machines

MDPI

Article

ADCF-YOLO: Aligning, Stabilizing and Decoupling Multi-Scale Features for Fine-Grained Refueling Interface Detection

Shengyou Zhou, Wen Cui, Shiming Chen, Zhaojie Wu, Gang Fang, Weixing Hua * and Yan Chen *

PLA Joint Logistics Support Force University of Engineering, Chongqing 401331, China; zhousyou@126.com (S.Z.); gwencuiwang@163.com (W.C.); skycsm@126.com (S.C.); georgewu2019@163.com (Z.W.); 18986966529@163.com (G.F.)
* Correspondence: huaweixing@163.com (W.H.); yansohucom@sohu.com (Y.C.)

Abstract

High-precision detection of refueling interfaces on special equipment is a prerequisite for robots to carry out intelligent refueling operations. However, the lack of datasets in this field, together with the presence of multiple connectors that are visually similar but functionally different in real refueling environments, makes existing methods difficult to transfer effectively. To this end, we constructed the Refueling Interface Dataset, which contains 10 categories, 946 images and 7659 bounding boxes, with an average class-wise cosine similarity of 0.956, forming a fine-grained benchmark of visually similar, weakly textured connectors. Based on YOLO26, we propose ADCF-YOLO, which improves cross-scale feature fusion in the neck and the training supervision. Adaptive Deformable Cross-scale Fusion compensates for the spatial misalignment introduced by upsampling and convolution through a neighborhood convex combination and fuses the two paths with a zero-initialized residual gate. Sub-pixel Phase-Stable Detection (SPSD) exploits the known and invertible sub-pixel shift to map shifted-view predictions back to the original coordinate system, converting an invariance requirement into dense, unambiguous pixel-wise supervision. Counterfactual Context Decoupling (CCD) applies two oppositely directed constraints—annulus replacement and target erasure—to suppress reliance on background features. On the refueling interface dataset, ADCF-YOLO raises mAP@0.5 from 0.934 to 0.959 and mAP@0.5:0.95 from 0.482 to 0.526 over YOLO26n, the highest of the six YOLO-family detectors at this scale, and yields positive gains on two public datasets. The two Transformer-based reference detectors that score higher do so by 0.002–0.007 on the two mAP metrics, at 7.2–16.4 times the GFLOPs and 11.5–11.9 times the parameters. The engineering value lies in deployment: SPSD and CCD act only during training and leave the inference graph and its cost unchanged, so the accuracy gain carries no additional inference cost.

Keywords: refueling interface; ADCF-YOLO; sub-pixel phase consistency; counterfactual context decoupling

Check for updates

Academic Editor: Dan Zhang

Received: 12 October 2026
Revised: 11 November 2026
Accepted: 17 November 2026
Published: 18 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Fuel supply is a core element of sustainment for special vehicles in field operations and sustained missions. The conventional refueling mode relies on manual operation, which is inefficient and also carries the risk of fuel vapor leakage and coupling damage [1]. As artificial intelligence advances rapidly, having robots perform refueling automatically can both improve efficiency and avoid accident hazards [2]. In refueling robot operations, however, the refueling interface must be detected with high precision. The detection result

not only decides whether an interface belongs to the target category but also serves as the input to subsequent pose estimation and compliant docking [3]. Engineering applications therefore demand high detection accuracy, robustness, and real-time performance.

Existing studies, however, concentrate on the detection of fuel tank caps on commercial vehicles. Vision-based tank cap detection techniques fall into three stages. The traditional image processing stage. Zhang et al. [4] combined morphological gradients with the Hough transform and raised the recognition accuracy of tank caps from 72% to 81%. Liang et al. [5] improved the Canny operator with pseudo-sphere filtering and inner-product energy to extract the edge features of tank caps. This stage requires handcrafted feature descriptors and comes with low detection accuracy, poor generalization, and slow computation. The machine learning stage. Liang et al. [6] applied multi-scale bottom-hat transform and improved maximum entropy threshold segmentation, extracted HOG features from the candidate regions, and classified them with an SVM, reaching 96.2% recognition accuracy. Li et al. [7] built classifiers over the geometric, texture, and color features of tank caps and achieved 99% recognition accuracy. This stage raised detection accuracy considerably, but the network's ability to extract object features remained weak. The deep learning stage. Zhang et al. [8] improved Faster R-CNN with ResNet and reached 98% accuracy. Recent work takes YOLOv8 as the baseline and pursues lightweight models and higher detection accuracy. Fu et al. [9] introduced BiFPN and C2f-Faster and reduced the parameter count by 56.9%. Qiao et al. [10] combined FastViT, Mamba, and AFPN to achieve global modeling and lower computational complexity. Zhang et al. [11] reached 99.2% mAP@0.5 with the improved BES-YOLO. Dai et al. [12] built SL-YOLO on a StarNet backbone and used pruning to cut the parameter count by 80.6%, reaching 98.7% mAP@50. This stage exploits the strong fitting capacity of neural networks to let the network learn object features automatically, improving detection accuracy and generalization.

Although these studies have made marked progress in detection accuracy and generalization, transferring them directly to refueling interface detection on special equipment raises the following problems. (1) Compared with fuel tank caps on commercial vehicles, refueling interfaces on special equipment differ clearly in structure and docking mode. In the public literature, studies dedicated to such detection are very limited, and no public dataset is available. (2) Neck feature fusion in existing studies usually adopts Concat, which cannot correct the spatial displacement caused by upsampling and convolution. (3) Under discrete grid sampling, aliasing of the few-pixel discriminative details causes localization jitter, and training lacks a consistency regularization mechanism. (4) When the categories in a dataset are highly similar in appearance, the model relies on background features rather than the object's own features when discriminating the target category.

To address these challenges, we propose a detector based on YOLO26, named ADCF-YOLO. It is tailored to the characteristics of the refueling interface dataset and optimizes neck feature fusion and training supervision, with the aim of improving the accuracy and robustness of refueling interface detection. The specific contributions of this paper are as follows:

(1) Dataset construction. We constructed the refueling interface dataset, which contains 10 connectors that are highly similar in appearance but functionally different, comprising 946 images and 7659 bounding boxes. The acquisition scenes cover indoor and outdoor environments and different illumination conditions. The average inter-class cosine similarity of the dataset is 0.97, providing a representative benchmark for fine-grained visual similarity in weakly textured connectors.

(2) Adaptive Deformable Cross-scale Fusion (ADCF). ADCF adopts the strategy of aligning spatial positions first and then enhancing feature fusion. The input features are first divided into an alignment branch and an enhancement branch. The alignment

branch predicts a discrete probability distribution at each spatial position and compensates, through a neighborhood convex combination, for the spatial misalignment introduced by upsampling and convolution. The enhancement branch applies depthwise separable convolution to enhance the detail features of this path. The fusion weight of the two paths is finally determined by a gating mechanism.

(3) Sub-pixel Phase-Stable Detection (SPSD). During training, a known sub-pixel translation is applied to the input image to construct a phase-shifted view; using the prior that the shift is known and analytically invertible, the student predictions are mapped back to the original coordinate system and compared with the teacher predictions pixel by pixel. Weak texture object regions and high teacher confidence are used to select the supervised locations, turning an abstract invariance requirement into dense supervision without matching ambiguity.

(4) Counterfactual Context Decoupling (CCD). During training, two types of counterfactual views are constructed, namely the annulus background replacement view and the target erasure view. Training pushes the model to keep its original prediction after the background is replaced and requires no response when the target is erased. Annulus replacement guarantees that the intervention variable is unique, and the softplus one-sided penalty strongly suppresses the response after target erasure.

2. Related Work

2.1. Multi-Scale Feature Fusion

In object detection, cross-scale fusion is the main means of feature enhancement. FPN [13] adopts a top-down propagation path, and PANet [14] adds a bottom-up path on top of it; together, they form a bidirectional pyramid and have become the paradigm of neck feature fusion in the YOLO family. In this bidirectional pyramid, however, all four fusion nodes concatenate along the channel dimension and do not distinguish the contributions of the two paths. Recent improvements proceed in two directions. The first is adaptive weight fusion. Such methods hold that features from different paths should carry different weights. BiFPN [15] introduces a learnable weight for each path, and ASFF [16] further assigns fusion weights adaptively at every position in the spatial dimension. The second is the resampling operator. Such methods target information fidelity during resampling and suppress the error that resampling introduces by making the sampling pattern vary adaptively with the feature. CARAFE [17] performs feature upsampling adaptively with a content-aware kernel. DySample [18] builds a dynamic upsampler from the perspective of point sampling, while the deformable convolutions DCN [19] and DCNv2 [20] let the sampling positions of the convolution vary adaptively with the perceived content. Weighted fusion and resampling operators do outperform concatenation and fixed interpolation, respectively. The two families nevertheless share a common limitation. Weighted fusion only changes the relative contributions of the two paths; it does not correct the displacement that the features acquire during resampling. A resampling operator aims at the optimal reconstruction of its own path rather than the optimal registration between the two paths, so it cannot perceive where the other path actually lies. Moreover, neither weighted fusion nor resampling operators impose an explicit constraint on the phase consistency of cross-scale features, so the sub-pixel misalignment that discrete grid sampling introduces accumulates level by level along the hierarchy. How to effectively correct the sub-pixel displacement that has already occurred between the two paths therefore remains an open problem in feature fusion.

2.2. Object Detection and Translation Invariance

Work on object detection falls into three lines. Data and loss. Low proportion of objects in a dataset. Kisantal et al. [21] proposed oversampling and instance augmentation, and Ghiasi et al. [22] adopted a copy–paste strategy that brought clear gains in instance segmentation. Imbalance between positive and negative samples and between easy and hard samples. Focal Loss [23] suppresses the gradient contribution of easy negatives by re-weighting; GHM [24] unifies the imbalance between easy and hard samples and between positive and negative samples from the perspective of the gradient distribution; and OHEM [25] raises the participation of hard samples through online hard example mining. Structure and inference. Raising the input resolution or processing object regions separately is the mainstream practice. SAHI [26] performs inference on overlapping slices of a high-resolution image. Translation invariance. Unlike the two preceding lines, this one attacks the root of the problem. Downsampling in a convolutional network aliases high-frequency components into low-frequency ones, so a one-pixel translation changes the feature map. Zhang et al. [27] restored translation invariance by inserting BlurPool before pooling and convolution, and Azulay et al. [28] quantified this phenomenon systematically, showing that data augmentation brings only local invariance for samples sufficiently similar to the training set. In addition, sub-pixel convolution [29] showed that a resolution transform can be parameterized explicitly at sub-pixel accuracy, providing a basis for modeling at the phase level. In the semi-supervised and self-supervised setting, consistency regularization is widely used to improve the stability of a model against input perturbations. Mean Teacher [30] provides a stable target with an exponential moving average teacher; FixMatch [31] supervises a strongly augmented view with pseudo-labels from a weakly augmented one; and Unbiased Teacher [32] and SoftTeacher [33] extend this idea to detection, improving pseudo-label quality through teacher-confidence weighting and box jittering.

Although the studies above improve the accuracy of object detection, BlurPool attains translation invariance through low-pass filtering, which suppresses high-frequency detail. The cues that distinguish the categories of the refueling interface dataset, however, lie mainly in high-frequency components of weak texture. Consistency regularization, moreover, lacks the step of explicit alignment under a known shift. In this work, we therefore take the known and analytically invertible shift as a prior, map the predictions of the shifted view back to the original image coordinate system, and compute the loss pixel by pixel, turning an invariance requirement into dense supervision without matching ambiguity.

2.3. Context Bias and Decoupling Under Weak Discriminative Evidence

FGIA [34] points out that the fundamental challenge in this area is small inter-class differences together with large intra-class differences. Research therefore focuses on locating and magnifying discriminative local regions. In metric learning, ArcFace [35] compresses intra-class distances and enlarges inter-class angles to increase the discriminative margin. When discriminative evidence is scarce, the model no longer relies on the object's own features but discriminates with the features of the object and its background. Su et al. [36] proposed context decoupling augmentation, which forces the model to attend to the object itself by replacing the region around it. Causal-YOLO [37] weakens spurious correlations through causal feature alignment. Cutout [38], a region-level occlusion augmentation, improves generalization by creating missing information. RSC [39] improves generalization by training in a way that suppresses dominant features and activates the remaining ones. The categories of the refueling interface dataset we construct, however, are highly similar in appearance, and their differences lie mainly in weak texture detail. Most of the above work is centered on suppressing interference, but it applies no positive constraint that drives the

model to learn the object's own features and therefore cannot rule out the possibility that the model depends on background features. Region-level occlusion or random replacement changes the visibility of the object and the background features at the same time, which makes it hard to attribute the prediction to background dependence and thus impossible to construct an interpretable supervision signal. A next step is therefore to apply annulus background replacement and target erasure together, keeping the object region intact and the context complete, respectively. Annulus replacement guarantees that the intervention variable is unique, and the two constraints form an oppositely directed pair.

