny model, the improved model demonstrated a 22.2% reduction in the total parameter count, a 18.3% reduction in computational complexity, an 0.5% increase in the *p*-value, a 2.7% increase in the recall rate, a 4% increase in the mAP@0.5 score, and 1.7% increase in the F1 score.

The complexity of the model is an important evaluation metric. In terms of the parameters, YOLOv7-Tiny-PDE had a parameter count of 4.68M, which is considered to be at a medium–low level; it is slightly higher than that of YOLOv8n and YOLOv11n, but obviously lower than that of MobileNetv2; ShuffleNetv2; and most other YOLO series models, such as YOLOv8s with 11.17M, YOLOv9s with 7.17M, and YOLOv10s with 8.04M, indicating that its parameter scale was relatively manageable. In terms of the GFLOPs, YOLOv7-Tiny-PDE had a value of 10.7, which was slightly higher than that of YOLOv8n and YOLOv11n and just behind that of the lightweight MobileNetv2 (10.2) and ShuffleNetv2 (10.6). It was significantly lower than that of YOLOv7 (103.2) and YOLOv7x (188.0), demonstrating its reliability in computational resource requirements.

Clearly, the improved YOLOv7-Tiny-PDE model exhibited a lower computational complexity and smaller model size while achieving a detection accuracy comparable to or even surpassing that of some high-performance models, demonstrating a balance between lightweight design and high precision. Furthermore, we randomly selected 10 apple images for fruit counting. Table 2 lists the detection-counting results of different models.

Table 2. Detection-counting results of apple fruit in images for different models.

Model	Number in Each Image	Manual Count	Detection Count	Detection Count Confidence
MobileNetv2	13, 15, 21, 12, 15, 13, 17, 21, 13, 22	171	162	0.947
ShuffleNetv2	13, 14, 22, 11, 15, 12, 17, 22, 12, 23	171	161	0.941
YOLOv8n	13, 14, 21, 11, 15, 13, 17, 22, 12, 22	171	160	0.936
YOLOv8s	13, 14, 24, 11, 15, 12, 16, 22, 13, 22	171	162	0.947
YOLOv9s	14, 14, 21, 11, 15, 13, 18, 22, 12, 22	171	162	0.947
YOLOv10s	14, 14, 21, 11, 15, 13, 16, 21, 12, 24	171	161	0.941
YOLO11n	14, 15, 21, 11, 16, 13, 17, 22, 13, 23	171	165	0.965
YOLOv7	13, 14, 21, 11, 15, 14, 15, 22, 12, 22	171	159	0.930
YOLOv7x	13, 14, 21, 11, 15, 13, 16, 21, 12, 22	171	158	0.924
YOLOv7-Tiny	13, 14, 21, 12, 16, 14, 17, 22, 12, 23	171	164	0.959
YOLOv7-Tiny-PDE	13, 14, 21, 13, 18, 14, 16, 22, 13, 23	171	167	0.977

3.2. Comparative Experiments of Different Detection Models in Various Environments

In unstructured orchards, the scene complexity is significantly higher. Considering the inherent growth characteristics of fruit trees, the fruits can usually be obscured by branches and leaves or partially damaged. Additionally, fruits of varying sizes and shapes tend to overlap, causing incomplete contour information for some fruits. During the feature extraction process, information from the occluded regions is further reduced, which undoubtedly increases the difficulty of detection. Furthermore, variations in field lighting conditions, such as direct sunlight and backlighting, introduce additional interference, making fruit recognition and counting even more challenging. Against such a backdrop, this study compared the performance of the YOLOv7x model, which is known for its superior detection accuracy; the lightweight YOLOv7-Tiny model; and the improved YOLOv7-Tiny-PDE model. Figures 11–14 highlight the detection failure regions in different environments using green and red bounding boxes.

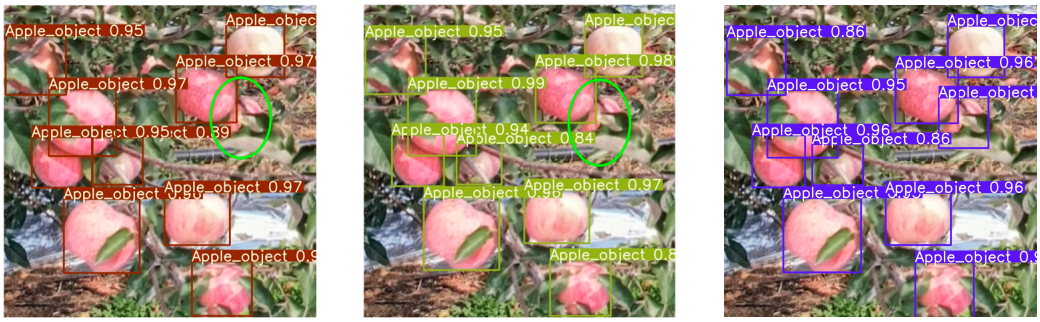

Figure 11. Detection performance under light occlusion.

