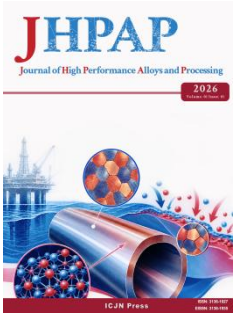
Journal Website: https://www.icjn.net/jhpap ISSN: 3136-1927 | EISSN: 3136-1935

Journal of High Performance Alloys and Processing (JHPAP)

ICJN Press Volume. 01 | Issue. 01 | Oct. 2026 DOI: 10.66729/jhpap202604013



ARTICLE

FCD-YOLO: An Improved Object Detection Algorithm for Wind Turbine Blade Surface Damage Detection

Rui Zhao[1,2], Chenyu Wang[1,*], Zixiang Yue[1,2]

1 College of Civil Engineering and Architecture, Xinjiang University, Urumqi Xinjiang 830017, China.
2 Xinjiang Key Laboratory of Building Structure and Earthquake Resistance, Urumqi Xinjiang 830017, China.

*Corresponding Author: Chenyu Wang, E-mail: 2165813304@qq.com

Abstract

Among routine wind turbine maintenance activities, the timely and accurate detection of blade surface damage is crucial for maintaining structural integrity and ensuring operational safety. In practical on-site wind farm inspection workflows, aerial images captured via unmanned aerial vehicles (UAVs) frequently suffer from miniature, geometrically irregular defect targets, highly cluttered natural backgrounds, fluctuating lighting intensity, heavy shadow occlusion, and persistent surface dirt contamination, which creates an intractable performance conflict that conventional detection frameworks struggle to balance high localization precision and low inference latency simultaneously. To address these challenges, this study proposes an improved detection model named FCD-YOLO. Within this architecture, lightweight FasterNet Blocks are fully embedded into both the backbone feature extraction pipeline and the neck multi-scale fusion branch to reduce computational redundancy and improve inference efficiency. In parallel, a Channel Prior Convolutional Attention (CPCA) module is embedded to selectively enhance damage-sensitive features by jointly considering channel-wise and spatial information and suppressing irrelevant background feature activations. Moreover, an auxiliary detection branch — auxiliary detection head (DetectAux) is incorporated to introduce supplementary supervisory cues during training, which facilitates more robust learning of features across different scales and alleviates missed detection risks for tiny, low-contrast blade defects. Comprehensive experimental validation indicates that the proposed model achieves mAP@0.5 and mAP@0.5:0.95 scores of 80.98% and 57.77%, respectively, surpassing the corresponding results of the YOLO11s baseline (79.22% and 56.08%). Side-by-side benchmark tests against multiple state-of-the-art mainstream detectors further validate the comprehensive competitive superiority of the proposed FCD-YOLO framework. Overall, this work provides a promising approach for workstation-based offline or near-real-time wind turbine blade damage inspection with remarkable lightweight characteristics and robust anti-interference capability for complex field environments.

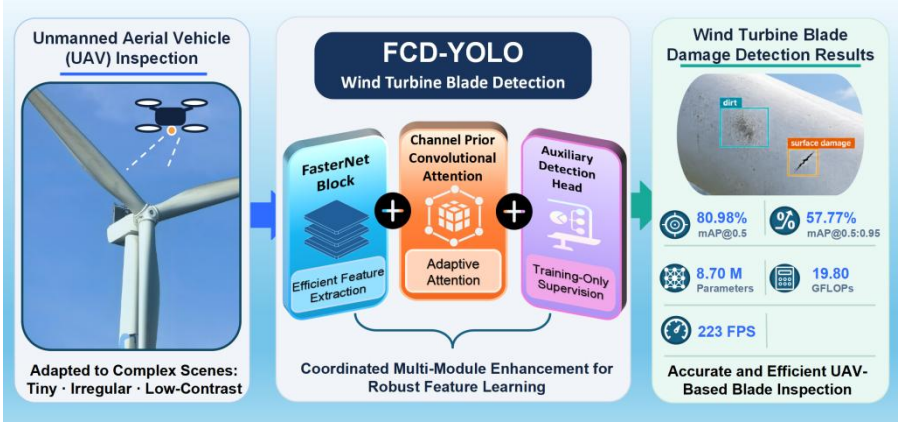
Keywords

Deep learning; YOLO11; Wind turbine; Damage detection

Citation: Rui Zhao, Chenyu Wang and Zixiang Yue. FCD-YOLO: An Improved Object Detection Algorithm for Wind Turbine Blade Surface Damage Detection. *Journal of High Performance Alloys and Processing*. 2026, 01 (1): 1-13. doi: 10.66729/jhpap202604013.

Copyright:© 2026 The Author(s). Published by ICJN Press.
This is an open access article under the https://creativecommons.org/licenses/by/4.0/
Received 13 April 2026; Revised 10 July 2026; Accepted 19 August 2026; Available online 30 October 2026.

Graphical Abstract



Highlights

- FCD-YOLO integrates FasterNet, Channel Prior Convolutional Attention (CPCA), and auxiliary detection head (DetectAux) for blade inspection and it maintains high detection accuracy even under several challenging conditions.
- Compared with YOLO11s, FCD-YOLO improves mAP@0.5 and mAP@0.5:0.95 by 1.76 and 1.69 percentage points, reaching 80.98% and 57.77%, respectively.
- It reduces parameters from 9.41 to 8.70 M and computational cost from 21.30 to 19.80 GFLOPs, while increasing inference speed from 208 to 223 FPS.
- FCD-YOLO achieves the best overall detection accuracy among general-purpose mainstream detectors on the public Wind Turbine Damage Dataset.

1. Introduction

With the continuous growth of global energy demand and the accelerated transformation of the global energy system toward low-carbon development, wind energy has become a core component of the low-carbon sustainable power generation system[1]. According to the Global Wind Report 2025, China is projected to add approximately 460 GW of newly installed onshore wind power capacity between 2025 and 2030, which demonstrates both the rapid expansion of the wind power industry and its growing strategic importance in long-term national energy system planning[2]. Along with the continuous large-scale deployment of wind power, this expansion places increasing demands on the power generation efficiency and operational reliability of wind turbine blades under long-duration cyclic service conditions. As the core component for wind energy capture, wind turbine blades are also one of the most failure-prone components in wind power systems, and their manufacturing cost accounts for approximately 15%–20% of the total cost of wind turbines[3]. The structural integrity and service condition of blades directly determine the overall operational performance and reliability of wind power systems. However, under long-term exposure to harsh outdoor environmental conditions and repeated cyclic mechanical loading, blade materials will inevitably undergo gradual degradation, further inducing surface and internal structural damage such as cracks and delamination[4,5], alongside surface erosion, scratch marks, and composite layer peeling defects commonly observed in actual operation.

Traditional non-destructive testing methods, including acoustic emission[6] and infrared thermography[7], exhibit certain capabilities in defect detection but remain subject to significant limitations. Acoustic emission inspection requires dense sensor arrangement on curved blade surfaces and is extremely vulnerable to random mechanical noise from wind field equipment, severely undermining detection stability. Infrared thermography enables non-contact inspection of delamination; however, its performance is strongly affected by ambient conditions and limited spatial resolution, making it less suitable for detecting small surface cracks distributed on thin blade tip regions.

Recent studies have highlighted the effectiveness of deep learning techniques for wind turbine blade damage detection in unmanned aerial vehicle (UAV)-based inspection scenarios. Compared with traditional image-processing and hand-crafted feature methods, deep neural networks can automatically extract discriminative features from complex blade images, thereby improving defect localization and recognition without manual feature design and parameter tuning. For instance, Tong *et al.* proposed WTBD-YOLOv8 to address challenges such as complex backgrounds, high image resolution, long inference time, and limited detection accuracy[8]. Meanwhile, Transformer-based methods have also been introduced into this field. Hu *et al.* developed an RDSS-YOLO framework with a Swin Transformer-based global feature fusion module for real-time blade surface damage assessment[9]. In addition, approaches based on large vision models, such as DINOv2-assisted feature extraction, have been explored to enhance recognition robustness under complex surface textures[10]. Given the scarcity and diversity of blade defects, few-shot learning has also gained increasing attention. Zhou *et al.* introduced a RAG-enhanced vision-language framework for few-shot blade damage diagnosis[11], while Gohar *et al.* investigated few-shot detection under class scarcity and imbalance[12].