3. Materials and Methods

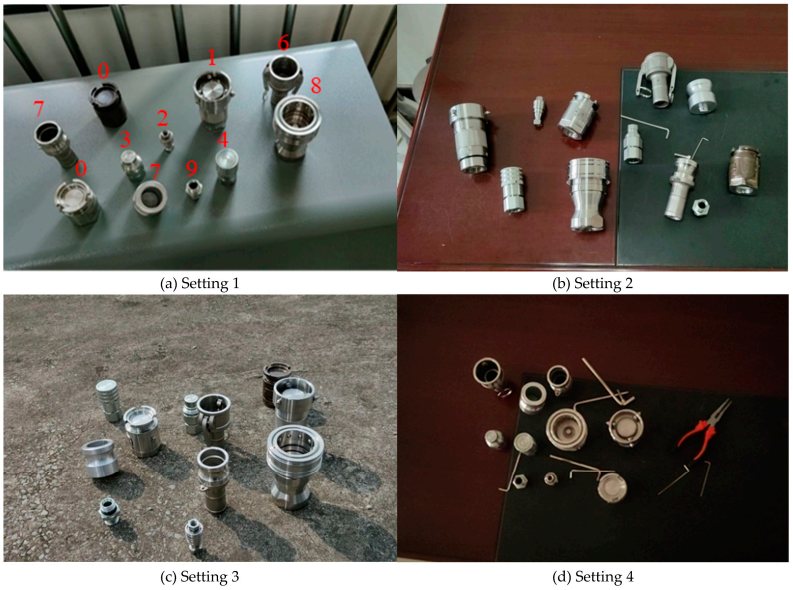
3.1. Dataset

3.1.1. Data Acquisition and Processing

Acquisition background: The refueling interface is made of stainless steel and has the shape of a symmetric cylinder. In a real refueling environment, the workspace of the refueling interface may contain equipment connectors that look similar in appearance but differ in function. If the detector identifies them wrongly, the refueling task fails and may even cause equipment damage and fuel vapor leakage. In this work, we therefore acquired 10 kinds of equipment connectors that are similar in appearance but different in function, and named the collection the refueling interface dataset.

Acquisition equipment and procedure: All images were newly collected, rather than taken from existing datasets. We collected the data with an Intel RealSense D435i (RealSense, Inc., Cupertino, United States). The 10 categories were randomly combined into 8 shooting scenes. The scenes cover indoor and outdoor environments as well as natural, strong and low illumination. During acquisition, each instance was fixed in the scene, the camera was held by hand and the shots were taken from multiple viewpoints, which introduces occlusion and truncation at various ratios. The shooting distance was 0.5–5 m, which reproduces the working distance of a refueling robot and at the same time introduces multi-scale variation of the objects. The collected data are shown in Figure 1, where Figure 1a is an indoor overexposed scene, Figure 1b is an indoor light scene, Figure 1c is an outdoor natural light scene, and Figure 1d is an indoor low-light scene.

Annotation and split: Two professionals annotated all images independently with LabelImg (v1.8.6). Before annotation, the structural features distinguishing the ten categories were specified in writing (notch shape, number of screws, flange diameter), and a single unified criterion was fixed for difficult samples: an instance is annotated when its discriminative structure is at least partially visible, and discarded when occlusion exceeds roughly two-thirds of the instance, when the discriminative structure is fully cut off by the image border, or when specular reflection removes it entirely. In order to avoid near-duplicate frames falling into the training and evaluation subsets at the same time, we divide the data set according to the scene. Among them, 5 scenes are used for training, 2 scenes are used for validation, and 1 scene is used for testing, corresponding to the number of pictures and 675, 190, and 81, and the corresponding number of labeled boxes is 5460, 1389, and 810, respectively.



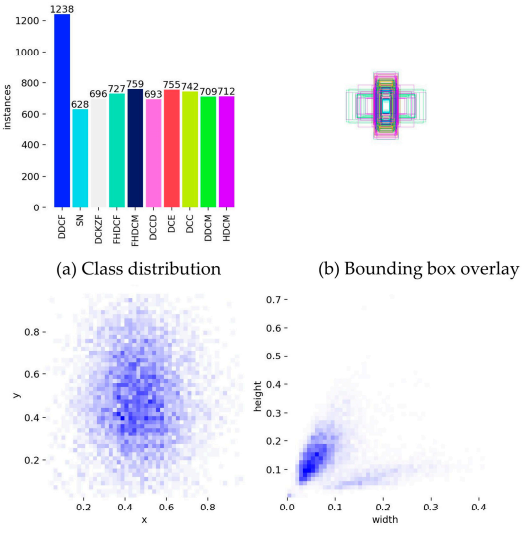
(a) Setting 1 (b) Setting 2

(c) Setting 3 (d) Setting 4

Figure 1. Sample images from the refueling interface dataset. (**a**) Setting 1; (**b**) Setting 2; (**c**) Setting 3; (**d**) Setting 4. The class indices annotated in red in (**a**) apply to all four settings: 0: Dry disconnect couplings female (DDCF), 1: Dry disconnect couplings male (DDCM), 2: Hydraulic disconnect couplings male (HDCM), 3: Flat hydraulic disconnect couplings female (FHDCF), 4: Flat hydraulic disconnect couplings male (FHDCM), 5: Disconnect couplings-E (DCE), 6: Disconnect couplings-C (DCC), 7: Disconnect couplings-CD (DCCD), 8: Disconnect couplings-KZF (DCKZF), 9: Screw nut (SN).

3.1.2. Dataset Description

Figure 2 gives the statistics of the refueling interface dataset. Figure 2a shows the number of instances per category: 10 categories, 946 images, 7659 bounding boxes, and 8.1 instances per image on average.



(a) Class distribution (b) Bounding box overlay

(c) Object center distribution (d) Box width-height distribution

Figure 2. Statistics of the refueling interface dataset. (**a**) Class distribution; (**b**) Bounding box overlay; (**c**) Object center distribution; (**d**) Box width–height distribution.

The most frequent category is DDCF (1238 instances, 16.16%), and the least frequent category is SN (628 instances, 8.20%). The ratio between the two is 1.97:1, far below the 10:1 threshold commonly used to flag long-tailed data, and the nine non-dominant classes all lie

within 8.2–9.9%. The dataset is therefore not long-tailed in the frequency sense. Figure 2b is the bounding-box overlay of the dataset, which reflects the spatial bias of the objects. Figure 2c is the spatial heat map of the object centers (x, y). Figures 2b and 2c show that the objects are concentrated in the lower middle part of the image and that no extreme positional bias is present. Figure 2d is the width–height scatter plot of the bounding boxes; the great majority of the objects occupy 5–15% of the image width and 10–20% of the image height. Following the COCO [40] definition, 25.1% of the boxes are small (area below 32^2 px), 73.0% medium (32^2 to 96^2 px) and 1.9% large (above 96^2 px) at the native image resolution. The dataset is therefore dominated by medium objects: the difficulty it poses is the similarity of appearance among ten weakly textured categories, not the small size of the objects.

Figure 3 shows the cosine similarity of the 10 categories obtained with a ResNet-50 [41]. The similarity is computed on instance crops, not on whole images and not on backbone feature maps. Each ground-truth box is expanded by a 10% margin, cropped to the image boundary, resized bilinearly to 224 × 224 and normalized with the ImageNet channel statistics; boxes with a shorter side below 4 px are discarded. Each instance is described by the 2048-d output of the global-average-pooling layer of a ResNet-50 pretrained on ImageNet-1k with its classification layer removed. Each category is represented by the arithmetic mean of its instance descriptors (class prototype), with at most 500 instances per category sampled uniformly. The similarity of two categories is the cosine of their prototypes. Two averages are reported separately: the mean over all 45 off-diagonal pairs of the 10 × 10 matrix, and the mean of the nine pairs involving the target category DDCF. Intra- and inter-class values are compared only at the same granularity (instance pairs), because averaging over instances cancels instance-level noise and makes prototype-to-prototype values systematically higher. As Figure 3 shows, the cosine similarity between the target category DDCF and the other 9 categories lies between 0.94 and 0.98, with a mean of 0.956. This indicates that the 10 categories are extremely hard to discriminate and that the discriminative evidence lies only in weak texture detail.

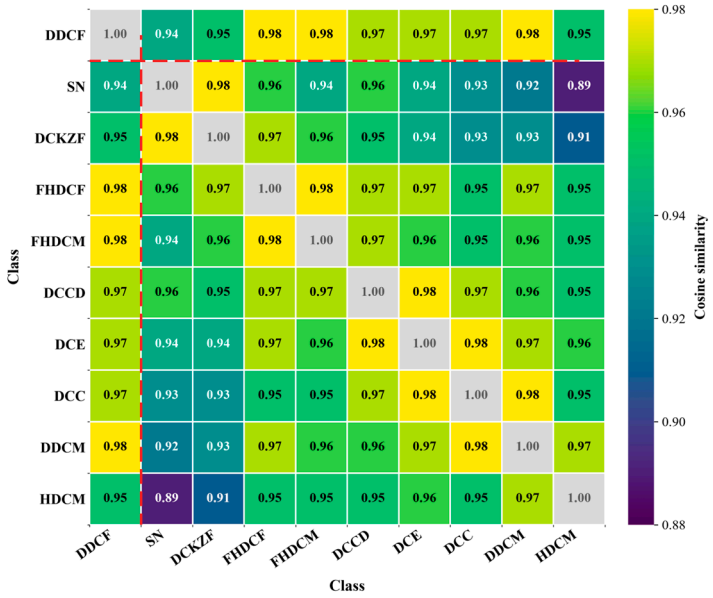


Figure 3. Cosine similarity of the refueling interface dataset.

3.2. Overall Architecture

A refueling robot works in unstructured scenes, and high-accuracy and real-time detection of the refueling interface are the prerequisites for a refueling robot to refuel

autonomously. For this reason, and compared with the earlier versions YOLOv8 [42], YOLOv10 [43], YOLO11 [44] and YOLO12 [45], YOLO26 [46] adopts an NMS-free architecture and offers advantages in accuracy and real-time performance on robot platforms that carry edge computing units, such as a refueling robot. We therefore take YOLO26 as the baseline and improve and optimize it, with the aim of raising the detection accuracy and robustness of the refueling interface in unstructured environments. The improved network is shown in Figure 4; the backbone and the detection head are kept identical to those of YOLO26, and the modifications are made in the cross-scale feature fusion of the neck and in the loss function.

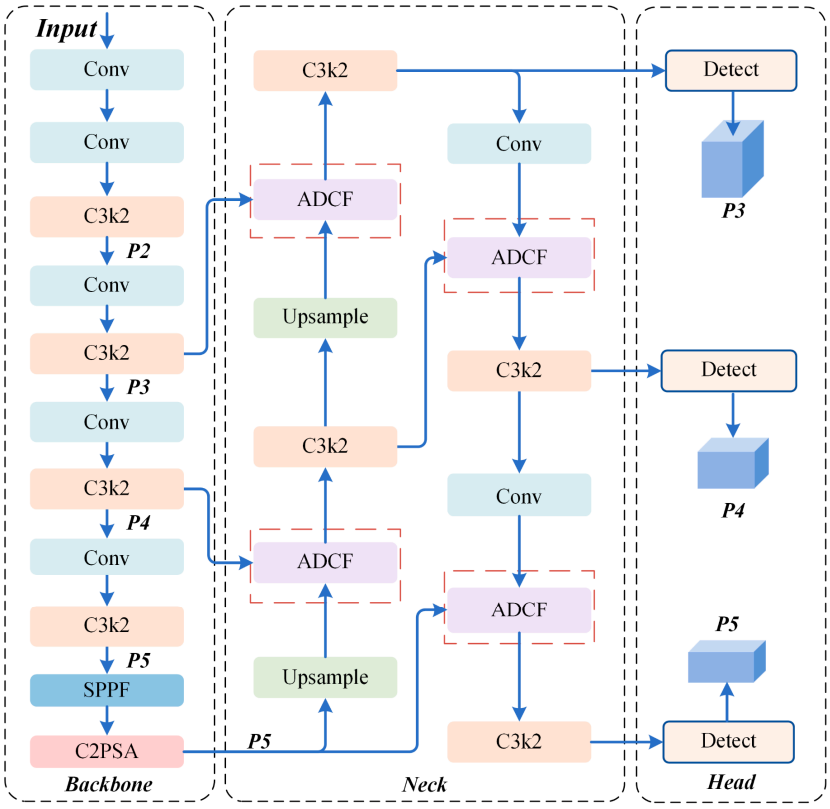


Figure 4. Network architecture of ADCF-YOLO.

The backbone extracts features. It is composed mainly of Conv and C3k2 and ends with SPPF and C2PSA, and it outputs the P3, P4 and P5 feature maps at strides of 8, 16 and 32, which feed the three decoupled detection heads. The neck is a bidirectional FPN–PAN feature pyramid. The FPN passes strong semantic deep features to the shallow levels by nearest-neighbor upsampling from top to bottom, and the PAN returns well-localized shallow features to the deep levels by stride-2 convolution from bottom to top. The neck contains four Concat operators, two in the FPN and two in the PAN. As shown in Figure 4, all Concat operators of the neck are replaced by ADCF, and after the replacement, the output channel number at each position still equals the sum of the channel numbers of the two inputs.