Figure 12. Detection performance under heavy occlusion.

Figure 13. Recognition performance under different lighting conditions.

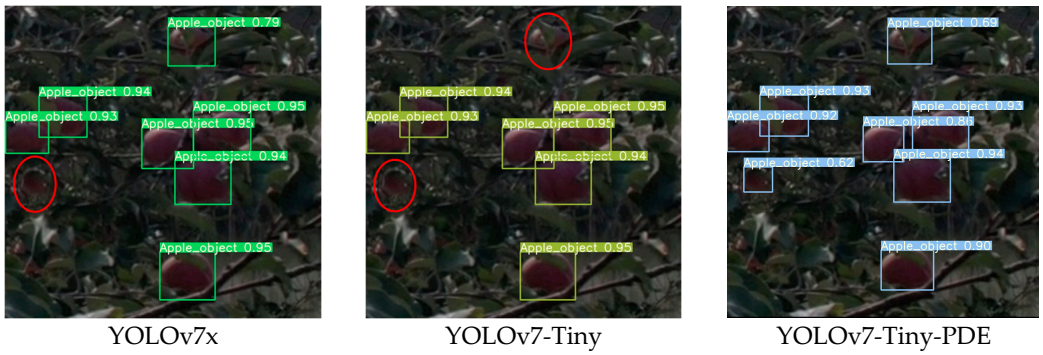

Figure 14. Recognition performance under backlit conditions.

3.2.1. Comparison of Detection Results Under Varying Occlusion Conditions

We analyzed the accuracy of the models in recognizing apples with different levels of occlusion, aiming at assessing the detection performance of apples under various occlusion

conditions. The lightly occluded test set A and the heavily occluded test set B were selected as the occlusion test datasets. Table 3 lists the detection results of the three models.

Table 3. Detection result comparison under varying occlusion conditions.

Model	Occlusion Condition	P/%	R/%	mAP@0.5/%	F1
YOLOv7x	Light occlusion	99.4	98.7	99.4	0.990
YOLOv7-Tiny		98.8	99.4	99.5	0.991
YOLOv7-Tiny-PDE		98.1	99.1	99.3	0.986
YOLOv7x	Dense occlusion	98.0	97.0	97.7	0.975
YOLOv7-Tiny		97.1	96.6	97.2	0.968
YOLOv7-Tiny-PDE		98.4	96.1	97.8	0.972

As shown in Figure 11, in scenarios with slight occlusion, both YOLOv7x and YOLOv7-Tiny could identify relatively prominent targets. However, due to the occlusion, the networks exhibited limited capabilities in extracting the contour features of the fruit, which resulted in some missed detections. In contrast, the improved YOLOv7-Tiny-PDE, enhanced with the DyHead module, demonstrated significantly improved sensitivity and representation abilities for target region features. This enhancement allowed the network to more accurately extract fruit boundary features from complex backgrounds, which effectively increased the detection accuracy of the apple positions.

As shown in Figure 12, occlusion leads to the loss of certain target features, and both YOLOv7-Tiny and YOLOv7x still exhibited missed detections under the same occlusion conditions. When the occlusion area exceeded 80%, the models struggled to effectively locate the fruit's boundaries and contours. In contrast, the improved YOLOv7-Tiny-PDE excelled at accurately detecting the occluded fruit, demonstrating a superior feature extraction capability. In summary, the proposed YOLOv7-Tiny-PDE network exhibited strong detection performance under varying degrees of occlusion.

3.2.2. Detection Result Comparison Under Varying Lighting Conditions

To assess whether the model was robust under varying lighting conditions, we selected 100 images that faced the light and 100 images that faced away from the light from the original dataset to form test sets C and D. During the field validation, we observed that the lighting conditions also affected the recognition and counting of the fruit. Table 4 lists the detection results. Under frontlit conditions, all three models exhibited a good detection performance. However, under backlit conditions, the insufficient lighting severely impacted the fruit's color and texture features; hence, all three models exhibited a weaker detection performance.

Table 4. Detection result comparison under different lighting conditions.

Model	Light Condition	P/%	R/%	mAP@0.5/%	F1
YOLOv7x	Bright light	98.5%	96.9%	97.2%	0.977
YOLOv7-Tiny		97.2%	96.7%	97.0%	0.969
YOLOv7-Tiny-PDE		97.7%	96.3%	97.2%	0.971
YOLOv7x	Dim light	98.8%	95.7%	96.1%	0.972
YOLOv7-Tiny		98.1%	94.6%	95.1%	0.963
YOLOv7-Tiny-PDE		98.3%	95.7%	97.1%	0.970

As shown in Figure 13, under well-lit conditions, when there was mutual occlusion between the leaves and fruit, the detection results of YOLOv7x and YOLOv7-Tiny still exhibited missed detection issues. In contrast, the improved YOLOv7-Tiny-PDE did not

show any missed detections, which could be attributed to its better capture of the semantic color information of the fruit.