Despite recent advances, practical UAV-based wind turbine blade inspection still faces challenges related to small defect regions, complex backgrounds, and the trade-off between detection accuracy and inference efficiency, while most existing improved You Only Look Once (YOLO) variants only optimize single performance indicators rather than balancing precision, speed, and small-target robustness

simultaneously. To address these issues, this study develops FCD-YOLO based on the YOLO11 framework by integrating three high-performance modules reported in recent computer vision studies. The effectiveness and applicability of the proposed model are further evaluated through comparative experiments with mainstream object detectors, including Faster Region-Based Convolutional Neural Network (Faster R-CNN)[13], Real-Time Detection Transformer (RT-DETR)[14], and other representative methods. The major technical contributions of this study are summarized as follows:

(1) FasterNet Block is integrated into the backbone and neck to improve inference efficiency by reducing floating-point operations (FLOPS) and memory access, while maintaining a favorable balance between computational cost and detection accuracy via partial convolution that only processes partial feature channels to eliminate redundant calculations.

(2) The network further incorporates a Channel Prior Convolutional Attention (CPCA) module to enhance sensitivity to informative defect features by dynamically adjusting attention responses along the channel and spatial dimensions, thereby improving the discrimination of subtle damage patterns through channel-prior guided spatial weighting instead of shared spatial masks for all feature channels.

(3) An auxiliary detection head (DetectAux) is incorporated to provide additional supervision during training, thereby improving multi-scale and multi-level feature extraction and enhancing model robustness under complex inspection conditions, particularly for small and irregular defects; auxiliary branches are discarded during inference to avoid extra computational overhead at the deployment stage.

Different from prior improved YOLO schemes customized for wind turbine defect identification, this research innovatively combines lightweight convolution, channel-spatial adaptive attention, and training-only auxiliary supervision to jointly resolve three core practical bottlenecks of on-site blade inspection.

2. Theory and Methods

2.1 Basic YOLO11 Model Architecture

Object detection models are generally categorized into two-stage and one-stage approaches according to their detection pipelines. Two-stage detectors first generate candidate regions and subsequently perform object classification and bounding-box refinement. Although these methods often achieve high detection accuracy, their greater computational complexity and lower inference speed may limit their suitability for near-real-time analysis. In contrast, one-stage detectors directly predict object categories and locations within a unified framework. As a representative one-stage detection framework, the YOLO family[15–18] formulates object detection as an end-to-end prediction task in which class probabilities and bounding-box coordinates are estimated simultaneously through a single forward pass. This streamlined design provides high inference efficiency while maintaining competitive detection accuracy, making YOLO-based models particularly suitable for real-time visual inspection applications.

YOLO11 is a recent member of the YOLO family that retains the conventional backbone-neck-head architecture while introducing several structural refinements to improve feature extraction, multi-scale feature fusion, and computational efficiency. The overall architecture of the baseline YOLO11 model is shown in **Figure 1**. In the backbone, a sequence of convolutional (Conv) layers progressively downsamples the input and extracts hierarchical features at multiple resolutions. The Convolutional Block 3 Kernel 2 (C3k2) module replaces the C2f structure used in earlier YOLO versions and adopts a flexible cross-stage feature-processing design that promotes feature reuse and improves representational efficiency without incurring substantial computational overhead. At the end of the backbone, the Spatial Pyramid Pooling-Fast (SPPF) module aggregates contextual information at multiple receptive-field scales through repeated max-pooling operations. The Cross-Stage Partial Spatial Attention (C2PSA) module is subsequently applied to emphasize informative spatial regions and suppress irrelevant background responses, thereby improving feature discrimination. In the neck, upsampling and feature concatenation operations combine shallow spatial details with deep semantic information, enabling effective multi-scale feature fusion. Finally, the detection head processes feature maps at three different scales to jointly predict object categories and bounding-box locations, allowing the model to detect targets of varying sizes from micro-defects to large-area blade surface failures.

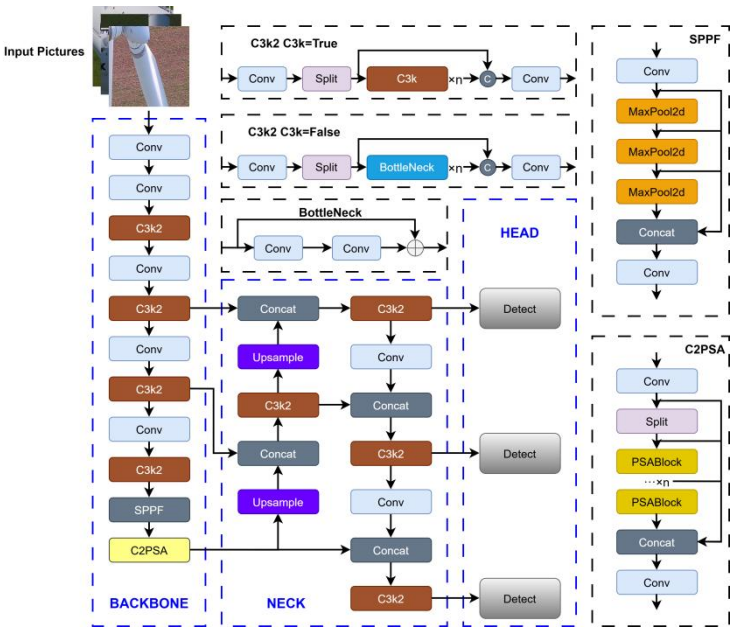


Figure 1. Architecture of the baseline YOLO11 model. **NOTE:** Conv: Convolution; SPPF: Spatial Pyramid Pooling-Fast; C2PSA: Cross-Stage Partial Spatial Attention; C3k2: Convolutional Block 3 Kernel 2. Copyright: Created by the authors using draw.io Diagrams, version 30.2.6.0.

2.2 Improved FCD-YOLO Model Architecture

Building on the YOLO11 architecture described above, this study develops an enhanced framework for wind turbine blade surface damage detection, as illustrated in the **Figure 2**. Specifically, the original Bottleneck units in the C3k2 module are replaced with FasterNet Blocks to reduce computational overhead and improve inference efficiency, achieving substantial reductions in parameter count and FLOPs without sacrificing feature extraction capability. In addition, CPCA is incorporated to enhance feature representation by adaptively emphasizing informative responses across both channel and spatial dimensions, thereby improving the detection of subtle and irregular surface defects. Furthermore, the DetectAux head is introduced to provide additional supervision during training, strengthen multi-scale feature learning, and improve model generalization under complex inspection conditions. Through the coordinated integration of these components, the proposed framework achieves a more favorable balance between detection accuracy and computational efficiency for wind turbine blade surface damage inspection and outperforms the original baseline across all core evaluation metrics.