3.3. Adaptive Deformable Cross-Scale Fusion

Feature fusion in the neck of YOLO26 uses Concat to join the strong semantic features of the deep levels with the fine-grained features of the shallow levels directly along the channel dimension. The advantage of Concat is that it adds no parameters, but it does not distinguish the contributions of the different paths. On the refueling interface dataset, the use of Concat has two evident limitations. First, the two feature maps come from different resampling

paths, and both upsampling and convolution introduce a spatial displacement that depends on the object position. Concatenation has no ability to correct positions, however, so the spatial correspondence of the weak-texture structures in the shallow features is weakened. At the same time, the strong semantic responses of the deep levels and the detail responses of the shallow levels do not match in numerical scale, and Concat leaves one of the two paths systematically suppressed by the subsequent convolution. We therefore introduce ADCF and replace all the Concat operators of the neck with it. The module is inspired by the misalignment diagnosis of [47] but differs in that it regresses the offset explicitly and aligns before fusion, without a uniform down-weighting. As Figure 5 shows, the idea of ADCF is to insert two preceding operators before the two feature maps are concatenated. The spatial misalignment of the aligned-path features is corrected first; the detail responses of the enhancement path are then compensated, and a global gating mechanism decides the fusion weight of the two paths. Finally, the result is added to the original features in residual form (shortcut). This design not only improves the feature fusion capability but also allows the improved model to make full use of the pretrained weights of the baseline.

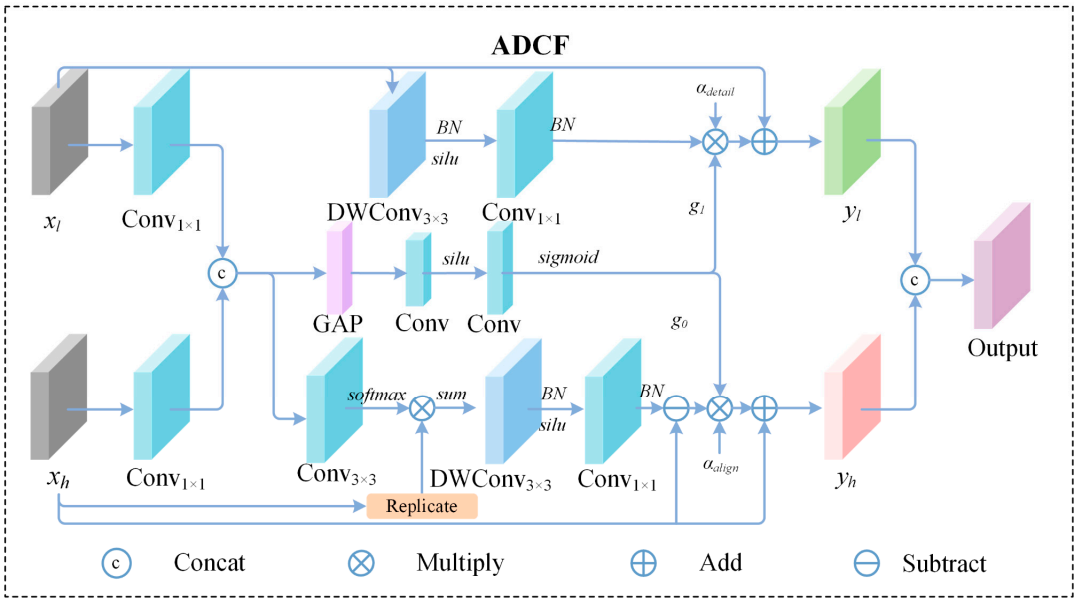


Figure 5. Structure of the ADCF module.

The ADCF module receives the two input features $x_h \in R^{C_h \times H \times W}$ and $x_l \in R^{C_l \times H \times W}$, where x_h is the upstream feature on the fusion path (from upsampling in the FPN node, from convolution in the PAN node), and xl is the skip feature of the same level. The two inputs are first mapped into a shared space:

$$f = [conv_{1\times1}(x_h); conv_{1\times1}(x_l)] \tag{1}$$

Here, Conv is the 1 × 1 convolution, and the number of output channels is set to 64. $[.;.]$ denotes concatenation along the channel dimension. f serves as the input to the subsequent alignment branch and gating branch.

The alignment branch does not regress a spatial offset directly; for every spatial position (i, j) of the feature map, it predicts a discrete distribution over a 3 × 3 neighborhood center at that position:

$$W = Softmax(Conv_{3\times3}(f)), W \in R^{9\times H\times W} \tag{2}$$

Here, Conv is the 3 × 3 convolution, the number of output channels is 9, and the weights and bias are set to 0 at initialization. Softmax normalizes along the channel dimension, so that $\sum_{k=1}^{9} W_{k,i,j} = 1$. The nine neighbors of every position are then taken from x_h, with

replicate padding at the boundary. Replicate padding keeps the statistics of the boundary features continuous and avoids the boundary artifacts that zero padding would introduce.

$$N_k(i,j;x_h) = x_h(i+\Delta y_k, j+\Delta x_k), (\Delta y_k, \Delta x) \in \{-1,0,1\}^2 \tag{3}$$

Because W is a probability distribution, Equation (4) is a convex combination of the neighborhood samples, so the output necessarily lies inside the convex hull spanned by the nine neighbors and cannot be sampled outside the neighborhood as an unconstrained offset would.

$$\tilde{x}_h(i,j) = \sum_{k=1}^{9} W_{k,i,j} N_k(i,j) \tag{4}$$

A depthwise separable convolution then extracts the features:

$$\begin{aligned} &\overline{x}_h(i,j) = \varphi_a(\tilde{x}_h) \\ &\varphi_a = DW_{3\times3}(\mathrm{g} = C_h) \to BN \to SiLU \to Conv_{1\times1} \to BN \end{aligned} \tag{5}$$

The task of the enhancement branch is fine-grained compensation. We use a set of depthwise separable convolutions identical to Equation (5) to extract its detail residual:

$$\begin{aligned} &d_l = \varphi_d(x_l) \in \\ &\varphi_d = DW_{3\times3}(g = C_l) \to BN \to SiLU \to Conv_{1\times1} \to BN \end{aligned} \tag{6}$$

Here, φ_d acts at the tensor level (not per position) and consists of one depthwise block followed by one pointwise block; its input and output both have C_l channels with unchanged spatial size so that $\mathrm{d_l}$ can be added to x_l directly; in the depthwise convolution, the number of groups equals the input channel count C_l, padding is 1 with zero values, and both convolutions are unbiased and followed by batch normalization.

The fusion weights of the two paths are produced from f by global pooling and then by a Sigmoid:

$$\begin{aligned} &(g_0, g_1) = Sigmoid(\phi_g(CAP(f))), (g_0, g_1) \in (0,1)^2 \\ &\phi_g = Conv_{1\times1}(2h \to h) \to SiLU \to Conv_{1\times1}(h \to 2) \end{aligned} \tag{7}$$

Here, f is globally pooled to give an output with $2h$ channels. A 1×1 convolution then reduces the number of channels to h, and SiLU adds a non-linear response. Finally, a 1×1 convolution reduces the number of channels from h to 2 and outputs g_0 and g_1. The gating mechanism uses global pooling rather than a pointwise prediction for the following reason: pointwise prediction can indeed control differences at the pixel level, but it raises the number of parameters and the memory cost markedly and does not suit deployment on a refueling robot. Global pooling reduces the computation from $O(HW)$ to $O(1)$ and adds almost no parameters, and the residual strength it produces is computed from the whole image, which avoids fluctuation of the gating coefficients caused by local noise.

The output of ADCF is formed by a residual connection, in order to reuse the original features and to strengthen the gradient flow.

$$y_h = x_h + \alpha_a \cdot g_0 \cdot (\overline{x}_h - x_h) \tag{8}$$

$$y_l = x_l + \alpha_d \cdot g_1 \cdot d_l \tag{9}$$

$$y = [y_h; y_l] \in R^{(C_h+C_l)\times H\times W} \tag{10}$$

In Equations (8) and (9), α_a and α_d are learnable scalars. Because $\alpha_a = \alpha_d = 0$ at initialization, the residual terms of Equations (8) and (9) are zero, so ADCF is equivalent to Concat at the start of training.

$$y|_{\alpha=0} = [x_h; x_l] \tag{11}$$

Because $\alpha_a = \alpha_d = 0$ at initialization, the residual terms in Equations (8) and (9) vanish and the module is functionally identical to Concat at epoch 0, so training starts exactly from the baseline solution. This is an identity, not a performance guarantee. It removes the risk of starting from a worse point, but it does not imply that the converged model is at least as accurate as the baseline.

3.4. Sub-Pixel Phase-Stable Detection

Downsampling in a convolutional network is discrete grid sampling. When an object undergoes a sub-pixel translation in the image, the combination of pixels that fall into the same convolution window changes accordingly, and the feature response jumps discontinuously [27]. The refueling interface dataset has a high proportion of weak texture objects, and a displacement of a single stride is enough to make the response peak jump back and forth between adjacent positions, which appears macroscopically as localization jitter of the same object between adjacent frames. Conventional random translation changes the position of the bounding box at the same time, so what the model learns is still the response in the translated coordinate system, and no consistency constraint can be imposed on the predictions before and after the translation. We therefore introduce SPSD in the training stage. The idea of SPSD is to construct a phase-shifted view and to compute the loss pixel by pixel after transforming the predictions of the two views into the same coordinate system. The design of SPSD is inspired by consistency regularization [30] but differs in that the shift is known and invertible, so the invariance need not be approached through perturbation approximation.

During training, the input image is shifted by a random sub-pixel translation, giving a shifted view. The purpose is to generate a small spatial perturbation and to increase the ability of the model to learn from position offsets. The original image and the shifted view are then propagated forward separately, giving the teacher prediction T (used only for supervision; its parameters are not updated) and the student prediction S.

T and S are produced by the same parameters θ within the same training step. The teacher branch is detached, so gradients are propagated only through S. This differs from Mean Teacher [30], whose teacher carries a memory of earlier steps through weight averaging: here, the target is the prediction of the identical network on the unshifted view at the same step, and the supervision correspondence is given analytically by the known shift rather than approximated by perturbation matching.

Because the shift is known, an affine transformation is used to transform the student prediction S into the coordinate system of the teacher prediction T:

$$\theta = \begin{bmatrix} 1 & 0 & \theta_x \\ 0 & 1 & \theta_y \end{bmatrix} \quad \theta_x = \frac{2(\delta_x/s_l)}{G_W - 1}, \theta_y = \frac{2(\delta_y/s_l)}{G_h - 1} \tag{12}$$

$$\mathrm{G}_\delta = A(\theta, \mathrm{G}_W, \mathrm{G}_H) \tag{13}$$

$$S' = GridSample(S, G_\delta) \tag{14}$$

Here, s_l is the downsampling stride of layer l, G_w, and G_h are the width and height of the feature map of that layer, and δx and δy are the horizontal and vertical components of the sub-pixel shift. G_δ is the sampling grid generated from θ, and GridSample denotes bilinear grid sampling. After the transformation of Equation (14), the student prediction S is in the same coordinate system as T, and the loss can be computed pixel by pixel. This design forms a sample pair: once the image is shifted by a sub-pixel translation, the prediction is shifted and aligned accordingly. The correction is applied to the dense prediction maps rather than to decoded boxes. For level l with downsampling stride s_l, an affine transform with translation components $\theta_x = 2\,(\delta_x/\mathrm{s_l})/(G_\mathrm{w} - 1)$ and $\delta_y = 2\,(\delta_y/\mathrm{s_l}))/(G_h - 1)$ resamples both the classification score map and the regression map of the shifted view back into the coordinate system of the unshifted view, after which the two maps are compared pixel by pixel. The ground-truth boxes are not shifted, because the supervision is defined in the original coordinate system; the shift is known, so the correspondence requires no label matching.

The teacher prediction in the shifted view may itself contain noise, and imposing a consistency constraint on the whole image would treat that noise as a label. We therefore select the supervised positions with two conditions, namely a ground-truth box of normalized area smaller than 0.01 and a teacher confidence above the threshold τ_s. As Equation (15) shows, an indicator function performs the region selection, and the teacher confidence is used as a pixel weight rather than as a simple threshold so that the supervision strength varies with the credibility of the teacher prediction.

$$w(i,j) = 1|(i,j) \in M_{small}| \cdot 1|\max_c p_{T,c}(i,j) \geq \tau_s| \cdot \max_c p_{T,c}(i,j) \quad (15)$$

Here, *(i, j)* is the row–column index on the prediction grid G_l (stride s_l) on which SPSD is supervised. $1[A]$ is the indicator function, taking the value 1 when the condition A holds and 0 otherwise. M_{small} is the small-object mask, defined as all positions on G_l that fall inside ground-truth boxes of normalized area ≤ 0.01—the ground-truth boxes being mapped from image coordinates to G_l by downsampling with b/s_l. $p_{T,c}$ (i,j) is the predicted probability of class c at position (i, j) in the teacher head.