As shown in Figure 14, under backlit conditions, YOLOv7-Tiny presented an obviously worse recognition performance versus the other two networks, with two apple targets being missed. YOLOv7x also failed to detect one fruit. In contrast, the improved YOLOv7-Tiny-PDE demonstrated better overall detection performance in this scenario, with no missed detections.

The above findings indicate that under low-light conditions, the network could not extract effective texture information of the fruit well due to their less pronounced color features. Furthermore, overlap and occlusion between the leaves and fruit exacerbated the detection challenge, which led to some fruit not being accurately recognized. Hence, the YOLOv7-Tiny-PDE network demonstrated superior detection performance under complex lighting conditions and exhibited higher recognition rates and lower missed detection rates in various lighting scenarios; hence, it applies better to fruit detection tasks in complicated orchard environments.

3.3. Ablation Study

This study employed a controlled variable method for comparatively examining the baseline model, with the objective of verifying whether the proposed model could effectively detect apple fruit. In this ablation study, a detection speed comparison metric was introduced, which included the total time for image preprocessing, non-maximum suppression (NMS), and inference. A lower value for this metric indicates better overall model performance. The five models were compared, with the results listed in Table 5. Based on the YOLOv7-Tiny network, we progressively set up ablation experiments. Specifically, Model 1 served as the baseline, while Models 2–4 introduced PConv, DyHead, and the EIoU loss function into the YOLOv7-Tiny architecture, respectively. Models 5–7 integrated two of the proposed improvements into the YOLOv7-Tiny baseline model, and Model 8 represented the YOLOv7-Tiny-PDE model.

Table 5. Comparison of metrics in ablation experiments.

Improved Model	Parameters	GFLOPs	P/%	R/%	mAP@0.5/%	F1	Speed/ms
YOLOv7-Tiny	6,007,596	13.1	96.70	94.10	94.10	0.954	19.8
YOLOv7-Tiny-PConv	4,728,108	10.7	96.10	94.30	94.60	0.952	19.1
YOLOv7-Tiny-DyHead	5,965,004	13.0	97.80	90.70	94.40	0.941	17.0
YOLOv7-Tiny-EIOU	6,007,596	13.1	97.10	93.00	94.30	0.950	15.6
YOLOv7-Tiny-PConv+DyHead	4,676,516	10.7	97.10	92.70	96.90	0.948	18.5
YOLOv7-Tiny-PConv+EIOU	4,728,108	10.7	96.60	95.80	96.20	0.962	17.3
YOLOv7-Tiny-EIOU+DyHead	5,965,004	13.0	97.50	94.70	97.70	0.961	16.6
YOLOv7-Tiny-PDE	4,676,516	10.7	97.20	96.60	97.90	0.969	17.5

According to Table 5, replacing the original backbone's ELAN with a Fasternet-based PConv increased the average precision by 0.5%. This improvement was due to PConv's focus on shallow features, enabling the better extraction of texture and edge information for small-scale targets. By enhancing local feature weighting, the model better emphasized visible regions (e.g., unobstructed apple stems or edges), which led to a 0.2% recall increase, indicating effective spatial information retention and the mitigation of feature loss from occlusion. Despite balancing lightweight design and accuracy, PConv's detection of extremely small targets (diameter < 50 pixels) remained limited. Additionally, the total parameters decreased by 21.3%, the GFLOPs decreased by 18.3%, and the detection speed improved by 3.5%. Since the original detection head in the neck network was replaced by DyHead, due to the correlation between precision and recall, the precision increased by 1.1%, while the

recall decreased by 3.6%. However, the mAP rose by 3.3%, and the detection speed rose by 14.1%. This improvement intensified the model's detection accuracy, as well as maintained the computational efficiency, and it provided better feature enhancement for multi-scale targets in complicated scenes. Finally, with the original CIoU loss function being replaced by the EIoU loss function, both the precision and accuracy improved a lot, with a significant 21.2% increase in the detection speed and a notably optimized convergence speed for target localization. The model parameters for all three combinations of the two improvement schemes were significantly reduced. In the PConv+DyHead scheme, the accuracy increased by 1.3%, the mAP improved by 2.9%, and the detection speed was enhanced by 4.7%. In the PConv+EIoU scheme, the mAP increased by 2.2%, and the detection speed improved by 9.5%, but the accuracy slightly decreased. In the EIoU+DyHead scheme, the accuracy, mAP, and detection speed all showed significant improvements.

After introducing the three improvement methods, the recall increased by 2.7%, the precision improved by 0.5%, the mAP rose by 4.0%, the F1 score rose by 1.6%, and the detection time decreased by 11.6%. The combination of these three modifications effectively leveraged their corresponding advantages, which resulted in an overall performance enhancement. This outcome confirms that the improved model, YOLOv7-Tiny-PDE, demonstrated efficiency and stability in handling complex scenes, particularly in the recognition and localization of occluded, dense, and multi-scale targets. The model shows advantages in being more lightweight, accurate, and easier to deploy.