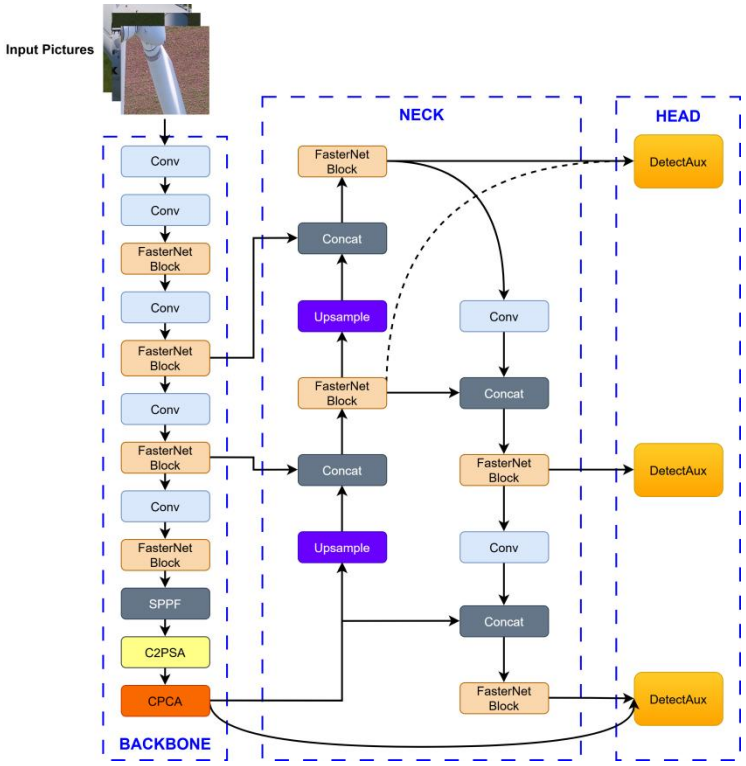


Figure 2. Architecture of the proposed FCD-YOLO model. **NOTE:** Conv: Convolution; SPPF: Spatial Pyramid Pooling-Fast; C2PSA: Cross-Stage Partial Spatial Attention; CPCA: Channel Prior Convolutional Attention; DetectAux: Auxiliary detection head. Copyright: Created by the authors using draw.io Diagrams, version 30.2.6.0.

To provide a clearer understanding of the specific function of each layer in the FCD-YOLO model, as well as the inter-layer connections and input-output relationships, a detailed layer-by-layer description is provided in **Table 1** for an input size of 640 pixels × 640 pixels. In the "From" column, "-1" indicates that the input is taken from the preceding layer, whereas brackets denote the joint input of feature maps from multiple layers for feature fusion. Since the s-scale configuration of YOLO11 is adopted, the channel width and number of repeated blocks are adjusted using a scaling factor of 0.50. Building on YOLO11, FCD-YOLO replaces the original C3k2 modules in the backbone and neck with FasterNet Blocks to enable lightweight and efficient feature extraction. In addition, CPCA is inserted after C2PSA to strengthen channel-spatial feature representation, while the original detection head is replaced with DetectAux, which comprises three main detection branches and three auxiliary branches for training-stage supervision, with auxiliary branches connected to intermediate multi-scale feature maps to deliver layered gradient constraints.

Table 1. Detailed architecture of the proposed FCD-YOLO network with 640 pixels × 640 pixels input

Layer	From	Module	Output Size (Pixels)	Output Channels	Description
0	-1	Conv	320 × 320	32	Initial downsampling, P1/2
1	-1	Conv	160 × 160	64	Downsampling, P2/4
2	**-1**	**FasterNet Block**	**160 × 160**	**128**	**FasterNet-based feature extraction, replaces the original Bottleneck-based C3k2 unit**
3	-1	Conv	80 × 80	128	Downsampling, P3/8
4	**-1**	**FasterNet Block**	**80 × 80**	**256**	**Backbone feature extraction for P3, responsible for small defect feature encoding**

Table 1. Detailed architecture of the proposed FCD-YOLO network with 640 pixels × 640 pixels (continued)

Layer	From	Module	Output Size (Pixels)	Output Channels	Description
5	-1	Conv	40 × 40	256	Downsampling, P4/16
6	**-1**	**FasterNet Block**	**40 × 40**	**256**	**Backbone feature extraction for P4, extracts medium-size damage semantic features**
7	-1	Conv	20 × 20	512	Downsampling, P5/32
8	**-1**	**FasterNet Block**	**20 × 20**	**512**	**Backbone feature extraction for P5, captures global large-area defect information**
9	-1	SPPF	20 × 20	512	Spatial pyramid pooling
10	-1	C2PSA	20 × 20	512	attention-enhanced feature aggregation
11	**-1**	**CPCA**	**20 × 20**	**512**	**Inserted after C2PSA to enhance channel-spatial representation**
12	-1	Upsample	40 × 40	512	Upsampling P5 features
13	[-1, 6]	Concat	40 × 40	768	Concatenation of upsampled P5 and backbone P4
14	**-1**	**FasterNet Block**	**40 × 40**	**256**	**Neck feature fusion**
15	-1	Upsample	80 × 80	256	Upsampling fused P4 features
16	[-1, 4]	Concat	80 × 80	512	Concatenation of upsampled P4 and backbone P3
17	**-1**	**FasterNet Block**	**80 × 80**	**128**	**Small-object detection feature map, P3/8**
18	-1	Conv	40 × 40	128	Downsampling from P3 to P4
19	[-1, 14]	Concat	40 × 40	384	Concatenation with neck P4 features
20	**-1**	**FasterNet Block**	**40 × 40**	**256**	**Medium-object detection feature map, P4/16**
21	-1	Conv	20 × 20	256	Downsampling from P4 to P5
22	[-1, 11]	Concat	20 × 20	768	Concatenation with CPCA-enhanced backbone P5 features
23	**-1**	**FasterNet Block**	**20 × 20**	**512**	**Large-object detection feature map, P5/32**
24	**[17, 20, 23, 17, 14, 11]**	**DetectAux**	**80 × 80, 40 × 40, 20 × 20**	**nc**	**Three main detection branches and three auxiliary detection branches**

NOTE: Conv: convolution; SPPF: Spatial Pyramid Pooling-Fast; C2PSA: Cross-Stage Partial Spatial Attention; CPCA: Channel Prior Convolutional Attention; DetectAux: Auxiliary detection head; nc: number of classes. Bold rows indicate layers containing modules newly introduced or modified relative to the original YOLO11 architecture.

2.2.1 FasterNet and PConv

The YOLO11 backbone adopts a hierarchical architecture composed of convolutional layers, residual connections, and bottleneck-based modules. Multi-stage feature extraction is performed primarily through C3k2 blocks, in which successive convolutional operations progressively capture spatial details and high-level semantic information. However, conventional convolutional operations are applied across all input channels, leading to redundant computation and substantial memory-access overhead. These limitations reduce inference efficiency and hinder deployment on resource-constrained devices such as embedded edge computing boards carried by unmanned aerial vehicles (UAVs).

To alleviate these limitations, FasterNet Blocks[19] are incorporated into the backbone and neck of YOLO11 to replace the original C3k2-based feature extraction blocks. The core FasterNet Block consists of a 3 × 3 partial convolution (PConv) followed by two 1 × 1 pointwise convolutions. Unlike standard convolution, which performs spatial filtering across all input channels, PConv applies convolution only to a selected subset of channels, while the remaining channels are directly retained and propagated. This partial channel-processing mechanism avoids repetitive computation on redundant feature channels and significantly reduces both floating-point operations and memory access, delivering around 26% FLOPs reduction compared with original C3k2 blocks according to ablation results. Meanwhile, the unprocessed channels preserve the original contextual information, preventing excessive information loss and maintaining stable feature transmission. The subsequent two 1 × 1 convolutions first expand and then restore the channel dimensions, enabling nonlinear transformation and sufficient cross-channel interaction. This process effectively integrates the spatial features extracted by PConv with the directly propagated features, thereby compensating for the restricted cross-channel interaction inherent in partial convolution. In addition, normalization, activation, and residual connections are incorporated to enhance nonlinear representation, facilitate gradient propagation, and improve training stability. Consequently, the FasterNet Block enables efficient spatial feature extraction and channel-wise feature fusion while maintaining low computational overhead.

Overall, the integration of the FasterNet Block not only reduces the computational redundancy of the original YOLO11 backbone but also improves the efficiency and discriminability of feature representation. This design provides a lightweight yet reliable foundation for accurate and real-time wind turbine blade surface damage detection. The architecture of the FasterNet Block is illustrated in **Figure 3**.