The consistency of the classification and the regression branch is computed with Smooth L1. The classification loss first maps the raw classification outputs of the student and the teacher into probability space through a Sigmoid function and then computes the smooth L1 error between them, as in Equation (16). The bounding-box regression loss computes the Smooth L1 loss directly on the coordinate-aligned predictions, as in Equation (17). The two losses constrain the student network in the two dimensions of object confidence and spatial position, ensuring that the output of the model keeps spatial consistency when the image undergoes a sub-pixel translation.

$$l_{cls} = SmoothL1(\sigma(S'_{cls}, \sigma(T_{cls})) \quad (16)$$

$$l_{box} = SmoothL1(S'_{box}, T_{box}) \quad (17)$$

The SPSD loss is the weighted normalized result:

$$l_{SPSD} = \frac{\sum_{i,j} w(i,j)[l_{cls}(i,j) + \lambda_b l_{box}(i,j)]}{\sum_{i,j} w(i,j)} \quad (18)$$

l_{SPSD} is added to the loss with a linear ramp:

$$l = l_{yolo26} + \lambda_s \cdot r(e) \cdot l_{spsd}, r(e) = min(\frac{e+1}{E_w}, 1) \quad (19)$$

Here, l_{yolo26} is the loss function of YOLO26, λ_s is the weight factor, $r(e)$ is the ramp factor, e is the current training epoch, and E_w is the total number of epochs of the ramp stage. During training, $r(e)$ increases with the number of epochs. When the total number of epochs is reached, $r(e)$ takes the value 1. The purpose of increasing the weight of l_{SPSD} gradually is to let the network learn the object features first and translation invariance afterward.

3.5. Counterfactual Context Decoupling

When the object's own features are weak and the background features are strong, the model relies on the background features as the discriminative evidence instead of using the object's own features [37]. In the refueling interface dataset, the mean cosine similarity over all 45 class pairs is 0.956, and 0.97 between the target class DDCF and the other nine, and the model very easily takes the background features of the object as the discriminative evidence. This leads to a high accuracy during training but to a large drop in accuracy when the application scenario changes. We therefore introduce CCD to solve this problem. The design is inspired by the work of Li et al. [48] but differs in that it constructs a bidirectional

constraint at the view level and needs no counterfactual synthesis at the feature level. The idea of CCD is to construct counterfactual views during training, requiring the model to discriminate the object correctly after its background has been replaced, and at the same time requiring the model not to respond after the object has been erased. CCD uses two oppositely directed supervision signals, which force the network to make its decisions only by learning the object's own features and not the background features.

Two intervention images are constructed from the input image during training. The first is the background replacement view, shown in Figure 6b. For the bounding box of each object, the difference between a ρ_e-fold expanded box and the original box forms an annulus region, shown as the area between the yellow dashed box and the red dashed box in Figure 6a. The images of the batch are cyclically shifted to produce a misaligned view, and the corresponding pixels are taken from it to fill the annulus region so that the background is replaced while the object is kept unchanged and the co-occurrence shortcut between object and background is cut. The reason for using annulus replacement rather than whole-image replacement is to make the intervention variable unique. If the whole image were replaced, the object itself would change as well, and the training supervision would lose its reference. The second is the target erasure view, shown in Figure 6c. A ρ_e-fold expanded region is taken around the object bounding box, and the object region is filled with the channel mean of the image itself, so that only the background is kept. Each erased region takes the per-image per-channel mean of the input instead of black or white, so no artificial feature that appears in neither training nor inference is introduced. Being derived from the image itself, the fill changes from image to image and cannot serve as a memorized no-object token. Black would, in addition, coincide with the fill of the default random erasing ($p = 0.4$) that every run receives, and noise would supply high-frequency content that acts as a cue of its own, narrowing the intended constraint to no response to noise. This view provides counterfactual supervision for the absence of the object and constrains the model not to respond when only the background is present, which effectively removes the dependence of the model on the object background.



(**a**) Original (**b**) Context replacement (**c**) Target erasure

Figure 6. Counterfactual views. (**a**) Original; (**b**) Context replacement; (**c**) Target erasure.

Two intervention images are constructed from the input image during training. Let $\rho \cdot B$ denote the center-expanded ground-truth box of B, 1_Ω denote the indicator function of a region, and $\otimes$ denote element-wise multiplication, with 1_Ω broadcast over the channels. The context-replacement view intervenes only on the annulus around the object, $Ar = \rho_r B \backslash B$, whereas the target-erasure view degrades $\Omega_e = \rho_e B$ to an uninformative background. Let x^{roll} be the donor image obtained by cyclically shifting the same batch by one position, and let $\mu(x)$ be the per-channel mean of that image itself. The two views are then constructed as:

$$x_{ctx} = x \otimes (1 + 1_{A_r}) + x^{rolll} \otimes 1_{A_r} \tag{20}$$

$$x_{bg} = x \otimes (1 - 1_{\Omega_e}) + \mu(x) \otimes 1_{\Omega_e} \tag{21}$$

The first keeps the object and replaces its surrounding context; the second removes the object and keeps the context. When an image contains several objects, the regions are united over their boxes. Both views are forwarded in the same 2B batch and share all parameters with the other passes.

The consistency loss of the background replacement view is:

$$l_{ctx} = \frac{\sum_{i,j} w'(i,j)[SmoothL1(\sigma(P_{ctx})) + \lambda'_b SmoothL1(B_{ctx}, B_T)]}{\sum_{i,j} w'(i,j)} \quad (22)$$

Here, P_{ctx} and B_{ctx} are the classification and box outputs of the annulus context-replacement view. T_{cls} and T_{box} are the corresponding outputs of the detached teacher head on the original view (Section 3.4). σ is the Sigmoid. λ_b' is the weight of the box term. $w'(i,j)$ has the same form as in Equation (15), except that the region mask is relaxed to the 1.0-fold object box and the confidence threshold is lowered to τ_c so that more objects are covered.

The erased region has no negative-sample annotation, so it is neither background nor any known category. The response of the target erasure view on the channel of the target category should therefore be reduced. We therefore apply a one-sided penalty with the Softplus function. When the model wrongly detects the target category in the erased region, the value of the loss function is very large, which drives the model to change its learning direction.

$$l_{bg} = \frac{1}{|\Omega_e||C|} \sum_{(i,j)\in\Omega_e} \sum_{c\in C} ln(1 + exp(z_{bg,c}(i,j))) \quad (23)$$

Here, Ω_e is the target erasure region (ρ_e times the outer spread frame), C is the set of channels corresponding to the truth category inside the frame, and $Z_{bg,c}(i,j)$ is the logit of the erasure view on the c-th channel at that position. Softplus tends to zero when the logit tends to negative infinity, which avoids producing a gradient in that case.

The loss function of CCD is:

$$l_{CCD} = \lambda_{ctx} \cdot l_{ctx} + \lambda_{bg} \cdot l_{bg} \quad (24)$$

Here, l_{ctx} and l_{bg} are given by Equations (22) and (23); λ_{ctx} and λ_{bg} are the weights of the two terms.

The total loss function of ADCF-YOLO is:

$$L = l_{yolo26} + \lambda_s r(e) l_{SPSD} + \lambda_c r(e) l_{CCD} \quad (25)$$

Here, l_{yolo26} is the detection loss of the baseline YOLO26—classification, regression and assignment—which are left unchanged. λs and λc are the overall weights of the two auxiliary losses. $r(e) = \min((e+1)/Ew, 1)$ is the linear ramp, where e is the current training epoch, and Ew is the number of ramp epochs.

The three improvements work together within one framework. ADCF improves the way features are fused and solves the problems of spatial misalignment and feature enhancement at the same time. SPSD and CCD do not change the network structure but only construct the corresponding views during training, and they take no part in the computation at inference.

To show how SPSD and CCD are coupled to the training of the backbone network, Figure 7 gives the complete training framework of ADCF-YOLO. Each iteration contains four forward passes that share the parameters f_θ. The original view produces the teacher prediction T—the teacher is one forward pass of the student's own weights with detached gradients; it introduces no extra parameters and is never updated by the optimization, unlike the exponential moving average teacher. The phase-shifted view produces the student prediction S, which is re-aligned to the coordinate frame of T by bilinear grid

sampling with the known shift, after which the SPSD consistency loss is applied. On the annulus background replacement view, the context-invariance term l_{ctx} against T is computed with the teacher confidence as the weight (Equation (22)). On the target-erasure view, the one-sided Softplus suppression l_{bg} is imposed on the erased region (Equation (23)). The two losses are combined into l_{CCD} by Equation (24), and the total loss is weighted by Equation (25).

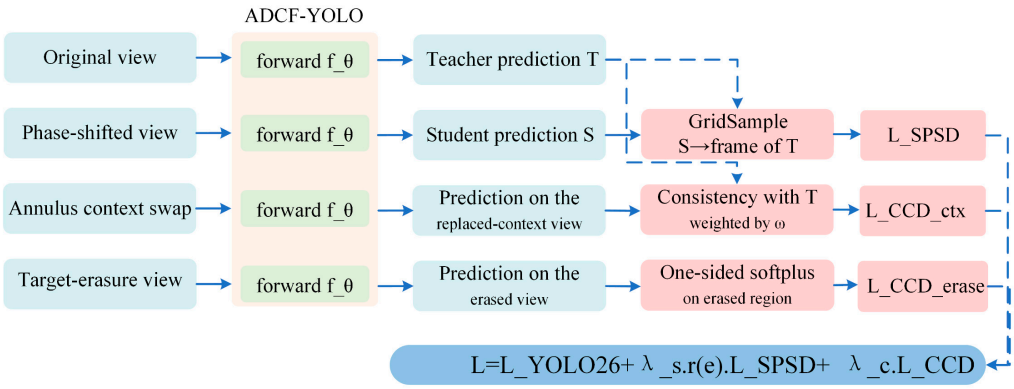


Figure 7. Training framework of ADCF-YOLO.

4. Experimental Results and Discussion

4.1. Experimental Settings

4.1.1. Datasets

The Industrial mechanical parts dataset [49] was constructed by Mateusz Wojtalik and is distributed on the Ultralytics Platform. It is used for the detection of mechanical parts in an industrial setting. The dataset was collected from the web and contains 9625 images and 38,157 bounding boxes. It has 8 categories, namely bearing, bolt, flange, gear, nut, retaining_ring, spring and washer. It contains 9625 annotated images and 38,157 instance annotations over eight categories, distributed with an official split of 7325/1925/375 images for training, validation and test (29,221/7518/1418 boxes). All the categories are made of metal, are symmetrical in appearance and have little texture. It is therefore used to verify the robustness of ADCF-YOLO on a different task in the same domain.

The VisDrone-DET dataset [50] is used for object detection from a drone viewpoint. It contains 10 categories, 8629 images and more than 2.6 million bounding boxes. Although VisDrone-DET and the refueling interface dataset differ in application scenario, the objects of VisDrone-DET are far smaller: 25.1% of the boxes in the refueling interface dataset are small by the COCO definition, against the overwhelming majority in VisDrone-DET. VisDrone-DET is therefore used to test how the method behaves when the object scale falls well below the range it was designed for.

4.1.2. Experimental Parameter Settings

All experiments in this paper use a unified hardware and software environment: the hardware is a single NVIDIA GeForce RTX 3090 (24 GB), and the software environment is Python 3.11.15, PyTorch 2.8.0 and CUDA 12.9. For the comparison experiments, the official source code of every model was retrained on the same split. The hyperparameters that control optimization—optimizer, learning-rate schedule, 100 epochs, batch size 16, and input size 640—are identical for all models and are listed in Table 1. The data-augmentation pipeline is the one shipped as the default with each official implementation and was not tuned per model; it is the only setting that differs between models, and it was left at its default precisely to avoid tuning the baselines.

Table 1. Hyperparameter settings.

Hyperparameter	Value	Hyperparameter	Value
Image size	640 × 640	h	64
Batch size	16	k	3
Epochs	100	α_a	0
Workers	8	α_d	0
Lr1	0.01	τ_s	0.20
Momentum	0.937	τ_c	0.05
λ_s	0.15	E_w	3
λ_c	0.10	ρ_r	1.6
δ_{max}	2.0	ρ_e	1.05

The hyperparameter settings are listed in Table 1. The hyperparameters fall into three groups with different statuses. A first group is fixed by design and not searched ($k = 3$, $h = 64$, $\alpha_a = \alpha_d = 0$, image size 640, batch 16, 100 epochs). The equivalent displacement centroid of a 3 × 3 convex combination is bounded by one feature point per axis, against a measured inter-path misalignment of 1.09 points at the FPN nodes and at most 0.22 at the PAN nodes, so k = 3 covers the measured range up to a residual of 0.09 points at the FPN nodes, which the gate can down-weight rather than correct. A second group follows from the data. With $\rho_r = 1.6$, the intervention occupies 2.01% of the image per object and 16.9% of the annulus is shared with neighbors, while 2.0 would push contamination above 20%; $\rho_e = 1.05$ leaves the 1.2 px margin that integer rounding requires on the mean object size of 46.9 × 48.0 px. $\tau_s = 0.20$ keeps 75.5% of teacher-detected objects at 97.6% class accuracy, and the trade-off is near-linear, so τ_s is not sensitive. The last group ($\lambda_s = 0.15$, $\lambda_c = 0.10$, $E_w = 3$, $\tau_c = 0.05$ and the sub-weights) was set empirically.