3.4. Comparison Experiment of Detection Head Attention Mechanism

Compared with single-level attention modules, such as CBAM, DyHead dynamically adapts to the detection requirements of multi-scale targets in complex scenarios by jointly optimizing the scale, spatial, and task-aware features. According to Table 6, the F1 score of DyHead (0.972) significantly outperformed those of CBAM (0.948) and SE (0.932). Its multi-level attention mechanism was more effective at extracting local salient features of occluded apples (e.g., stems or edge textures).

Table 6. Comparison of detection attention mechanism metrics.

Module	Parameters (M)	GFLOPs	mAP@0.5/%	F1
CBAM	5.12	11.2	95.3	0.948
SE	4.89	10.8	94.1	0.932
DyHead	5.96	13.0	97.8	0.972

The experiments demonstrated that DyHead, while maintaining a lightweight design, significantly outperformed the traditional attention modules at dynamic feature aggregation. For example, under similar parameter counts, although DyHead's computational complexity was slightly higher than that of CBAM and SE, its mAP@0.5 improved by 2.6% and 3.9% compared with that of CBAM and SE, respectively, which validated its overall advantages in complex orchard scenarios.

3.5. Improved Network Model Combined with DeepSort for Counting Performance

The improved YOLOv7-Tiny and YOLOv7-Tiny-PDE models were combined with the DeepSort algorithm to track and count apple fruit. Figure 15 illustrates the counting results.

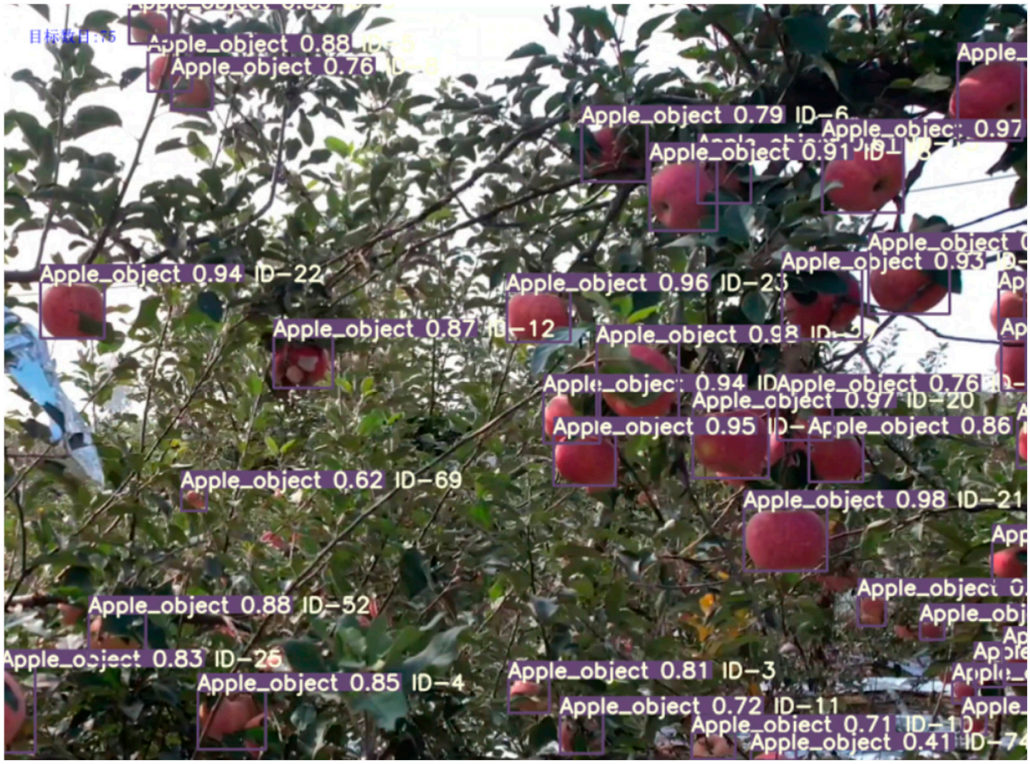

Figure 15. Fruit-counting results.

To validate the performance of the combined algorithm, the pre- and post-improvement model weights were integrated into the DeepSort algorithm and applied to video detection tasks for the evaluation of the tracking and counting effectiveness. In this study, the MOTA metric was used to assess the tracking performance on five selected apple fruit video segments as test inputs.

The *MAE* (mean absolute error) was used to test the difference in the average absolute error between the predicted and true values. By taking the absolute value, this disregards the direction of the error (positive or negative), ensuring that all differences are non-negative. A smaller MAE value indicates a higher prediction accuracy.

$$MAE = \frac{1}{m}\sum_{i=1}^{m}\left|\frac{x_{test}^{(i)} - y_{test}^{(i)}}{y_{test}^{(i)}}\right|, \tag{16}$$

In this equation, x_test(i) represents the ground truth count obtained through manual annotation in the video sequence, while y_test(i) denotes the predicted count generated by the multi-object tracking algorithm. The variable m indicates the total number of videos being evaluated, and i is the index of the current video. This metric directly assesses the overall performance of the model and serves as an effective measure of the detection and counting accuracy.