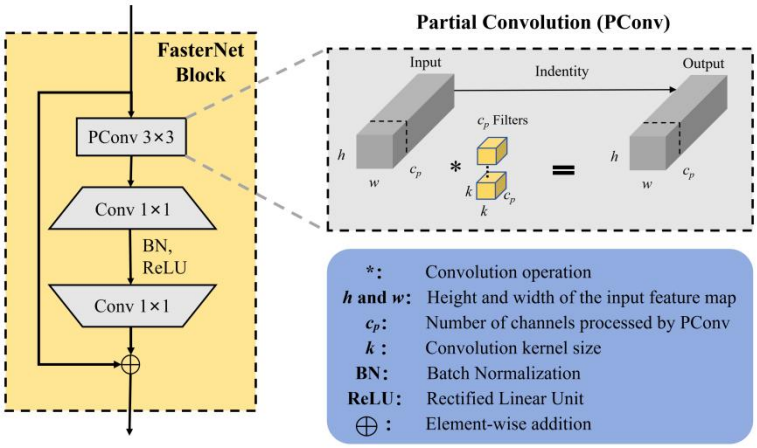


Figure 3. Architecture of the FasterNet block and PConv. **NOTE:** Conv: Convolution; PConv: Partial convolution. Copyright: Redrawn and adapted from Chen *et al*.[19] using WPS Office Presentation, version 12.1.0.26895.

2.2.2 Channel Prior Convolutional Attention (CPCA)

Most conventional attention mechanisms model channel and spatial dependencies separately or sequentially, but their spatial attention maps are generally generated by compressing channel information. As a result, all channels share the same spatial attention distribution, making it difficult to adapt spatial responses to channel-specific characteristics. Although the Convolutional Block Attention Module (CBAM)[20] combines channel and spatial attention, it still applies a shared spatial attention map to all output channels, whereas the Squeeze-and-Excitation (SE) mechanism[21] focuses only on channel-wise recalibration without explicitly modeling spatially salient regions. These limitations reduce their effectiveness in capturing channel-dependent spatial variations, especially for small and irregular defect features, which often occupy tiny pixel areas and carry weak feature activations in raw images.

To address these limitations, the CPCA module[22] is incorporated into the network, as illustrated in **Figure 4**. CPCA adopts a cascaded architecture in which channel attention is first employed to establish channel priors, followed by channel-aware spatial attention modeling. During the channel attention stage, global average pooling and global max pooling are applied to the input feature maps to capture complementary statistical information. The pooled features are then processed by a shared multilayer perceptron to generate channel-wise attention weights. Through element-wise multiplication, these weights adaptively recalibrate the input features and strengthen channels that contain informative defect-related responses. The resulting channel-prior features are subsequently fed into a depth-wise convolutional module, which captures spatial dependencies independently within each channel and generates channel-specific spatial attention responses, breaking the shared spatial mask restriction of traditional CBAM attention. A convolution-based feature-mixing operation is then performed to promote information interaction across channels and integrate spatial features at different receptive-field scales. Finally, the refined attention features are multiplied element-wise with the channel-prior features to produce the enhanced output representation.

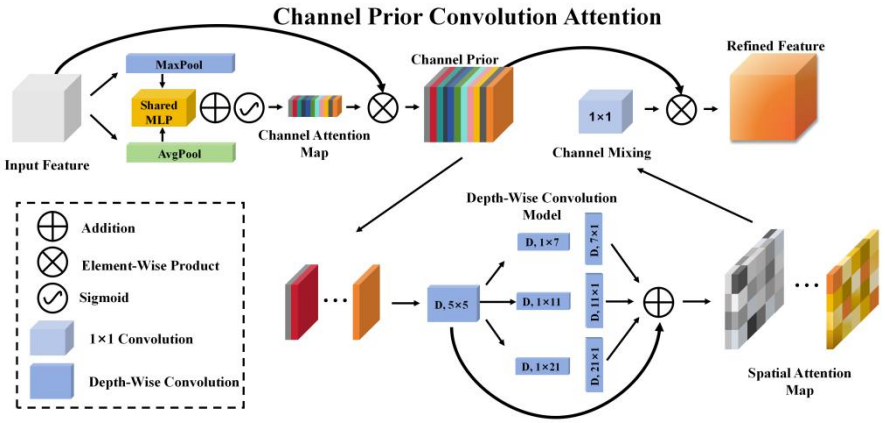


Figure 4. Schematic of the CPCA mechanism. **NOTE:** MLP: Multi-Layer Perceptron; AvgPool: Average pooling. Copyright: Redrawn and adapted from Huang *et al*.[22] using WPS Office Presentation, version 12.1.0.26895.

Overall, CPCA provides more adaptive and discriminative channel-aware spatial feature modeling. For wind turbine blade inspection, this mechanism enables the network to better emphasize subtle defect regions, such as cracks, erosion, delamination, and scratches, while suppressing interference from blade textures, illumination variations, shadows, and complex backgrounds. Consequently, CPCA improves both the localization accuracy and discriminative capability of the network, providing more reliable feature representations for detecting small and inconspicuous blade surface defects under complex inspection conditions, and contributes the largest mean average precision (mAP) gain among all three improved modules according to ablation test data.

2.2.3 DetectAux: Auxiliary Detection Head

To enhance multi-scale feature representation and improve the fusion of features at different semantic levels, the DetectAux head[23] is incorporated into the model. In wind turbine blade surface damage detection, variations in blade geometry, defect morphology, illumination, viewing angle, and background conditions can weaken defect-related feature responses and increase the likelihood of missed detections, particularly for small or low-contrast defects. During training, the auxiliary branches operate in parallel with the primary detection branches, providing additional supervision at multiple levels of the feature pyramid. This auxiliary supervision strengthens gradient propagation, promotes the learning of more discriminative intermediate features, and accelerates network convergence, shortening the training loss decline period and reducing the final model loss value.

As illustrated in **Figure 5**, the DetectAux module contains six detection branches, including three primary detection heads and three auxiliary detection heads operating at different feature-pyramid scales. The primary heads are responsible for generating the final classification and bounding-box regression results, whereas the auxiliary heads provide additional supervision for intermediate feature maps during training. By introducing multi-scale auxiliary supervision, the network can strengthen gradient propagation, improve feature learning at different semantic levels, and enhance its sensitivity to defects with varying sizes and appearances. In particular, the auxiliary branches help the backbone and neck retain more informative spatial and semantic features, thereby reducing missed detections of small or low-contrast blade surface damage. During inference, only the three primary detection heads are retained while the auxiliary branches are removed, so no extra computation overhead is introduced in the practical deployment phase. Therefore, the DetectAux design improves training effectiveness and detection accuracy without introducing additional computational overhead during deployment, allowing the model to maintain efficient real-time inference.

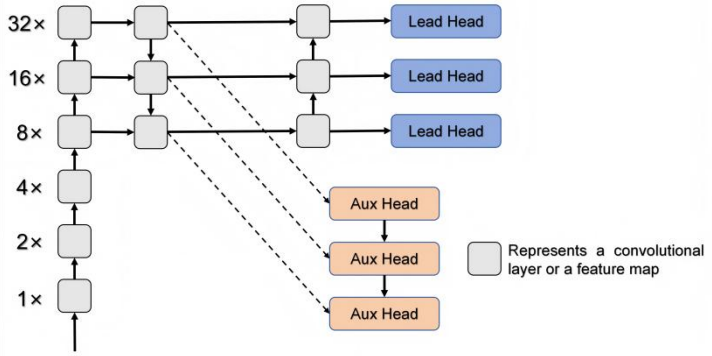


Figure 5. Overall architecture of the DetectAux head. **NOTE:** Aux: auxiliary detection head. Copyright: Redrawn and adapted from Wang *et al.*[23] using WPS Office Presentation, version 12.1.0.26895.