4.1.3. Evaluation Metrics

To evaluate the detection performance of the models, four metrics were selected from three perspectives: localization accuracy, category discrimination and overall statistics. Precision and recall evaluate the detection results from the aspects of trustworthy prediction and actual retrieval, respectively, and are computed as in Equations (26) and (27). AP is defined as the area under the precision–recall curve, as in Equation (28). mAP averages the AP of each category by Equation (29) and serves as the metric of the overall cross-category performance of a model. In view of the characteristics of the refueling interface dataset, we focus on the mAP@0.5 and mAP@0.5:0.95 metrics. From the deployment point of view, we use FPS (end-to-end), GFLOPs, Params and Model Size to measure the deployment cost of a model.

$$Precision = \frac{TP}{TP + FP} \tag{26}$$

$$Recall = \frac{TP}{TP + FN} \tag{27}$$

$$AP = \int_0^1 P(r)dr \tag{28}$$

$$mAP = \frac{1}{N_{class}} \sum_{c=1}^{N_{class}} AP_c \tag{29}$$

4.2. Comparison with Mainstream Detectors

To fully evaluate the effectiveness of ADCF-YOLO, YOLOv8, YOLOv10, YOLO11, YOLO12 and YOLO26 were chosen as the comparison models. The n variant is used for every model to keep speed and efficiency balanced. Its parameter counts range from 2.27 M to 3.01 M, the same order of magnitude as the 2.76 M of ADCF-YOLO. The comparison is thus

confined to differences in algorithm design at an equal scale, avoiding the confusion that would be introduced by different model sizes. In addition to the YOLO series, Transformer-based and fine-grained detection methods were also selected for cross-analogy pairs, and the models are RF-DETR [51], RT-DETR-L [52] and CSDETR [53]. All models share the same 100-epoch budget and split, with official pre-trained initializations throughout except CSD-DETR, whose implementation does not support them.

4.2.1. Quantitative Analysis

Table 2 gives the quantitative comparison of the nine models on the refueling interface dataset. Values are the mean ± standard deviation over three independent training runs that share the split and the hyperparameters and differ only in the random seed.

Table 2. Experimental results of ADCF-YOLO and the mainstream detectors on the refueling interface dataset. The experimental results are from the test set.

Baseline	Precision	Recall	mAP@0.5	mAP@0.5:0.95	FPS	GFLOPs	Params (M)	Model Size (MB)	Latency (ms)
YOLOv8	0.920	0.906	0.913 ± 0.031	0.462 ± 0.028	335	8.1	3.008	6.3	86.3
YOLOv10	0.914	0.921	0.918 ± 0.058	0.467 ± 0.067	303	6.5	2.267	5.8	91.2
YOLO11	0.918	0.916	0.927 ± 0.019	0.475 ± 0.024	337	6.3	2.584	5.5	93.3
YOLO12	0.901	0.930	0.924 ± 0.058	0.448 ± 0.063	262	6.3	2.558	5.6	116.1
YOLO26	0.923	0.932	0.934 ± 0.028	0.482 ± 0.041	349	5.2	2.376	5.4	110
ADCF-YOLO	0.943	0.955	0.959 ± 0.023	0.526 ± 0.027	287	6.7	2.763	6.2	190
RF-DETR	0.961	0.959	0.961 ± 0.040	0.533 ± 0.025	164	48.1	31.89	127.87	243
RT-DETR-L	0.948	0.957	0.958 ± 0.028	0.532 ± 0.037	72	109.9	32.83	66.26	282
CSDETR	0.892	0.875	0.916 ± 0.062	0.471 ± 0.091	89	18.12	17.33	36.63	127

The precision and recall of ADCF-YOLO are 0.943 and 0.955, which are 0.020 and 0.023 above those of YOLO26n and are the highest within the YOLO family, so the gain covers detection rate as well as localization quality. mAP@0.5 reaches 0.959, an improvement of 2.68% over YOLO26n and 3.45% over YOLO11 (0.927), the second-best YOLO-family model. mAP@0.5:0.95 rises from 0.482 to 0.526, a relative gain of 9.13% over YOLO26n and 10.74% over the second-best YOLO-family result, YOLO11 (0.475). The three Transformer-based and hybrid detectors occupy a different cost level. RF-DETR records the highest precision (0.961), recall (0.959), mAP@0.5 (0.961) and mAP@0.5:0.95 (0.533); RT-DETR-L is 0.001 below ADCF-YOLO on mAP@0.5 and 0.006 above it on mAP@0.5:0.95; CSDETR is 0.043 and 0.055 below. Although the accuracy of the DETR family is higher, its real-time performance does not suit refueling robots. The margin over ADCF-YOLO is only 0.002–0.007 on the two mAP metrics, and it is obtained at 7.2–16.4 times the GFLOPs and 11.5–11.9 times the parameters, a compute budget that the edge platforms of a refueling robot cannot accommodate, while the throughput measured on the same GPU falls to 164 and 72 FPS against 287 FPS. What deserves attention is the difference in dispersion among the models on mAP@0.5 and mAP@0.5:0.95. On mAP@0.5, the five YOLO-family comparison models span 0.913–0.934, a range of only 0.021. On mAP@0.5:0.95, however, the span widens to 0.448–0.482, with a range of 0.034. ADCF-YOLO alone extends the two ranges to 0.046 and 0.078: the gap it opens over the family is 0.025 on the loose metric but 0.044 on the strict one, so the differences among models concentrate at the high IoU thresholds. Across all nine detectors, the two ranges are 0.048 and 0.085. The reason for this phenomenon is that the 10 categories are highly similar in appearance, and the models readily fall into spatial misalignment and background dependence when discriminating between them. Such deviations are penalized weakly at low IoU thresholds and penalized more heavily at high IoU thresholds. The three improvements of ADCF-YOLO focus on

spatial misalignment, aliasing and background dependence. The gain on mAP@0.5:0.95 is therefore more pronounced, which agrees with the direction expected from the design of the method.

In terms of efficiency, ADCF-YOLO adds 0.387 M parameters, its GFLOPs increase from 5.2 to 6.7, and its FPS decreases from 349 to 287. The change in the efficiency figures comes mainly from the depthwise separable convolution and the gating branch of the ADCF module, whereas SPSD and CCD take no part in the computation at inference time. Although the efficiency metrics of ADCF-YOLO are lower, the trade-off is worthwhile, given the increase in accuracy. This is because a high-accuracy detection result is more valuable for a refueling robot in an engineering application.

Deployment feasibility. ADCF-YOLO requires 2.76 M parameters, 6.7 GFLOPs, and 6.2 MB of storage. During docking, the refueling interface is quasi-static, and the manipulator advances at centimeter-per-second velocities, so a detection rate of 5–10 Hz suffices for closed-loop visual guidance. Measured end-to-end on a CPU-only laptop platform (Intel Core i7-10750H), ADCF-YOLO achieves approximately 5.3 FPS and the YOLO26n baseline approximately 9.1 FPS, indicating that the task requirement is met even without hardware acceleration; edge modules with integrated accelerators, such as the Jetson Orin family, provide further headroom. Benchmarking on such hardware is left as future work.

Since the two auxiliary views of SPSD and CCD are synthesized at every training step, the training cost of ADCF-YOLO is higher than that of YOLO26n. The comparison is performed under the hyperparameter settings of Table 1. YOLO26n takes 675.53 s in total, or 6.76 s per epoch. Adding SPSD raises the total to 1461.63 s (14.62 s per epoch, 2.16× the baseline), adding CCD to 2289.12 s, and the complete ADCF-YOLO to 3564.91 s. The growth tracks the number of passes paid per step: 1.08, 1.13 and 1.32 times the corresponding pass ratio of two, three and four views. Four views are evaluated in each iteration—the original image, the SPSD-shifted view, and the CCD context-replaced and erased views—but only the original view is detached from the graph: it supplies the teacher target T and updates no parameter. The other three are student passes, and all of them do carry gradients, through l_{SPSD}, l_{ctx} and l_{bg}, respectively. The four terms are summed into the single scalar of Equation (25), so one optimizer step is taken per iteration, as in the baseline; what grows is the number of forward passes and the fact that the backward pass now covers three views.

4.2.2. Class-Level Detection Performance Analysis

To verify whether the performance gain is evenly distributed across the categories, Table 3 reports the experimental results of the 10 categories of the refueling interface dataset on YOLO26 and ADCF-YOLO in terms of mAP@0.5 and mAP@0.5:0.95.

It can be seen from the table that mAP@0.5:0.95 is improved for all 10 categories by 0.044 on average. The average per-category improvement of mAP@0.5 is 0.025. This shows that the overall gain is not pulled up by a few categories but follows from an improvement over all of them. To evaluate the performance of a single category at the two levels of detection and box accuracy, we define $g_c = mAP@0.5(c) - mAP@0.5:0.95(c)$. The average gap of g_c across the 10 categories of the baseline model is 0.452, and the largest gaps are those of SN and DCKZF, at 0.524 and 0.503. These categories are almost fully detected at the loose threshold, but their accuracy falls below 0.51 at the strict threshold. In ADCF-YOLO, by contrast, the average gap of the 10 categories is 0.433, 0.019 lower than that of the baseline; the gap narrows on four categories, by 0.021 to 0.132, and widens on the other six by no more than 0.023. This shows that the three improvements proposed in this paper are not only effective on a few categories; they also provide a gain at both thresholds for all 10 of them.

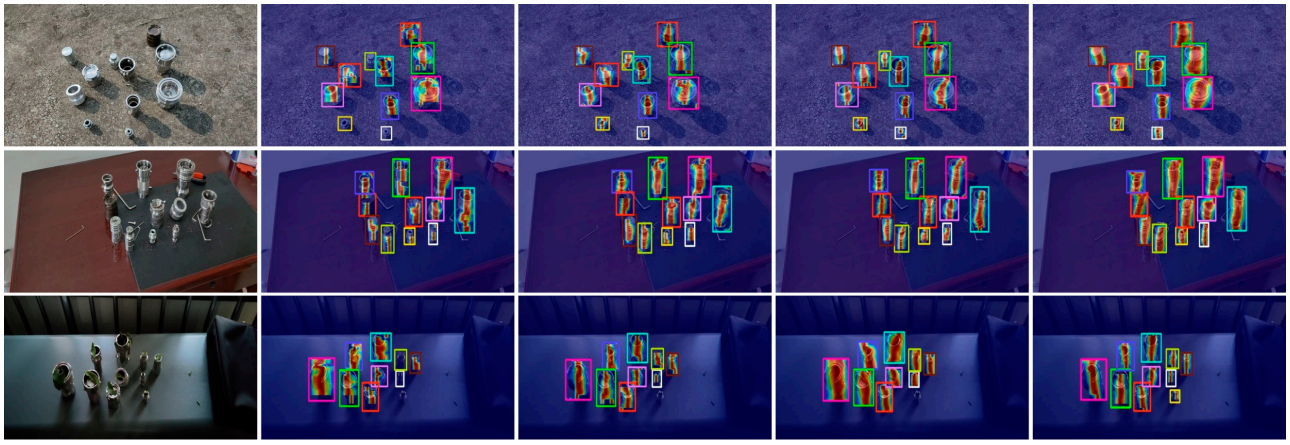
Table 3. mAP@0.5 and mAP@0.5:0.95 for each category on the refueling interface dataset.

Class	YOLO26		ADCF-YOLO	
	mAP@0.5	**mAP@0.5:0.95**	**mAP@0.5**	**mAP@0.5:0.95**
DDCF	**0.975**	**0.558**	**0.989**	**0.636**
SN	0.953	0.429	0.965	0.573
DCKZF	0.944	0.441	0.963	0.481
FHDCF	0.955	0.486	0.962	0.514
FHDCM	0.902	0.457	0.937	0.486
DCCD	0.948	0.516	0.970	0.535
DCE	0.936	0.447	0.966	0.473
DCC	0.908	0.514	0.949	0.532
DDCM	0.911	0.529	0.938	0.554
HDCM	0.903	0.442	0.951	0.473
All	0.934	0.482	0.959	0.526

Measured by relative improvement in mAP@0.5:0.95, the three categories with the largest gains are SN (+33.6%), DDCF (+14.0%) and DCKZF (+9.1%), and the relative gain is positive for all 10 categories, ranging from +3.5% to +33.6%. The gain is therefore not concentrated on a few categories, and its distribution matches the two mechanisms targeted by the three improvements. SN and DCKZF are the two categories with the largest baseline gaps g_c (0.524 and 0.503): they are almost fully detected at the loose threshold, but their boxes are poorly localized at the strict threshold. SN obtains the largest strict-IoU gain of the whole table (+0.144), and DCKZF improves by +0.040, which is consistent with ADCF reducing the spatial misalignment of the features and SPSD smoothing the box regression: the categories with the largest localization headroom gain the most. At the loose threshold, by contrast, the largest absolute gains occur in the coupling categories most similar to DDCF in appearance, HDCM (+0.048), DCC (+0.041) and FHDCM (+0.035). The evidence that discriminates these categories from DDCF lies in faint texture structures, and when the model cannot resolve these details, it turns to the co-occurrence of the object with its background as a shortcut. Once CCD cuts this channel by counterfactual erasure, the model is forced to learn the object's own features for discrimination, so the confusable coupling categories benefit most at the detection level. The target category DDCF achieves the best results on both mAP@0.5:0.95 and mAP@0.5. This is due not only to the fact that DDCF has more bounding boxes than the other categories, but also to the fact that ADCF-YOLO improves spatial misalignment and background dependence.