Table 7 presents the MOTA of the detection results after combining the pre- and post-improvement models with the DeepSort tracking algorithm, as well as the MAE between the improved overall algorithm and the manual counting results. Here, target number A and target number B represent the number of targets before and after the improvement, respectively, while MAE-A and MAE-B denote the average error values of the models before and after the improvement when combined with DeepSort.

Table 7. DeepSort tracking and counting metrics.

Video ID	Manual Count	Target Number A	Target Number B	MAE-A	MAE-B
1	316	355	362	0.123	0.145
2	221	237	251	0.072	0.136
3	144	157	164	0.090	0.139
4	151	163	166	0.079	0.099
5	101	109	113	0.079	0.118
Average	---	---		0.089	0.127

As shown in Table 8, for the selected video segments, the improved algorithm's MOTA increased by 2.2% compared with the baseline model. The number of identity switches (IDSWs) was significantly reduced, and the IDF1 score improved by 10.9%, demonstrating the enhanced target association consistency provided by the DyHead and EIoU loss function. However, tracking failures still occurred when the fruit was occluded by dense branches and leaves at a rate of over 90% and when there were sudden motion changes between consecutive frames. In such cases, the tracking algorithm may experience identity switches due to the loss of appearance features.

Table 8. Comparison of tracking performance metrics.

Model	MOTA/%	IDF1	IDSW
DeepSort	89.3	0.82	12.3
YOLOv7-Tiny-PDE+DeepSort	91.3	0.91	5.7

3.6. Algorithm Performance Evaluation in Field Scenarios

In practical orchard deployment, the algorithm's energy consumption directly affects the device's endurance and usability. The YOLOv7-Tiny-PDE algorithm was deployed on a drone platform (P230) during the field validation in this study, with the aim of assessing the improved model's energy efficiency. The drone was equipped with a powerful Graphics Processing Unit (GPU) and applied the NVIDIA Jetson AGX Xavier embedded platform to the processing of the algorithm [51]. The testing scenarios covered various lighting conditions (frontlit and backlit) and occlusion environments (mild occlusion and dense occlusion). The system was run for a continuous 10 min video stream detection task, recording both the average and peak power consumptions.

The test results indicate that YOLOv7-Tiny-PDE exhibited significant energy consumption advantages while maintaining a high detection accuracy. According to Table 9, the average power consumption of the improved model was 8.3 W, 17.8% lower than that of the baseline YOLOv7-Tiny (10.1 W). In dense occlusion scenarios, the collaborative optimization of the DyHead and EIoU loss function improved the model inference efficiency, where it reduced the peak power consumption from 12.5 W to 10.9 W, a decrease of 12.8%. Additionally, a comparison of different hardware load states (GPU utilization) shows that the PConv module effectively reduced the memory access frequency, which led to a decrease in the GPU utilization from 78% to 65%, and thus, further reduced the energy consumption fluctuations.

Table 9. Energy consumption comparison in field scenarios.

Model	Average Power Consumption/W	Peak Power Consumption/W	GPU Utilization/%
YOLOv7-Tiny	10.1	12.5	78
YOLOv7-Tiny-PDE	8.3	10.9	65

This result confirms that the lightweight design of the improved model significantly reduced the computational redundancy while also lowering the energy consumption in the field deployments. It provides feasible support for the long-term operation of resource-constrained devices, such as drones and mobile robots.

4. Conclusions

This study integrated PDE into the original YOLOv7-Tiny model to obtain the improved YOLOv7-Tiny-PDE algorithm, combined with the DeepSort algorithm. "PDE" stands for partial convolution (PConv), dynamic detection head (DyHead), and loss function (EIOU). The algorithm uses regular convolutions to replace an efficient layer aggregation network in the backbone, substitutes the CIoU loss function, and replaces the original detection head in the neck network to enable apple detection and counting.

In complex scenarios with occlusion, object detection networks often struggle with parameter redundancy and a high computational load. PConv reduces parameters and unnecessary computations while maintaining detection accuracy. In unstructured apple orchards, particularly in complex scenes and multi-scale target situations, feature extraction is often insufficient, resulting in missed or incorrect detections. Introducing the DyHead effectively suppressed the background interference and captured features more comprehensively, which improved the detection accuracy for various scales and occlusions. The EIOU loss function accelerated the model convergence by minimizing the difference between the predicted and practical truth boxes in terms of the width and height, and thereby enhanced the optimization performance.