Figure 6 further illustrates the internal structures of the Lead Heads and Aux Heads within the DetectAux module. Both head types adopt a decoupled design in which bounding-box regression and classification are handled by separate branches. In the Lead Head, the cv2 branch generates the distributional predictions used for bounding-box regression with Distribution Focal Loss (DFL), whereas the cv3 branch produces classification scores. Similarly, the Aux Head uses cv4 and cv5 for auxiliary bounding-box (bbox) regression and classification, respectively. The key difference lies in their feature sources and deployment roles: the Lead Heads receive the final neck outputs from Layers 17, 20, and 23 and are retained during inference, whereas the Aux Heads are connected to intermediate features from Layers 17, 14, and 11, with Layer 11 enhanced by CPCA. During training, the auxiliary branches provide additional supervision for multi-scale feature learning, with their corresponding loss terms weighted and combined with the main detection loss to jointly optimize the network parameters.

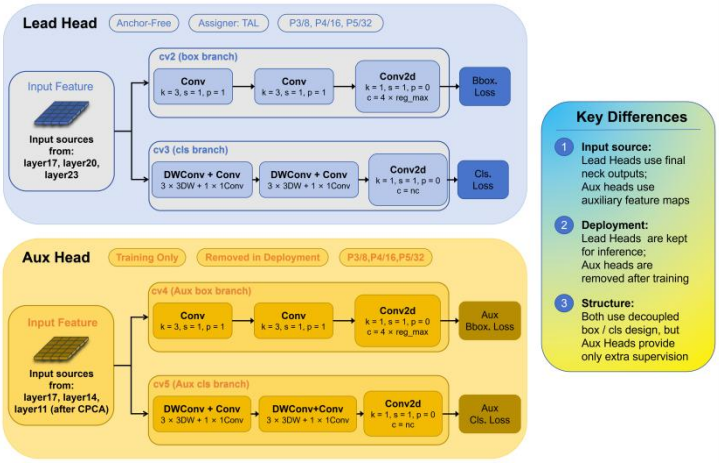


Figure 6. Internal structures of the lead and auxiliary detection heads. **NOTE:** DWConv: Depth-wise Convolution; k: Kernel size; s: Stride; p: Padding; c: Number of output channels; Bbox: bounding-box; Cls: classification; nc: number of classes; TAL: Task Alignment Learning; P3/8: P3 feature-pyramid level with an output stride of 8, and the same notation applies to P4/16 and P5/32. Copyright: Created by the authors using WPS Office Presentation, version 12.1.0.26895.

In summary, incorporating the DetectAux module allows the network to maintain high detection accuracy while operating efficiently during inference. This design improves the ability of the YOLO11-based model to detect small surface defects on wind turbine blades and strengthens its robustness under complex operating conditions, especially for faint damage targets easily covered by shadow or blade surface dirt.

3. Experiments and Analysis

3.1 Dataset

To effectively evaluate the detection performance of FCD-YOLO, this study employed the open-source Wind Turbine Damage Dataset obtained from the Kaggle platform[24,25]. According to the original data source, the dataset comprises inspection images collected in 2017 and 2018 from the same Nordtank wind turbine located at the wind energy facilities of the Technical University of Denmark (DTU) in Roskilde, Denmark. The images were captured under varying lighting conditions, viewing angles, and blade surface conditions. In addition, the dataset creators provided detailed image annotations and predefined defect categories, resulting in a relatively diverse and well-structured dataset for assessing the robustness of defect detection models. The detection targets considered in this dataset were dirt accumulation and blade surface damage, both of which are common surface conditions encountered during wind turbine blade inspection, matching two core detection objects of actual wind farm UAV inspection tasks.

The dataset contains 2,995 labeled images, all of which were resized to 640 pixels × 640 pixels to ensure input consistency and computational efficiency. The predefined training, validation, and test splits provided with the original dataset were retained, comprising 2,395,300, and 300 images, respectively. Representative examples of the dataset are shown in **Figure 7**, covering the blade tip, middle blade and root regions with various light and background conditions.



(a) (b) (c)

Figure 7. Representative examples from the Wind Turbine Damage Dataset. (a) Small dirt spots near the blade tip; (b) A small area of surface damage on the wind turbine blade; (c) Multiple localized regions of coating deterioration along the nacelle panel joint. Copyright: Images obtained from the publicly available Wind Turbine Damage Dataset[25].

3.2 Experimental Environment

Model training and evaluation were conducted on a workstation equipped with an Intel Xeon Platinum 8272CL CPU (Intel Corporation, Santa Clara, CA, USA) and an NVIDIA GeForce RTX 4080 SUPER GPU (NVIDIA Corporation, Santa Clara, CA, USA). The experimental environment was based on Windows 10 (Microsoft Corporation, Redmond, WA, USA), with Python 3.8 (Python Software Foundation, Beaverton, OR, USA), PyTorch 2.0.1 (PyTorch Foundation, San Francisco, CA, USA), and CUDA 12.6 (NVIDIA Corporation, Santa Clara, CA, USA). PyCharm 2024.3.1 (JetBrains s.r.o. Prague, Czech Republic) was used as the integrated development environment. The complete hardware and software configurations are summarized in **Table 2**. All comparative algorithms adopt identical experimental environments to guarantee fair performance comparison.

Table 2. Model training environment

Software or Hardware	Specification
Operating System	Windows 10
CPU	Intel (R) Xeon (R) Platinum 8272CL
GPU	NVIDIA GeForce RTX 4080 SUPER
CUDA	12.6
Programming Language	Python 3.8
Framework	PyTorch 2.0.1
IDE	PyCharm 2024.3.1

NOTE: CPU: Central Processing Unit; GPU: Graphics Processing Unit; CUDA: Compute Unified Device Architecture; IDE: Integrated Development Environment.

The model training hyperparameters are configured as follows: the batch size and number of training epochs are set to 32 and 1,000, respectively, with all input images resized to 640 pixels × 640 pixels. Automatic mixed-precision training is enabled to reduce graphics processing unit (GPU) memory consumption and improve computational efficiency. The learning rate, Intersection over Union (IoU) threshold, and stochastic gradient descent (SGD) momentum factor are set to 0.01, 0.50, and 0.937, respectively. The detailed hyperparameter settings are presented in **Table 3**. In the subsequent comparative experiments, Faster R-CNN, RT-DETR, and the other YOLO-series models are trained under the same key settings wherever applicable, including the dataset split, input image size, batch size, number of training epochs, and evaluation metrics, to ensure a fair and consistent comparison.

Table 3. Training hyperparameters

Parameter Type	Value	Description
Batch Size	32	Controls the number of samples processed per gradient update.
Training Epochs	1,000	Specifies the number of complete passes through the training dataset and influences model convergence; setting a sufficient number of epochs allows all evaluated models to achieve adequate convergence.
Automatic Mixed Precision (AMP)	True	Enables mixed-precision computation to reduce GPU memory consumption and accelerate training while maintaining numerical stability.
Image Size	640	Input image resizing dimension (width × height in pixels), affecting both precision and processing speed.
Intersection over Union (IoU)	0.50	Used to determine whether a predicted bounding box sufficiently overlaps with the corresponding ground-truth box.
Learning Rate	0.01	Controls the step size of parameter updates during optimization and directly influences convergence speed and training stability.
SGD Momentum Factor	0.937	Accelerates gradient descent and reduces oscillations, enhancing optimization stability.

NOTE: GPU: Graphics Processing Unit; SGD: Stochastic Gradient Descent.