4.2.3. Visualization Analysis

Beyond the numerical comparison, we also look at where each model actually places its responses, using class-activation heatmaps (Figure 8). The backward target is formed by adding the class scores of the top 2% candidate detections above 0.2 confidence, and gradients are aggregated at the target layers into one map per image, min–max normalized and overlaid on the input with a jet color scale. We deliberately do not renormalize inside the boxes, which is what makes background leakage visible in the first place. The rectangles are each model's own predictions, produced at a 0.2 confidence threshold and 0.65 NMS IoU, with a per-class color assignment fixed across all four columns. Only the converged weights and one forward–backward pass are involved, with nothing from training.



(**a**) Input (**b**) YOLOV8 (**c**) YOLO11 (**d**) YOLO26 (**e**) ADCF-YOLO

Figure 8. Heatmap comparison of ADCF-YOLO and the other models. (**a**) Input; (**b**) YOLOv8; (**c**) YOLO11; (**d**) YOLO26; (**e**) ADCF-YOLO.

Column (a) is the input, and columns (b)–(e) give the response maps of YOLOv8, YOLO11, YOLO26 and the proposed detector, over three scenes—concrete outdoors, a dark indoor table, low light—with 1 to 10 objects per frame. There is only one criterion to apply: the objects occupy 5–15% of the image width (Figure 2d) and are surrounded by strong background structure, so a focused detector should place a compact high-intensity region inside its predicted box and leave the background at the low end of the scale.

The three baseline columns spread instead. Their response center does fall near the object, yet high-intensity pixels run along the boundary onto background texture, onto neighboring objects and onto equipment edges; the peak drifts away from the geometric center, and in the crowded indoor row the hot regions of adjacent objects join into a single blob. Our detector's response stays tight: the peak sits on the object center, most of the high-intensity area falls inside the predicted box, and both cross-object interference and background leakage drop sharply. The SN instance in the low-light row is a special case. None of the three baselines produces a detection above 0.2 there, and since a missed object contributes nothing to the backward target, that position shows neither a box nor a hot region.

On the mechanism, the observations above line up with the reasoning in Section 3.3, though this section does not treat them as causal. Stacking the two resampling paths with equal weights leaves the displacement from upsampling and stride-2 convolution uncorrected; the spatial correspondence among the thin structures in the shallow features—bayonet catches, pins, flange notches—gets washed out during fusion, and the head is left to assemble a blurred, low-resolution blob around the object from semantic responses alone. ADCF aligns the main path first, restores the detail carried by the auxiliary path, and then lets a global gate decide how much intervention is allowed; the multi-scale features arriving at the head are consequently more consistent in space, and the hot region returns to the object.

4.3. Ablation Experiments and Mechanism Analysis

4.3.1. Ablation Experiments

The ablation results on the refueling interface dataset are given in Table 4. Each of the eight configurations is a single training run, so we read the table against the run-to-run standard deviation of the strict metric in Table 2 (0.027–0.041) and interpret only differences that exceed this band. Introduced separately, ADCF raises mAP@0.5:0.95 from 0.482 to 0.512 and SPSD to 0.508, gains of 0.030 and 0.026 that sit at the level of one standard deviation,

while CCD alone reaches 0.483, i.e., no measurable strict-IoU effect, which is consistent with its role as a detection-level regularizer rather than a localization one. Among the pairs, ADCF with CCD reaches 0.515, and SPSD with CCD reaches 0.526, exceeding the baseline by 0.033 and 0.044; the ADCF–SPSD pair scores 0.475, within the noise band of the baseline, so we do not claim a strict-IoU degradation for it, although it yields the lowest recall of the eight configurations (0.897). The complete model returns 0.526 on the strict metric, the same value as SPSD with CCD, while adding 0.008 in mAP@0.5 and 0.047 in recall over that pair, with precision unchanged at 0.943. On the loose metric, the joint gain is 0.025, close to the sum of the single gains (0.023). What the eight combinations support is therefore narrower than uniform synergy: the strict-IoU improvement is carried by SPSD and CCD in combination, ADCF adds mainly on the loose metric and on recall, and the low recall of the ADCF–SPSD pair (0.897) is recovered once CCD is present (0.955).

Table 4. Ablation experiments on the refueling interface dataset.

Baseline	ADCF	SPSD	CCD	Precision	Recall	mAP@0.5	mAP@0.5:0.95
YOLO26n	×	×	×	0.923	0.932	0.934	0.482
YOLO26n	✓	×	×	0.937	0.916	0.945	0.512
YOLO26n	×	✓	×	0.924	0.918	0.939	0.508
YOLO26n	×	×	✓	0.915	0.949	0.941	0.483
YOLO26n	×	✓	✓	0.943	0.908	0.951	0.526
YOLO26n	✓	✓	×	0.906	0.897	0.931	0.475
YOLO26n	✓	×	✓	0.917	0.952	0.943	0.515
YOLO26n	✓	✓	✓	0.943	0.955	0.959	0.526

Table 5 gives the ablation results on the Industrial mechanical parts dataset. On this dataset, the baseline mAP@0.5:0.95 is 0.787, and it rises to 0.813, 0.792 and 0.815 when ADCF, SPSD and CCD are introduced separately, with the corresponding gains being 0.026, 0.005 and 0.028, so that CCD yields the largest single-module gain. The sum of the individual gains of the three improvements, 0.059, is higher than the joint gain of 0.031. The reason for this phenomenon may be that the baseline model already has a strong localization capability on this dataset, so that the room for improvement is limited. The three improvements therefore overlap in the bottleneck region, and their marginal returns cover one another. Each of the three improvements addresses a well-defined direction of optimization, and the size of its gain depends on how severe that failure mode is in the dataset. When spatial misalignment, phase aliasing and background dependence coexist in a dataset, the three are complementary. When one failure mode does not dominate, the gains of the improvements overlap, but none of them causes a negative deviation.

Table 5. Ablation experiments on the Industrial mechanical parts dataset.

Baseline	ADCF	SPSD	CCD	Precision	Recall	mAP@0.5	mAP@0.5:0.95
YOLO26n	×	×	×	0.959	0.881	0.935	0.787
YOLO26n	✓	×	×	0.959	0.914	0.952	0.813
YOLO26n	×	✓	×	0.937	0.886	0.939	0.792
YOLO26n	×	×	✓	0.968	0.893	0.948	0.815
YOLO26n	✓	✓	✓	0.973	0.908	0.964	0.818

Next, each proposed module was placed against existing designs of the same purpose, by replacing that module alone and keeping the training protocol of Section 4.1 unchanged. BiFPN [15], ASFF [16] and DySample [18] replace ADCF in the neck, BlurPool [27] replaces the SPSD penalty, and Cutout [38] replaces the CCD constraints. Table 6 reports the eight configurations. Replacing ADCF costs between 0.006 and 0.021 in mAP@0.5 and between

0.004 and 0.023 in mAP@0.5:0.95, with DySample the strongest of the three substitutes and BiFPN the weakest. The configuration with ADCF also holds the highest precision of the four, 0.937 against 0.902–0.928, while the three substitutes reach a higher recall by 0.010 to 0.025; those additional detections are not matched by a gain in either mAP metric, so the advantage of ADCF lies in classification precision and localization quality rather than in detection rate. BlurPool, in place of SPSD, is lower on all four columns: 0.933 and 0.498 in the two mAP measures against 0.939 and 0.508 for SPSD, precision 0.918 against 0.924, and recall 0.824 against 0.918, a drop of 0.094. This is consistent with the design argument of Section 2.2: BlurPool attains shift invariance through low-pass filtering, which suppresses the high-frequency components in which the weak-texture evidence of the ten categories lies, so stability is bought at the cost of missed detections. SPSD constrains the response without removing frequency content and dominates this comparison. The comparison with Cutout is mixed and is reported as it stands. Cutout is lower in precision by 0.012, in recall by 0.014 and in mAP@0.5 by 0.014, yet it reaches a higher mAP@0.5:0.95, 0.496 against 0.483. The gain of CCD over Cutout is therefore concentrated in detection completeness at loose IoU thresholds and does not extend to box tightness at strict thresholds.

Table 6. Comparison between the proposed model and similar methods.

YOLO26	Precision	Recall	mAP@0.5	mAP@0.5:0.95
+ADCF	0.937	0.916	0.945	0.512
+BiFPN	0.902	0.926	0.924	0.489
+ASFF	0.915	0.934	0.936	0.499
+DySample	0.928	0.941	0.939	0.508
+SPSD	0.924	0.918	0.939	0.508
+BlurPool	0.918	0.824	0.933	0.498
+CCD	0.915	0.949	0.941	0.483
+Cutout	0.903	0.935	0.927	0.496

4.3.2. Gating Distribution and Residual Scaling Mechanism of ADCF

The residual fusion proposed in Section 3.3 uses zero-initialized learnable scalars (α_{align}, α_{detail}), together with global gates (g_0, g_1) to adjust the fusion weights jointly. At the start of training, it is equivalent to Concat, and the expressive power of the residual branches is added gradually during training. To verify whether the training process follows this design, this section plots the training trajectories of (α_{align}, α_{detail}) and (g_0, g_1), as shown in Figure 9.

Figure 9a,b plot the curves of α_{align} and α_{detail} at the four ADCF nodes of the neck as a function of the training epoch. Both curves start from zero, which shows that at the beginning of training the operator is equivalent to Concat. In the early stage of training, both curves deviate monotonically from zero with the epoch, which shows that the residual branches are activated step by step. Otherwise, the loss curve in the early stage of training would show a jump inconsistent with the baseline, and the gains in the ablation experiments would be hard to distinguish from randomness. In Figure 9a, α_{align} is negative at Neck-P4 and Head-P4. This shows that when $\alpha < 0$, the residual term does not suppress the alignment result but extrapolates away from the current fusion direction, which is equivalent in effect to applying a high-pass enhancement to the features. Since $x_h - \overline{x}_h$ is the difference between a convex combination of the neighborhood and the position itself, it is exactly the high-frequency component of that position relative to the neighborhood average. A negative α_{align} therefore means that the network tends to enhance local detail rather than smooth alignment at that level, which agrees with the characteristic of the refueling interface dataset that the evidence for discrimination lies only in faint texture structures. This result also shows that leaving α unconstrained by an activation function in Equation (8) is a necessary design

choice. Constraining it to positive values would deprive the network of the expressive power in this direction. Figure 9c,d plot the distributions of g0 and g1 at the four ADCF nodes. It can be seen from the figure that the gate coefficients are not uniformly distributed over (0, 1) but diverge markedly with the level and with the path. This shows that the global gate does adjust the fusion weights of the two paths node by node, and that the module does not degenerate into an equal-weight concatenation of the alignment branch and the enhancement branch. It also demonstrates the effectiveness of choosing global pooling rather than pointwise prediction in Section 3.3. Global pooling produces only two scalars, which avoids the problem that the single-point noise response of an object amplifies the jitter of the gate coefficient.