The improved model excelled at apple recognition and localization, where it enabled precise detection and tracking in complex environments while it enhanced the model's lightweightness, accuracy, and counting performance. The experimental results show that the improved YOLOv7-Tiny-PDE algorithm performed better than the custom apple dataset: the parameters were reduced by 22.2%; the GFLOPs were reduced by 18.3%; the P and R values increased by 0.5% and 2.7%, respectively; the mAP@0.5 and F1 scores were increased by 4% and 1.7%; and MOTA was increased by 2%. These improvements will assist orchard managers in achieving more efficient orchard management, reducing labor costs, and accurately and intelligently estimating the fruit yield in complex natural environments.

Based on the existing research status in the fields of object detection and fruit yield estimation, future research directions can be envisioned in the following aspects:

(1) Small fruit targets and growth cycle patterns: The performance of current algorithm in detecting small targets shall be further improved. Further optimization is required, as the existing dataset lacks sufficient differentiation in color and maturity for apples at different growth stages, particularly for light-colored, immature, or yellow-green apples, which are underrepresented in the detection samples. Variations in apple varieties at the same maturity stage may also lead to estimation biases, indicating significant potential for improving the accuracy of actual fruit yield estimation.

(2) Generalization and real-time deployment: The current data collection is limited in time and location, making it difficult to fully represent diverse orchard environments and potentially affecting the model's generalization ability. Future research will explore model compression and channel pruning techniques to accurately trim redundant channels without compromising the detection accuracy, thereby reducing the model size and improving the efficiency. Additionally, the model will be adapted to hardware devices used in practical agricultural environments in order to ensure efficient operation on such devices.

(3) Stability between video frames: Future work will explore the integration of time-series models, such as long short-term memory (LSTM) networks or gated recurrent units (GRUs), to effectively integrate temporal information between video frames. By learning

feature variations in the time dimension, this approach improves the model's prediction accuracy of fruit movement trajectories, thereby enhancing the tracking stability.

Future work will explore advanced improvement methods and evaluation criteria, along with multi-object tracking models, to balance the lightweight design, detection accuracy, and speed. Techniques such as model compression and channel pruning will enhance the efficiency. Data collection will expand to cover diverse times, regions, and growth stages. The focus will also be on deploying lightweight networks on embedded and resource-constrained edge devices, minimizing the accuracy loss and enhancing the practical applicability.

Author Contributions: Conceptualization, D.C., W.L., R.T., Y.L., J.Z., X.L. and L.Y.; methodology, D.C. and W.L.; software, D.C. and W.L.; validation, R.T., W.L. and Y.L.; formal analysis, Y.L., X.L. and L.Y.; investigation, Y.L., X.L. and L.Y.; resources, D.C., W.L. and J.Z.; data curation, D.C., W.L. and J.Z.; writing—original draft preparation, D.C. and W.L.; writing—review and editing, D.C., W.L. and Y.L.; visualization, Y.L. and X.L.; project administration, W.L. and Y.L.; funding acquisition, Y.L., W.L. and L.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Central Government Guides Local Funds for Science and Technology Development (No. 246Z7401G) and the Ministry of Education Supply-Demand Matching Employment Education Program: Teaching Innovation and Employment in GIS Majors Integrated with AI Technology (No. 2024120557855).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data used in this study are publicly available and can be freely accessed at https://pan.baidu.com/s/1Gx8HmEfmXFSfBw88FTG5TQ?pwd=y8vmon 20 January 2025.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Fan, P.; Lang, G.; Guo, P.; Liu, Z.; Yang, F.; Yan, B.; Lei, X. Multi-Feature Patch-Based Segmentation Technique in the Gray-Centered RGB Color Space for Improved Apple Target Recognition. *Agriculture* **2021**, *11*, 273. [CrossRef]
2. Fan, P.; Lang, G.; Yan, B.; Lei, X.; Guo, P.; Liu, Z.; Yang, F. A Method of Segmenting Apples Based on Gray-Centered RGB Color Space. *Remote Sens.* **2021**, *13*, 1211. [CrossRef]
3. Fan, P.; Yan, B.; Wang, M.; Lei, X.; Liu, Z.; Yang, F. Three-finger grasp planning and experimental analysis of picking patterns for robotic apple harvesting. *Comput. Electron. Agric.* **2021**, *188*, 106353. [CrossRef]
4. Fu, L.; Gao, F.; Wu, J.; Li, R.; Karkee, M.; Zhang, Q. Application of consumer RGB-D cameras for fruit detection and localization in field: A critical review. *Comput. Electron. Agric.* **2020**, *177*, 105687. [CrossRef]
5. Wang, N.; Joost, W.; Zhang, F. Towards sustainable intensification of apple production in China-Yield gaps and nutrient use efficiency in apple farming systems. *J. Integr. Agric.* **2016**, *15*, 716–725. [CrossRef]
6. Rakun, J.; Stajnko, D.; Zazula, D. Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry. *Comput. Electron. Agric.* **2011**, *76*, 80–88. [CrossRef]
7. Gongal, A.; Amatya, S.; Karkee, M.; Zhang, Q.; Lewis, K. Sensors and systems for fruit detection and localization: A review. *Comput. Electron. Agric.* **2015**, *116*, 8–19. [CrossRef]
8. Wang, C.; Tang, Y.; Zou, X.; SiTu, W.; Feng, W. A robust fruit image segmentation algorithm against varying illumination for vision system of fruit harvesting robot. *Optik* **2017**, *131*, 626–631. [CrossRef]
9. Chaivivatrakul, S.; Dailey, M.N. Texture-based fruit detection. *Precis. Agric.* **2014**, *15*, 662–683. [CrossRef]
10. Bulanon, D.; Kataoka, T. Fruit detection system and an end effector for robotic harvesting of Fuji apples. *Agric. Eng. Int. CIGR E-J.* **2010**, *12*, 203–210.
11. Hed, B.; Centinari, M. Hand and mechanical fruit-zone leaf removal at prebloom and fruit-set was more effective in reducing crop yield than reducing bunch rot in 'riesling' grapevines. *Horttechnology* **2018**, *28*, 296–303. [CrossRef]
12. Payne, A.B.; Walsh, K.B.; Subedi, P.; Jarvis, D. Estimation of mango crop yield using image analysis– segmentation method. *Comput. Electron. Agric.* **2013**, *91*, 57–64. [CrossRef]
13. Sengupta, S.; Lee, W.S. Identification and determination of the number of immature green citrus fruit in a canopy under different ambient light conditions. *Biosyst. Eng.* **2014**, *117*, 51–61. [CrossRef]