3.3 Evaluation Metrics

To objectively evaluate model performance in practical scenarios, this study adopts an evaluation framework with two core aspects: detection accuracy and computational efficiency. Detection accuracy is assessed using precision (P), recall (R), and mAP, while computational efficiency is evaluated in terms of GFLOPs, frames per second (FPS), and parameter count. These metrics provide a comprehensive assessment of detection performance, computational complexity, model size, and real-time inference capability, reflecting the model's practical applicability. The formulations of P, R, and mAP are defined as follows:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2}$$

Precision and Recall are two fundamental metrics used to evaluate object detection performance. Precision measures the proportion of correct detections among all targets predicted by the model and therefore reflects the model's ability to avoid false alarms. Recall measures the proportion of correctly detected targets among all ground-truth targets and reflects the model's ability to minimize missed detections. In the specific calculation process, TP counts true positives correctly identified by the model, FP represents false positives from incorrect predictions, FN indicates false negatives corresponding to missed targets, and N is the number of categories in the dataset.

$$AP = \int_0^1 P(R)dR \tag{3}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{4}$$

According to the IoU threshold adopted for evaluation, mAP can be further categorized into mAP@0.5 and mAP@0.5:0.95. Specifically, mAP@0.5 denotes the mean average precision calculated at a single IoU threshold of 0.50 and reflects the overall detection capability of the model under relatively lenient localization requirements. By contrast, mAP@0.5:0.95 is obtained by averaging the AP values measured at ten IoU thresholds ranging from 0.50 to 0.95 in increments of 0.05. Owing to its stricter requirements for bounding-box overlap, this metric provides a more comprehensive assessment of detection precision, recall, and localization accuracy. The calculation methods for AP and mAP are presented in Equations (3) and (4).

3.4 Ablation Experiment

Table 4 presents the comprehensive results of the ablation experiments conducted using YOLO11s as the baseline model. Each configuration was independently evaluated over five runs with different random seeds, and the results are reported as the mean ± standard deviation to systematically assess the effects of different module combinations on detection performance and computational complexity.

Table 4. Results of ablation experiment

YOLO11s	+ CPCA	+ FasterNet	+ Aux	Precision (%)	Recall (%)	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Parameters (M)	FLOPs (G)	FPS
√				82.79 ± 0.25	75.74 ± 0.29	79.22 ± 0.22	56.08 ± 0.19	9.41	21.30	208 ± 3
√	√			83.27 ± 0.27	76.92 ± 0.34	79.93 ± 0.23	56.61 ± 0.18	9.86	22	212 ± 2
√		√		81.76 ± 0.31	76.54 ± 0.37	79.28 ± 0.26	56.12 ± 0.24	**7.52**	**15.60**	**231** ± 4
√			√	82.83 ± 0.29	76.84 ± 0.35	79.79 ± 0.24	56.54 ± 0.16	9.41	21.30	202 ± 5
√	√	√		83.21 ± 0.26	77.39 ± 0.33	80.04 ± 0.23	57.11 ± 0.20	8.69	19.80	214 ± 3
√	√		√	83.34 ± 0.24	77.91 ± 0.31	80.47 ± 0.21	57.39 ± 0.17	9.86	22	204 ± 5
√		√	√	82.95 ± 0.28	77.63 ± 0.36	79.86 ± 0.25	56.90 ± 0.21	8.24	19.10	217 ± 1
√	√	√	√	**83.44** ± 0.28	**78.30** ± 0.33	**80.98** ± 0.26	**57.77** ± 0.18	8.70	19.80	223 ± 2

NOTE: Values are reported as the mean ± standard deviation over five independent runs with different random seeds. FLOPs: floating-point operations. Bold values indicate the best result in each column among all configurations.

CPCA, FasterNet, and Aux modules improve the baseline model from complementary perspectives. The CPCA module primarily enhances detection accuracy by jointly modeling channel and spatial information, enabling the network to focus more effectively on subtle defect-related regions. Compared with the baseline, CPCA increases Precision, Recall, mAP@0.5, and mAP@0.5:0.95 by 0.48, 1.18, 0.71, and 0.53 percentage points. Similarly, the DetectAux mainly strengthens multi-scale feature learning through additional training supervision. It increases recall by 1.10 percentage points and improves mAP@0.5 and mAP@0.5:0.95 by 0.57 and 0.46 percentage points, which helps reduce missed detections. By contrast, the FasterNet Block primarily improves model compactness and computational efficiency. It reduces the number of parameters from 9.41 M to 7.52 M and the computational complexity from 21.30 GFLOPs to 15.60 GFLOPs, corresponding to reductions of approximately 20.10% and 26.80%, while increasing the inference speed from 208 FPS to 231 FPS. Meanwhile, its mAP@0.5 and mAP@0.5:0.95 slightly increase to 79.28% and 56.12%, indicating that the lightweight design does not compromise overall detection performance, even bringing a minor accuracy gain due to optimized feature extraction mode.

When the three modules are integrated simultaneously, the proposed FCD-YOLO achieves the best overall performance among all configurations. Compared with the original YOLO11s, the improved model not only increases mAP@0.5 from 79.22% to 80.98%, but also reduces the computational complexity from 21.30 GFLOPs to 19.80 GFLOPs while maintaining a high inference speed of 223 FPS. These results indicate that CPCA, FasterNet, and Aux provide complementary benefits, enabling the model to achieve a better balance among detection accuracy, model compactness, and inference efficiency.

3.5 Comparative Analysis of Different Algorithms

To further evaluate the advancement and reliability of the proposed FCD-YOLO model, we conducted comparative experiments against several widely used object detectors, including Faster R-CNN, RT-DETR, and other representative models from the YOLO family. Each model was independently evaluated over five runs with different random seeds, and the results are reported as the mean ± standard deviation to provide a more reliable comparison of model performance. The detailed results for all evaluation metrics are presented in **Table 5**, covering two-stage detectors, transformer-based real-time detectors and mainstream YOLO variants released in recent years.

Table 5. Results of comparative experiment

Models	Precision (%)	Recall (%)	mAP@0.5 (%)	mAP@0.5:0.95 (%)	Parameters (M)	FLOPs (G)	FPS
Faster R-CNN	51.75 ± 0.62	70.35 ± 0.71	61.72 ± 0.55	40.54 ± 0.48	36.72	145.50	42 ± 2
RT-DETR	66.62 ± 0.54	62.33 ± 0.66	68.51 ± 0.43	46.27 ± 0.39	31.99	103.70	114 ± 4
YOLOv8	82.62 ± 0.31	76.15 ± 0.42	79.03 ± 0.28	55.10 ± 0.24	11.13	28.40	217 ± 1
YOLOv10	82.35 ± 0.34	72.23 ± 0.51	78.86 ± 0.37	56.41 ± 0.32	**8.40**	24.40	**289** ± 5
YOLO12	82.28 ± 0.30	77.39 ± 0.36	80.35 ± 0.24	56.64 ± 0.22	9.07	**19.30**	187 ± 1
YOLO13	74.57 ± 0.47	74.52 ± 0.49	79.37 ± 0.31	51.19 ± 0.25	9	20.00	186 ± 3
Ours (FCD)	**83.44** ± 0.28	**78.30** ± 0.33	**80.98** ± 0.26	**57.77** ± 0.18	8.70	19.80	223 ± 2

NOTE: Faster R-CNN: Faster Region-Based Convolutional Neural Network; RT-DETR: Real-Time Detection Transformer. Values are reported as the mean ± standard deviation over five independent runs with different random seeds. Bold values indicate the best result in each column among all compared models.

As shown in **Table 5**, FCD-YOLO achieves the best overall detection performance among the evaluated models, with a precision of 83.44%, a recall of 78.30%, an mAP@0.5 of 80.98%, and an mAP@0.5:0.95 of 57.77%. Compared with the second-best results, FCD-YOLO improves precision, recall, mAP@0.5, and mAP@0.5:0.95 by 0.82, 0.91, 0.63, and 1.13 percentage points, respectively. These improvements are moderate but consistent across all four accuracy metrics, indicating a more balanced detection capability.