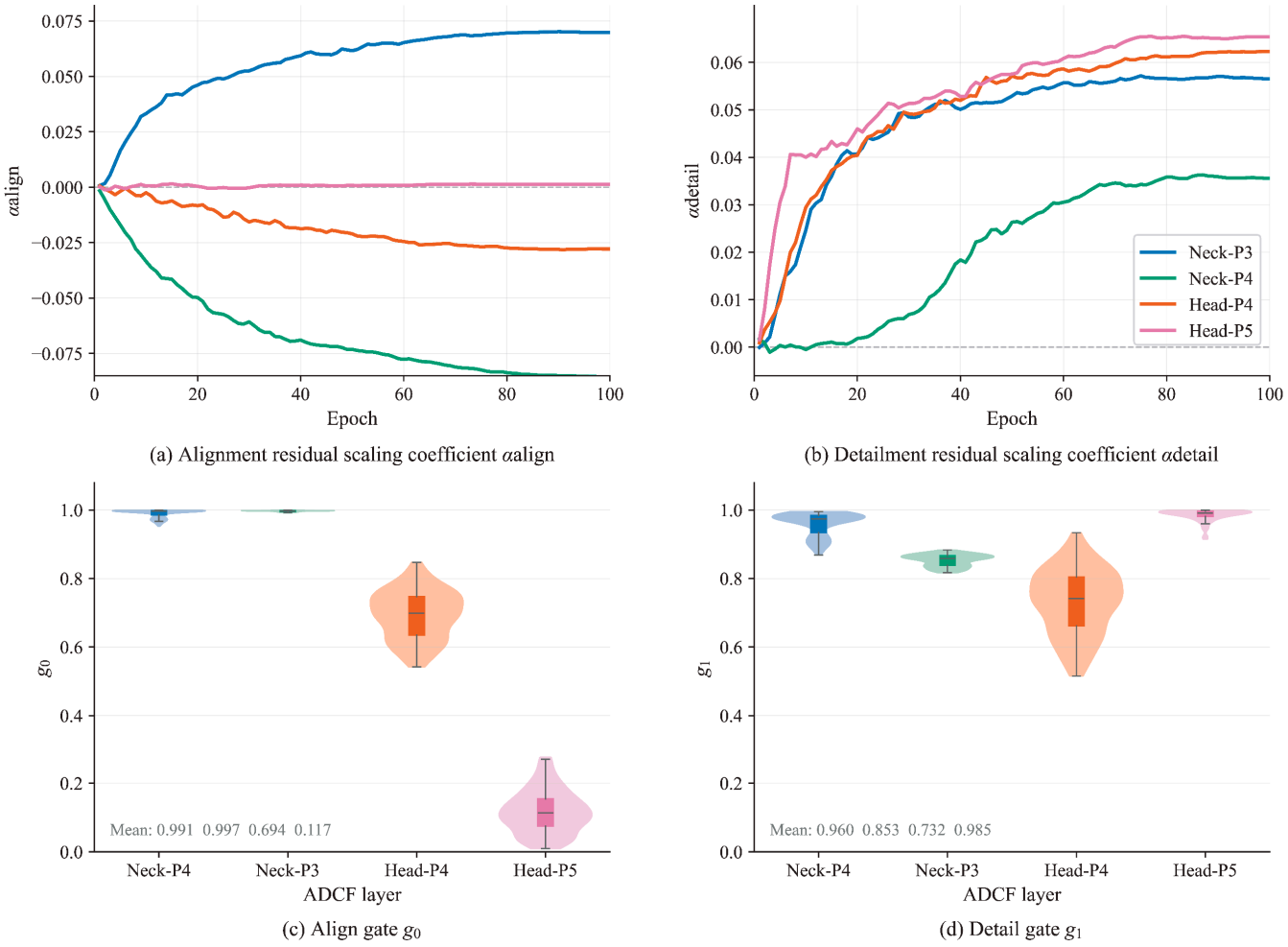


Figure 9. Residual scaling coefficients and gate distributions of ADCF. (**a**) Alignment residual scaling coefficient α_{align}; (**b**) Detailment residual scaling coefficient α_{detail}; (**c**) Align gate g_0; (**d**) Detail gate g_1.

4.3.3. Verification of the Sub-Pixel Translation Stability of SPSD

To reveal the working mechanism of SPSD, this section carries out a set of translation stability experiments. On the refueling interface dataset, a translation sweep over $\delta \in \{-2.00, -1.75, \ldots, +2.00\}$ px with a step of 0.25 px was performed by bilinear grid sampling, and the residual drift at the center of the predicted box was adopted as the stability metric. The results are shown in Figure 10: both curves respond numerically over the interval, which shows that phase aliasing really exists and is not caused by the fluctuation of individual samples. The average residual drift of the baseline YOLO26 is 0.892 px, 0.193 px lower than the perturbation amplitude of $\delta = 2$ px. For the refueling interface dataset, this is enough to

make the localization box jitter noticeably between adjacent frames. Two further conclusions can be drawn from Figure 10. The curves are not monotonic but fluctuate with δ, yet the two curves fluctuate in the same pattern, which shows that this variation comes from phase aliasing itself rather than from training noise. At the same time, the curve obtained after SPSD is introduced lies below the baseline, and the overall average residual drift falls to 0.810 px, a relative reduction of 9.2%. This shows that, by constructing a phase-shifted view and aligning the student prediction pixel by pixel with the teacher prediction after it has been resampled back to the coordinate system of the original image, SPSD does impose an effective constraint on cross-view consistency and makes the response of the network more stable under sub-pixel perturbation.

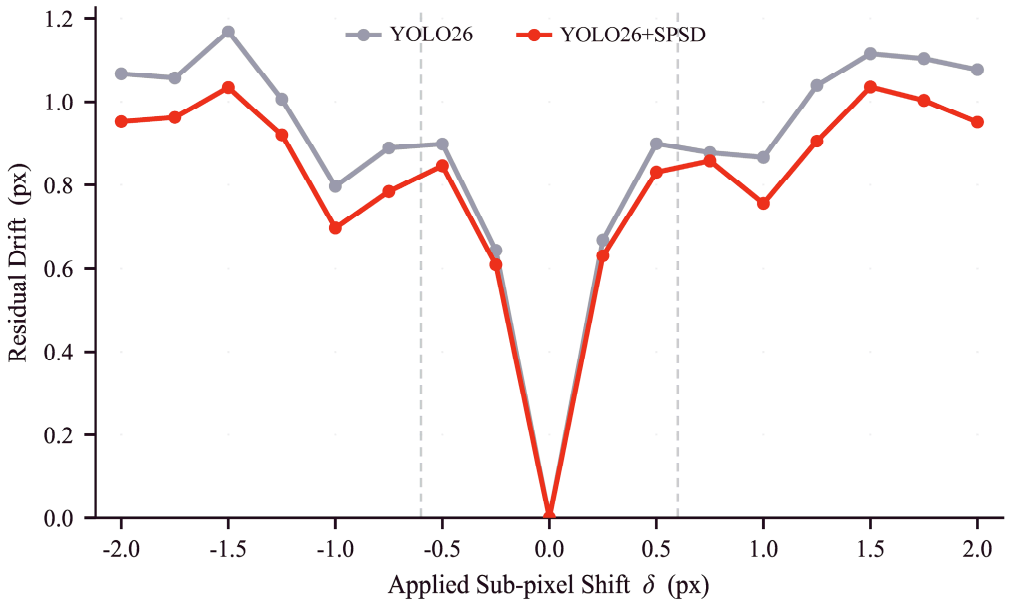


Figure 10. Effect of SPSD on stability under sub-pixel translation.

4.3.4. Verification of the Counterfactual Mechanism of CCD

The background suppression term proposed in Section 3.5 applies a one-sided penalty with Softplus to the response on the target-category channel of the target-erasure view; its purpose is not to suppress the response strength indiscriminately but to suppress specifically the spurious response that remains after the target has been erased. To verify whether CCD works according to this mechanism, this section carries out a set of counterfactual experiments. In the experiment, the object region is filled with the mean of the image channels to construct the target-erasure view, and it is then checked whether the model still produces a response in the object region. Figure 11a,b are two-dimensional heatmaps of the residual response obtained by aligning and normalizing the bounding boxes and superimposing them. Figure 11c,d are the profile curves along the width of the bounding box together with their difference. It can be seen from the figure that the baseline model retains a clear residual response at the original position of the object, which shows that its prediction depends to a considerable extent on background features that co-occur with the object. After CCD is introduced, this region is markedly weakened. The overall average residual response falls from 4.61×10^{-4} to 2.58×10^{-4}. Even more telling is the spatial selectivity of the suppression strength. The difference curve of Figure 11d is negative at 20 of the 24 sampling positions, with an average drop of 35.4%. Let ζ be the distance from the box center normalized by the box half-width and half-height, so that $\zeta = 1$ on the box boundary: the inner region is $\zeta \leq 0.5$ and the band near the boundary is $\zeta \geq 0.8$; the average response in the inner region of the box ($\zeta \leq 0.5$) falls by 52.0%, whereas the average response

near the box boundary ($\zeta \geq 0.8$) falls by only 21.1%. If CCD simply lowered the whole response map indiscriminately, the drops in the two regions would be roughly the same. In the actual experiment, however, the drop inside the box reaches 2.5 times that at the boundary. This shows that CCD acts mainly on the spurious response that remains after the object has been erased, while the contribution of the genuine background outside the box to detection is preserved. By counterfactual context decoupling, CCD makes the object response focus more on the object's own features, which explains at the mechanism level why accuracy improves after CCD is introduced.

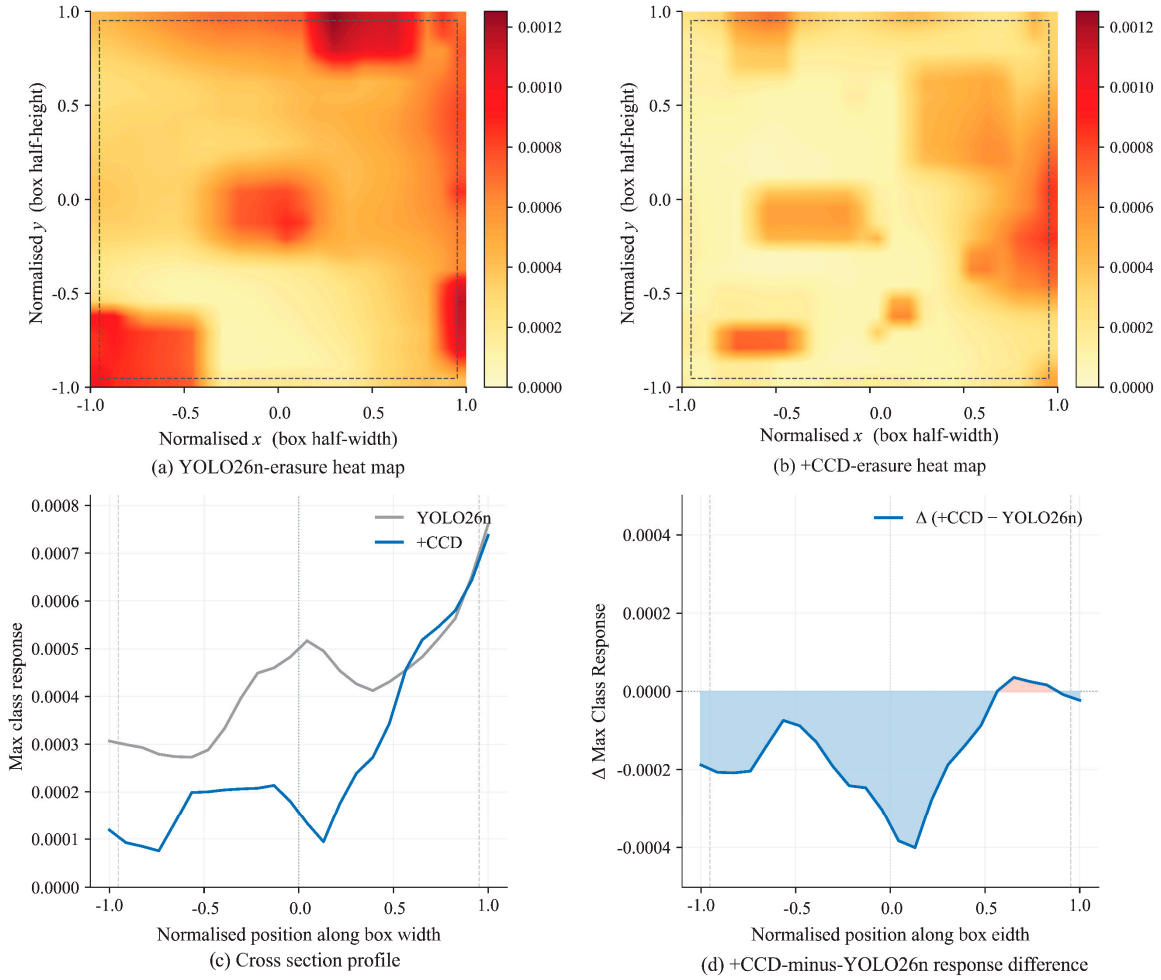


Figure 11. Counterfactual experiment on the background suppression term of CCD. (**a**,**b**) Two-dimensional distribution of the maximum residual class response after target erasure; (**c**,**d**) profile curves along the width of the bounding box and their difference.

To show the working mechanism of CCD directly from the visual side, Figure 12a–c present the input image, the YOLO26n saliency map and the ADCF-YOLO saliency map of two samples. The strip beneath each saliency map divides the total attribution mass of that image into the share inside the bounding boxes (green) and the share outside them (red); it carries no numerical values and serves only for a quick reading of where the response goes. On the baseline strips, red dominates, and the response falls mainly on background structures such as the workbench edge and the ground. On the ADCF-YOLO strips, however, green dominates, and the response has already converged onto the object itself. Figure 12d gives the pixel-wise difference Δ = *ours* − *baseline*, beneath which is a seismic color scale (blue: response removed; red: response added). The added response concentrates on the object region, whereas the removed response lies mainly in the background.

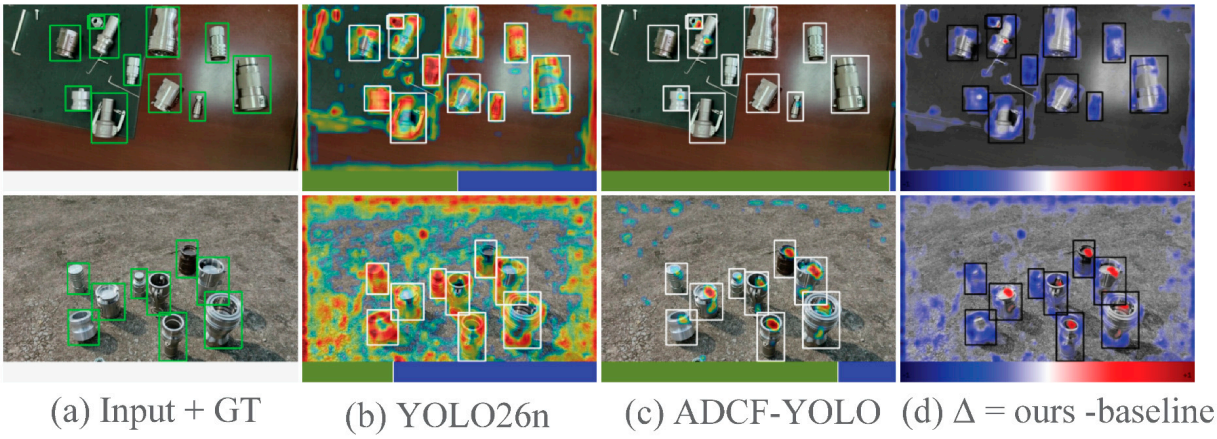


Figure 12. Attribution migration from context to the object.