14. Kurtulmuş, F.; Kavdir, I. Detecting corn tassels using computer vision and support vector machines. *Expert. Syst. Appl.* **2014**, *41*, 7390–7397. [CrossRef]
15. Mai, C.; Zheng, L.; Xiao, C.; Li, M. Comparison of apple recognition methods under natural light. *J. China Agric. Univ.* **2016**, *21*, 43–50.
16. Lin, G.; Tang, Y.; Zou, X.; Xiong, J.; Fang, Y. Color-, depth-, and shape-based 3D fruit detection. *Precis. Agric.* **2020**, *21*, 1–17. [CrossRef]
17. Lv, J.; Zhao, D.; Ji, W. Fast tracing recognition method of target fruit for apple harvesting robot. *Trans. Chin. Soc. Agric. Mach.* **2014**, *45*, 65–72.
18. Luo, L.; Tang, Y.; Lu, Q.; Chen, X.; Zhang, P.; Zou, X. A vision methodology for harvesting robot to detect cuting points on peduncles of double overlapping grape clusters in a vineyard. *Comput. Ind.* **2018**, *99*, 130–139. [CrossRef]
19. Si, Y.; Qiao, J.; Liu, G.; Gao, R.; He, B. Recognition and location of fruits for appleharvesting robot. *Trans. Chin. Soc. Agric. Mach.* **2010**, *41*, 148–153.
20. Moallem, P.; Serajoddin, A.; Pourghassem, H. Computer vision-based apple grading for golden delicious apples based on surface features. *Inf. Process. Agric.* **2017**, *4*, 33–40. [CrossRef]
21. Xiao, Y.; Tian, Z.; Yu, J.; Zhang, Y.; Liu, S.; Du, S.; Lan, X. A review of object detection based on deep learning. *Multimed. Tools Appl.* **2020**, *79*, 23729–23791. [CrossRef]
22. Häni, N.; Roy, P.; Isler, V. A comparative study of fruit detection and counting methods for yield mapping in apple orchards. *J. Field Robot.* **2020**, *37*, 263–282. [CrossRef]
23. Xiong, J.; Liu, Z.; Tang, L. Research on visual detection technology of green citrus in natural environment. *Trans. Chin. Soc. Agric. Mach.* **2018**, *49*, 45–52.
24. Peng, H.X.; Huang, B.; Shao, Y.Y. General improved SSD model for picking object recogni-tion of multiple fruits in natural environment. *Trans. Chin. Soc. Agric. Eng.(Trans. CSAE)* **2018**, *34*, 155–162.
25. Sun, J.; He, X.; Ge, X.; Wu, X.; Shen, J.; Song, Y. Detection of key organs in tomato based on deep migration learning in a complex background. *Agriculture* **2018**, *8*, 8196. [CrossRef]
26. Chu, X.; Zheng, A.; Zhang, X.; Sun, J. Detection in crowded scenes: One proposal, multiple predictions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 12214–12223.
27. Chen, J.; Liu, H.; Zhang, Y.; Zhang, D.; Ouyang, H.; Chen, X. A Multiscale Lightweight and Efficient Model Based on YOLOv7: Applied to Citrus Orchard. *Plants* **2022**, *11*, 3260. [CrossRef]
28. Sekharamantry, P.K.; Melgani, F.; Malacarne, J. Deep Learning-Based Apple Detection with Attention Module and Improved Loss Function in YOLO. *Remote Sens.* **2023**, *15*, 1516. [CrossRef]
29. Li, X.; Pan, J.; Xie, F.; Zeng, J.; Li, Q.; Huang, X.; Liu, D.; Wang, X. Fast and accurate green pep- per detection in complex backgrounds via an improved YOLOv4-tiny model. *Comput. Electron. Agric.* **2021**, *191*, 106503. [CrossRef]
30. Lai, Y.; Ma, R.; Chen, Y.; Wan, T.; Jiao, R.; He, H. A Pineapple Target Detection Method in a Field Environment Based on Improved YOLOv7. *Appl. Sci.* **2023**, *13*, 2691. [CrossRef]
31. Tian, Y.; Yang, G.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *Comput. Electron. Agric.* **2019**, *157*, 417–426. [CrossRef]
32. Zhou, Y.; Tang, Y.; Zou, X.; Wu, M.; Tang, W.; Meng, F.; Zhang, Y.; Kang, H. Adaptive Active Positioning of Camellia oleifera Fruit Picking Points: Classical ImageProcessing and YOLOv7 Fusion Algorithm. *Appl. Sci.* **2022**, *12*, 12959. [CrossRef]
33. Ji, W.; Pan, Y.; Xu, B.; Wang, J. A Real-Time Apple Targets Detection Method for Picking Robot Based on ShufflenetV2-YOLOX. *Agriculture* **2022**, *12*, 856. [CrossRef]
34. Wu, D.; Jiang, S.; Zhao, E.; Liu, Y.; Zhu, H.; Wang, W.; Wang, R. Detection of Camellia oleifera Fruit in Complex Scenes by Using YOLOv7 and Data Augmentation. *Appl. Sci.* **2022**, *12*, 11318. [CrossRef]
35. Liu, X.; Li, G.; Chen, W.; Liu, B.; Chen, M.; Lu, S. Detection of dense Citrus fruits by combining coordinated attention and cross-scale connection with weighted feature fusion. *Appl. Sci.* **2022**, *12*, 6600. [CrossRef]
36. Wang, F.; Jiang, J.; Chen, Y.; Sun, Z.; Tang, Y.; Lai, Q.; Zhu, H. Rapid detection of Yunnan Xiaomila based on lightweight YOLOv7 algorithm. *Front. Plant Sci.* **2023**, *14*, 1200144. [CrossRef] [PubMed]
37. Zhao, H.; Qiao, Y.; Wang, H.; Yue, Y. Apple fruit recognition in complex orchard environment based on improved YOLOv3. Trans. *Chin. Soc. Agric. Eng.* **2021**, *37*, 127–135.
38. Yang, F.; Lei, X.; Liu, Z.; Fan, P.; Yan, B. Fast Recognition Method for Multiple Apple Targets in Dense Scenes Based on CenterNet. *Trans. Chin. Soc. Agric. Mach.* **2022**, *53*, 265–273.
39. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
40. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 583–596. [CrossRef] [PubMed]