FCD-YOLO also maintains relatively low model complexity, with 8.70 M parameters and 19.80 GFLOPs. Its parameter count is lower than those of Faster R-CNN, RT-DETR, YOLOv8, YOLO12, and YOLO13, while remaining slightly higher than that of YOLOv10. In terms of computational cost, FCD-YOLO requires only 0.50 GFLOPs more than YOLO12 and substantially fewer FLOPs than Faster R-CNN, RT-DETR, YOLOv8, and YOLOv10. With an inference speed of 223 FPS, FCD-YOLO ranks second among all evaluated models, surpassed only by YOLOv10. Although YOLOv10 achieves a markedly higher inference speed and YOLO12 requires slightly fewer FLOPs, both models exhibit lower overall detection accuracy than FCD-YOLO. In particular, FCD-YOLO improves recall over YOLOv10 by 6.07 percentage points while also achieving higher values for both mAP metrics, which is critical for wind turbine inspection to avoid missing potential blade damage hazards.

Overall, these results indicate that FCD-YOLO provides a favorable balance among detection accuracy, model complexity, and inference efficiency. In the application scenario considered in this study, UAV-acquired images and videos are transmitted to a local workstation for offline or near-real-time analysis rather than processed directly on onboard edge devices. Under this setting, the inference speed of 223 FPS is sufficient to support efficient wind turbine blade surface damage inspection, fully meeting the processing demand of the UAV video stream returned to the ground workstation.

3.6 Visual Evaluation Experiment

To further evaluate the effectiveness and practical robustness of the proposed improvements, several representative images were selected from the wind turbine blade damage test set and processed using offline data augmentation to simulate challenging inspection conditions. The simulated scenarios included low image contrast, intense illumination, shadow interference, and surface contamination with visual characteristics similar to those of actual blade damage. These disturbances are frequently encountered during UAV-based wind turbine inspection and may obscure defect features, reduce the contrast between damaged and intact regions, or cause background interference to be misidentified as damage. Moreover, these

augmented images were used solely for post-training qualitative evaluation and were not included in model training or hyperparameter selection, eliminating data leakage risk in the model training stage. The qualitative detection results are presented in **Figure 8**.

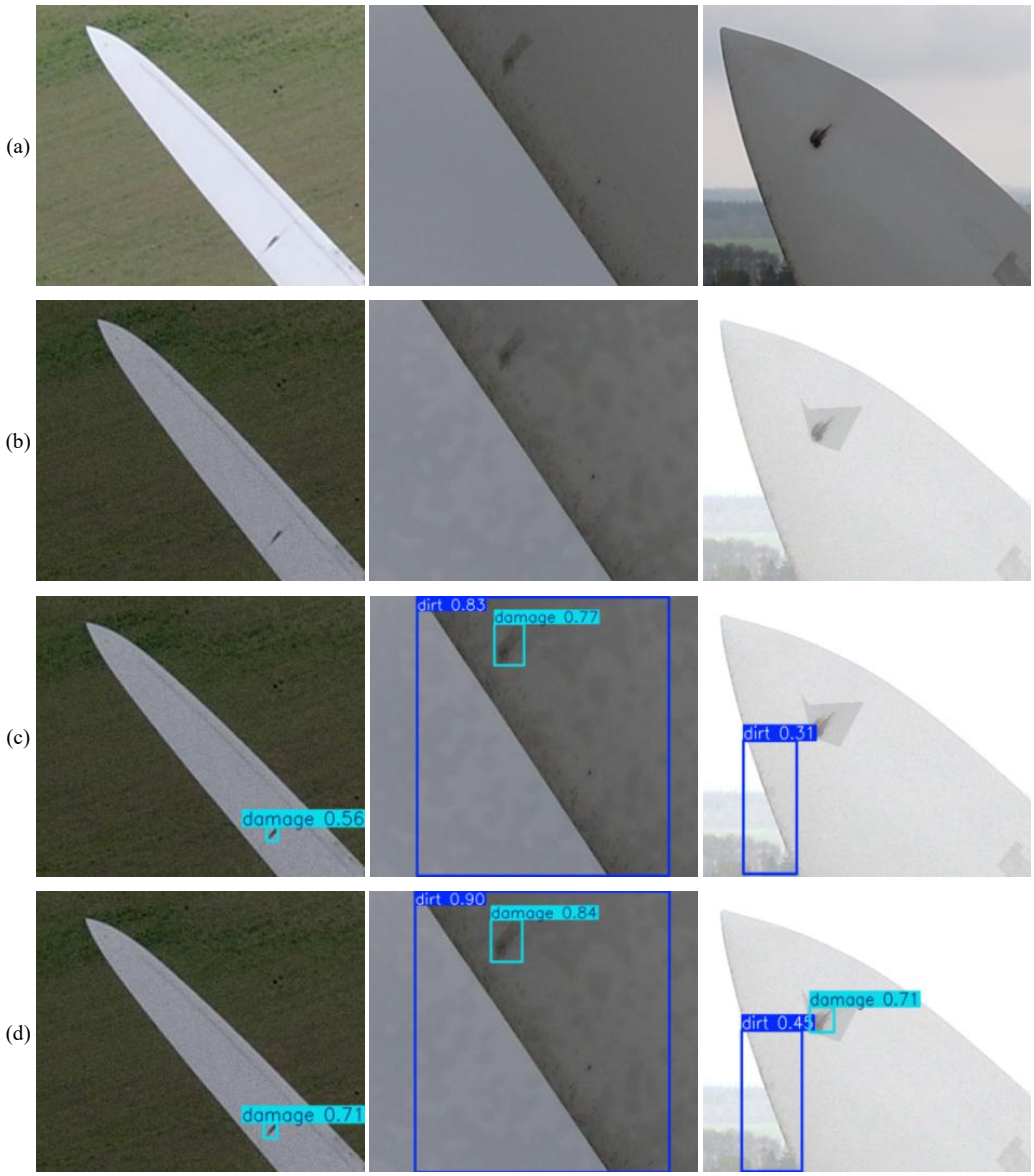


Figure 8. Visual evaluation of detection outcomes. (a) Original image; (b) Augmented image; (c) YOLO11s; (d) FCD-YOLO. Copyright: The original images were obtained from the Wind Turbine Damage Dataset[25]; image augmentation and detection visualization were performed by the authors.

The visualization results show that YOLO11s exhibits reduced detection stability under these complex conditions. In particular, it produces inaccurate localization and occasional missed detections when defect features are weakened by low contrast, intense illumination, shadows, or interference from surface contamination. By comparison, FCD-YOLO demonstrates more consistent detection and localization performance on the same test images, with fewer missed detections and more reliable localization of blade surface damage. These results indicate that the proposed improvements enhance the model's ability to distinguish genuine damage from visually similar interference patterns and improve its robustness in complex UAV inspection environments.

4. Conclusions and Future Prospects

This study proposed FCD-YOLO, an improved object detection framework for wind turbine blade surface damage detection in UAV-acquired images. FasterNet Blocks were introduced into the backbone and neck to reduce computational redundancy, while the CPCA module was incorporated to enhance channel-spatial feature representation. In addition, the DetectAux head was employed during training to strengthen multi-scale supervision and improve feature learning for small and irregular defects. FCD-YOLO achieved a precision of 83.44%, a recall of 78.30%, an mAP@0.5 of 80.98%, and an mAP@0.5:0.95 of 57.77%, with 8.70 million parameters and 19.80 GFLOPs. These results demonstrate that the proposed model achieves a favorable balance between detection accuracy and computational efficiency, supporting workstation-based offline or near-real-time analysis of UAV inspection images and outperforms all mainstream contrast algorithms on the public wind turbine blade defect dataset.

Although FCD-YOLO achieves a high inference speed on a local workstation, the present study mainly focuses on offline or near-real-time analysis after UAV image transmission rather than

direct onboard processing, limiting its direct deployment on lightweight UAV edge computing equipment at the current stage. Future studies could evaluate its deployment on resource-constrained embedded platforms, such as NVIDIA Jetson devices, and explore pruning, quantization, TensorRT acceleration, and lightweight structural optimization to reduce computational and memory requirements while preserving detection accuracy, realizing end-to-end real-time detection onboard UAVs. Beyond deployment-oriented optimization, the proposed framework could also be extended from surface damage detection to damage characterization and fatigue life prediction[26]. Future research could quantitatively characterize defects in terms of their type, size, location, and morphological features, integrate this information with operational loads, environmental conditions, and maintenance records, and explore a comprehensive framework that combines visual defect detection, quantitative damage assessment, and fatigue life prediction, thereby further enhancing the applicability of UAV-based wind turbine blade inspection in engineering practice, constructing a full-process intelligent health monitoring system for wind turbine blades.