41. Liu, X. Robust Fruit Counting: Combining Deep Learning, Tracking, and Structure from Motion. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1045–1052. [CrossRef]
42. Stein, M.; Bargoti, S.; Underwood, J. Image based mango fruit detection, localisation and yield estimation using multiple view geometry. *Sensors* **2016**, *16*, 1915. [CrossRef]
43. Bargoti, S.; Underwood, J. Deep fruit detection in orchards. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 3626–3633.
44. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.
45. Halstead, M.; McCool, C.; Denman, S.; Perez, T.; Fookes, C. Fruit Quantity and Ripeness Estimation Using a Robotic Vision System. *IEEE Robot. Autom. Lett.* **2018**, *3*, 2995–3002. [CrossRef]
46. Bhattarai, U. A weakly-supervised approach for flower/fruit counting in apple orchards. *Comput. Ind.* **2022**, *138*, 103635. [CrossRef]
47. Häni, N. Apple Counting using Convolutional Neural Networks. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2559–2565. [CrossRef]
48. Wang, C.-Y.; Liao, H.-Y.M.; Yeh, I.-H. Designing network design strategies through gradient path analysis. *arXiv* **2022**, arXiv:2211.04800.
49. Wu, C.; Ye, M.; Zhang, J.; Ma, Y. YOLO-LWNet: A Lightweight Road Damage Object Detection Network for Mobile Terminal Devices. *Sensors* **2023**, *23*, 3268. [CrossRef] [PubMed]
50. Dai, X.; Chen, Y.; Xiao, B.; Chen, D.; Liu, M.; Yuan, L.; Zhang, L. Dynamic head: Unifying object detection heads with attentions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 7373–7382.
51. Luo, W.; Zhao, Y.; Shao, Q.; Li, X.; Wang, D.; Zhang, T.; Liu, F.; Duan, L.; He, Y.; Wang, Y.; et al. Procapra Przewalskii Tracking Autonomous Unmanned Aerial Vehicle Based on Improved Long and Short-Term Memory Kalman Filters. *Sensors* **2023**, *23*, 3948. [CrossRef] [PubMed]