Author Contributions

Conceptualization, Rui Zhao; methodology, Rui Zhao and Zixiang Yue; formal analysis and investigation, Chenyu Wang; supervision, Zixiang Yue; writing—original draft preparation, Chenyu Wang; writing—review and editing, Rui Zhao, Chenyu Wang, and Zixiang Yue. All authors have read and approved the final version of the manuscript and agree to be accountable for all aspects of the work.

Data Availability

The original Wind Turbine Damage Dataset is publicly available at the Kaggle repository. Additional processed data supporting the findings of this study are available from the corresponding author upon reasonable request, including trained model weights, ablation test logs, and visualization result pictures.

Acknowledgments

The authors would like to thank the senior members of our research group for their valuable technical guidance and constructive suggestions throughout this study. The authors also gratefully acknowledge the Xinjiang Key Laboratory of Building Structure and Earthquake Resistance for providing the experimental facilities and computational resources required for this research.

Conflicts of Interest

The authors declare no competing financial interests, no commercial institutions participated in the design and execution of this research work.

References

[1] Zou L, Chen A, Yang X, *et al.* An improved method of AUD-YOLO for surface damage detection of wind turbine blades. *Scientific Reports*. 2025, 15 (1): 1-16. [Google Scholar] [CrossRef]

[2] Global Wind Energy Council. Global Wind Report 2025. Lisbon, Portugal: Global Wind Energy Council, 23 April 2025. Available online: https://www.gwec.net/reports/globalwindreport/2025 (accessed on 3 September 2026).

[3] Jureczko M, Pawlak M, Mężyk A. Optimisation of wind turbine blades. *Journal of Materials Processing Technology*. 2005, 167 (2-3): 463-471. [Google Scholar] [CrossRef]

[4] Kaewniam P, Cao M, Alkayem NF, *et al.* Recent advances in damage detection of wind turbine blades: A state-of-the-art review. *Renewable and Sustainable Energy Reviews*. 2022, 167: 1-24. [Google Scholar] [CrossRef]

[5] McKenna R, Ostman v.d. Leye P, Fichtner W. Key challenges and prospects for large wind turbines. *Renewable and Sustainable Energy Reviews*. 2016, 53: 1212-1221. [Google Scholar] [CrossRef]

[6] Pan X, Liu Z, Xu R, *et al.* Early warning of damaged wind turbine blades using spatial-temporal spectral analysis of acoustic emission signals. *Journal of Sound and Vibration*. 2022, 537: 1-14. [Google Scholar] [CrossRef]

[7] Sheiati S, Chen X. Advances in computer vision-based structural health monitoring techniques for wind turbine blades. *Renewable and Sustainable Energy Reviews*. 2025, 224: 1-18. [Google Scholar] [CrossRef]

[8] Tong L, Fan C, Peng Z, *et al.* WTBD-YOLOv8: An improved method for wind turbine generator defect detection. *Sustainability*. 2024, 16 (11): 1-17. [Google Scholar] [CrossRef]

[9] Hu W, Fang J, Zhang Y, *et al.* Digital twin of wind turbine surface damage detection based on deep learning-aided drone inspection. *Renewable Energy*. 2025, 241: 1-21. [Google Scholar] [CrossRef]

[10] Li X, Tian J, Pang X, *et al.* Wind turbine blade defect recognition method based on large-vision-model transfer learning. *Sensors*. 2025, 25 (14): 1-21. [Google Scholar] [CrossRef]

[11] Zhou Q, Zhang Y, Chen Z, *et al.* Few-shot visual reasoning for wind turbine blade damage detection via RAG with vision-language model. *IFAC-PapersOnLine*. 2025, 59 (30): 37-42. [Google Scholar] [CrossRef]

[12] Gohar I, Halimi A, Weng Kean DY, *et al.* Addressing class scarcity and imbalance for few-shot detection of wind turbine blade surface defects. 2025 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), Bandung, Indonesia, 04-07 November 2025. Available online: https://ieeexplore.ieee.org/abstract/document/11383379 (accessed on 3 September 2026).

[13] Ren S, He K, Girshick R, *et al.* Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2017, 39 (6): 1137-1149. [Google Scholar] [CrossRef]

[14] Zhao Y, Lv W, Xu S, *et al.* DETRs beat YOLOs on real-time object detection. 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16-22 June 2024. Available online: https://ieeexplore.ieee.org/abstract/document/10657220 (accessed on 3 September 2026).

[15] Redmon J, Divvala S, Girshick R, *et al.* You only look once: Unified, real-time object detection. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27-30 June 2016. Available online: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=10.1109%2Fcvpr.2016.91&btnG= (accessed on 3 September 2026).

[16] Redmon J, Farhadi A. YOLOv3: An incremental improvement. 2018. Computer Science, Computer Vision and Pattern Recognition. Available online: https://arxiv.org/abs/1804.02767 (accessed on 3 September 2026).

[17] Terven J, Córdova-Esparza DM, Romero-González JA. A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*. 2023, 5 (4): 1680-1716. [Google Scholar] [CrossRef]

[18] Wang A, Chen H, Liu L, *et al*. YOLOv10: Real-time end-to-end object detection. Advances in Neural Information Processing Systems 37 (NeurIPS 2024), Vancouver, Canada, 10-15 December 2024; 107984-108011. Available online: https://proceedings.neurips.cc/paper_files/paper/2024/hash/c34ddd05eb089991f06f3c5dc36836e0-Abstract-Conference.html (accessed on 3 September 2026).

[19] Chen J, Kao SH, He H, *et al*. Run, don't walk: Chasing higher FLOPS for faster neural networks. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17-24 June 2023. Available online: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=10.1109%2Fcvpr52729.2023&btnG= (accessed on 3 September 2026).

[20] Woo S, Park J, Lee JY, *et al*. CBAM: Convolutional block attention module. Computer Vision — ECCV 2018, Munich, Germany, 06 October 2018; 3-19. Available online: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=10.1007%2F978-3-030-01234-2_1&btnG= (accessed on 3 September 2026).

[21] Hu J, Shen L, Albanie S, *et al*. Squeeze-and-excitation networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, Calvin L. Rampton Salt Palace Convention Center, United States of America, 18-22 June 2018; 2011-2023. Available online: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=10.1007%2F978-3-030-01234-2_1&btnG= (accessed on 3 September 2026).

[22] Huang H, Chen Z, Zou Y, *et al*. Channel prior convolutional attention for medical image segmentation. *Computers in Biology and Medicine*. 2024, 178: 1-11. [Google Scholar] [CrossRef]

[23] Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17-24 June 2023; 7464-7475. Available online: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=10.1109%2Fcvpr52729.2023.00721&btnG= (accessed on 3 September 2026).

[24] Shihavuddin ASM, Chen X, Fedorov V, *et al*. Wind turbine surface damage detection by deep learning aided drone inspection analysis. *Energies*. 2019, 12 (4): 1-15. [Google Scholar] [CrossRef]

[25] *Wind Turbine Damage Dataset (Faster RCNN format)*, Version 1, kaggle, 2025. Available online: https://www.kaggle.com/datasets/stmlen/nordtank-windturbine-dataset-faster-rcnn-format (accessed on 3 September 2026).

[26] Zhang Y, Wang XK, Jia YF, *et al*. Fatigue life prediction and feature contribution analysis of surface-strengthened Ti-6Al-4V using a preprocessing neural network. *Metallurgical and Materials Transactions A*. 2025, 56: 4991-5005. [Google Scholar] [CrossRef]