To test whether CCD changes the model's reliance on the background, 80 additional images were acquired from scenes disjoint from the training set. The background replacement follows the construction of Section 3.5: the test images are cyclically shifted by one position so that every image obtains a donor image, the ground-truth boxes are kept unchanged, and every pixel outside the union of the boxes is replaced by the corresponding pixel of the donor. The intervention therefore differs from the training-time view of Equation (20) only in that the whole background is replaced instead of the annulus alone: the object is preserved exactly, while all of its original context is removed. Detection rate is the fraction of annotated objects still detected at IoU = 0.5. On the original images, the detection rates of YOLO26 and YOLO26+CCD are 96.1% and 97.2%; after replacement, they are 92.3% and 92.9%. The CCD model thus starts 1.1 points higher. The complementary object-erasure experiment gives false-positive rates of 3.07% for YOLO26 and 1.99% for YOLO26+CCD, and the residual-response analysis of Section 4.3.4 points in the same direction at the feature level.

4.4. Validation on Additional Datasets

Each dataset was retrained and evaluated with the same pre-trained weights and the same hyperparameters in Table 1, without any weight transfer across datasets. What this section therefore tests is not zero-shot transfer but whether the three proposed changes remain effective when the categories, the background composition and the acquisition conditions all change. Table 7 gives the results on the Industrial mechanical parts dataset. ADCF-YOLO achieves the best precision (0.973), recall (0.908), mAP@0.5 (0.964) and mAP@0.5:0.95 (0.818). mAP@0.5:0.95 is 0.031 higher than the baseline and 0.011 higher than the second-best result, YOLOv12. This dataset and the refueling interface dataset both contain metallic, weakly textured, rotationally symmetric connectors, but their category sets, background composition and acquisition conditions are independent of each other. ADCF-YOLO remains effective here, which indicates that its gain does not come from overfitting to the specific categories or scenes of one dataset.

Table 8 gives the generalization results on VisDrone-DET. The mAP@0.5 and mAP@0.5:0.95 of ADCF-YOLO are 0.331 and 0.188, improvements of 0.014 and 0.012 over the baseline; recall rises from 0.342 to 0.349, and precision falls slightly from 0.452 to 0.445. Compared with the refueling interface dataset, the gain on this dataset is markedly smaller, and precision decreases slightly. This is because VisDrone-DET is collected from a drone platform and differs markedly from the refueling interface dataset in imaging viewpoint, scale distribution and scene composition, which makes it a typical cross-domain transfer task. The model therefore completes the transfer mainly through its general feature extraction capability, whereas the three improvements of ADCF-YOLO rest on the assumption

that the object scale and the grid stride are close; when the object scale drops further into the extremely small range, part of the design ceases to function. ADCF-YOLO improves stably and substantially on a different task in the same domain and keeps a positive but limited gain on cross-domain scene tasks.

Table 7. Generalization experiment on the Industrial mechanical parts dataset.

Baseline	Precision	Recall	mAP@0.5	mAP@0.5:0.95	GFLOPs	Params (M)	Model Size (MB)	FPS	Latency (ms)
YOLOv8	0.945	0.807	0.881	0.738	8.1	3.008	6.3	335	86.3
YOLOv10	0.939	0.801	0.865	0.722	6.5	2.267	5.8	303	91.2
YOLO11	0.946	0.805	0.853	0.717	6.3	2.584	5.5	337	93.3
YOLO12	0.936	0.894	0.913	0.807	6.3	2.558	5.6	262	116.1
YOLO26	0.959	0.881	0.935	0.787	5.2	2.376	5.4	349	110
ADCF-YOLO	**0.973**	**0.908**	**0.964**	**0.818**	6.7	2.763	6.2	287	190

Table 8. Generalization experiment on the VisDrone-DET.

Baseline	Precision	Recall	mAP@0.5	mAP@0.5:0.95	GFLOPs	Params (M)	Model Size (MB)	FPS	Latency (ms)
YOLOv8	0.409	0.327	0.316	0.181	8.1	3.008	6.3	335	86.3
YOLOv10	0.404	0.306	0.302	0.173	6.5	2.267	5.8	303	91.2
YOLO11	0.405	0.321	0.308	0.174	6.3	2.584	5.5	337	93.3
YOLO12	0.429	0.319	0.319	0.183	6.3	2.558	5.6	262	116.1
YOLO26	0.452	0.342	0.317	0.176	5.2	2.376	5.4	349	110
ADCF-YOLO	0.445	0.349	0.331	0.188	6.7	2.763	6.2	287	190

5. Conclusions

This paper proposes ADCF-YOLO for the detection of refueling interfaces that are highly similar in appearance. The main contribution of this paper is the proposal of ADCF, SPSD and CCD, which optimize cooperatively against the three failure modes of spatial position deviation, phase jitter and background dependence. ADCF replaces the equal-weight concatenation of the neck with a gated residual fusion operator that aligns first and then enhances, compensating explicitly for the spatial misalignment introduced by resampling through a neighborhood convex combination. The two training-time supervisions, SPSD and CCD, suppress the localization jitter caused by sub-pixel translation from the phase dimension and cut the dependence of the model on background features from the causal dimension. Neither of them changes the network structure or the computational cost at inference time, so they carry zero deployment cost. In addition, to solve the problem of the scarcity of datasets for the refueling interfaces of special-purpose equipment, we constructed the refueling interface dataset. On this dataset, the mAP@0.5:0.95 of ADCF-YOLO rises from the baseline value of 0.482 to 0.526, and mAP@0.5 reaches 0.959.

The ablation experiments show that the three improvements act in positive synergy: no two-module combination matches the complete model on mAP@0.5 and on recall, where the best pairs still trail by 0.008 (SPSD+CCD on mAP@0.5) and 0.003 (ADCF+CCD on recall); on mAP@0.5:0.95, the SPSD+CCD pair reaches 0.526 and equals the complete model. The mechanism analysis shows that the average residual drift under sub-pixel translation is reduced by 9.2% and the residual response in the central region after target erasure is reduced by 52.0%. ADCF-YOLO keeps a positive gain on the two public datasets Industrial mechanical parts and VisDrone-DET, which shows that its benefit comes from handling a problem common to weak texture object detection rather than from overfitting to a particular

scene. The experimental results show that ADCF-YOLO can provide high-accuracy input for the pose estimation and compliant docking of a refueling robot, thereby reducing the risk of equipment damage and fuel leakage caused by false detections. The method in this paper nevertheless has limitations in four respects. Because the test set is a single held-out scene, the improvements should be read as a consistent trend rather than as a precision claim. The effective supervision of SPSD covers only about 1.6% of the pixels, which ensures reliability but limits the gain of this single item, and it partly fails when the object scale drops into the extreme range, which is the main reason for the narrower gain in the cross-domain experiment. CCD depends on the circular shift of samples within a batch, so its decoupling effect weakens under small batches. Meanwhile, this paper is limited to applications on edge devices. Future work will therefore benchmark the model on embedded modules together with TensorRT conversion, in addition to denser phase consistency supervision and cross-batch context intervention.

Author Contributions: Conceptualization and methodology, S.Z.; resources, W.C.; writing—original draft preparation, Z.W.; writing—review and editing, G.F.; visualization, S.C.; supervision, W.H.; funding acquisition, Y.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Science and Technology Research Program of Chongqing Municipal Education Commission (Grant KJZD-M202412901, Grant KJZD-M202312901, and Grant KJQN202312908), the Chongqing Municipal Graduate Research Innovation (Grant CYB25343), and the China Postdoctoral Science Foundation (Grant 2026T191158).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset is available on the Ultralytics Platform: https://platform.ultralytics.com/fei-li/datasets/(accessed on 08 November 2026).

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

References

1. Bi, Z.; Luo, C.; Miao, Z.; Zhang, B.; Zhang, C. Automatic robotic recharging systems—Development and challenges. *Ind. Robot.* **2020**, *48*, 95–109. [CrossRef]
2. Zhou, S.; Cui, W.; Bai, W.; Chen, S.; Hua, W.; Wu, Z.; Chen, Y. Current research status and key technological advances of refueling robots. *Machines* **2026**, *14*, 892. [CrossRef]
3. Yang, M.; Liu, J. Research on six-degree-of-freedom refueling robotic arm positioning and docking based on RGB-D visual guidance. *Appl. Sci.* **2024**, *14*, 4904. [CrossRef]
4. Zhang, X.; Ma, M.; Zhao, Y. Oil tank cap detection based on mathematical morphology and Hough transformation. *Manuf. Autom.* **2018**, *40*, 119–122.
5. Liang, M.; Liu, X.; Huang, P. Edge detection method suitable for automobile fuel tank cover. *Comput. Eng. Des.* **2021**, *42*, 2358–2365.
6. Liang, M.; Huang, P.; Liu, X. Visual detection method for automobile fuel tank cover based on HoG and SVM in unstructured environment. *Mach. Tool Hydraul.* **2022**, *50*, 20–25.
7. Li, W.; Zhang, H.; Wan, C. Application and performance evaluation of object detection technology based on computer vision. In Proceedings of the 14th Asian Control Conference, Dalian, China, 5–8 July 2024; pp. 76–80.
8. Zhang, Q.; Li, Y. Research on recognition method of car fuel tank cap based on residual network. *J. Phys. Conf. Ser.* **2021**, *1748*, 042017. [CrossRef]
9. Fu, H.; Hu, C.; Guo, J.; An, K. Research on detection method of fuel tank cover based on lightweight YOLO. In Proceedings of the 3rd 2023 International Conference on Autonomous Unmanned Systems, Nanjing, China, 9–11 September 2023; pp. 342–352.
10. Qiao, Q.; Guo, J.; Wang, L.; Guo, Y.; An, K. FastVim-YOLO: Lightweight fuel tank cap detection network for refueling robots. In Proceedings of the 4th 2024 International Conference on Autonomous Unmanned Systems, Shenyang, China, 19–21 September 2024; pp. 492–505.

11. Zhang, X.; Guan, Y.; Li, Q. Research on intelligent detection models for automotive fuel caps in autonomous refueling systems under complex scenarios. In Proceedings of the 2024 3rd International Conference on Artificial Intelligence and Intelligent Information Processing, Tianjin, China, 25–27 October 2024; pp. 205–210.
12. Dai, J.; Fu, L.H.; Li, Y.Q.; Zhao, J.W.; Hanajima, N. Lightweight SL-YOLO algorithm for automotive fuel tank cover detection. *Signal Image Video Process.* **2025**, *19*, 11. [CrossRef]
13. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.M.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the 30th IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
14. Liu, S.; Qi, L.; Qin, H.F.; Shi, J.P.; Jia, J.Y. Path aggregation network for instance segmentation. In Proceedings of the 31st IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
15. Tan, M.X.; Pang, R.M.; Le, Q. EfficientDet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020; pp. 10778–10787.
16. Liu, S.; Huang, D.; Wang, Y.-H. Learning spatial fusion for single-shot object detection. *arXiv* **2019**, arXiv:1911.09516. [CrossRef]
17. Wang, J.Q.; Chen, K.; Xu, R.; Liu, Z.W.; Loy, C.C.; Lin, D.H. CARAFE: Content-Aware ReAssembly of FEatures. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3007–3016.
18. Liu, W.Z.; Lu, H.; Fu, H.T.; Cao, Z.G. Learning to upsample by learning to sample. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–6 October 2023; pp. 6004–6014.
19. Dai, J.F.; Qi, H.Z.; Xiong, Y.W.; Li, Y.; Zhang, G.D.; Hu, H.; Wei, Y.C. Deformable convolutional networks. In Proceedings of the 16th IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 764–773.
20. Zhu, X.Z.; Hu, H.; Lin, S.; Dai, J.F. Deformable convNets v2: More deformable, better results. In Proceedings of the 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 9300–9308.
21. Kisantal, M.; Wojna, Z.; Murawski, J.; Naruniec, J.; Cho, K. Augmentation for small object detection. *arXiv* **2019**, arXiv:1902.07296. [CrossRef]
22. Ghiasi, G.; Cui, Y.; Srinivas, A.; Qian, R.; Lin, T.Y.; Cubuk, E.D.; Le, Q.; Zoph, B. Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 2917–2927.
23. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.M.; Dollár, P. Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 318–327. [CrossRef]
24. Li, B.Y.; Liu, Y.; Wang, X.G. Gradient Harmonized Single-Stage Detector. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence/31st Innovative Applications of Artificial Intelligence Conference/9th AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 8577–8584.
25. Shrivastava, A.; Gupta, A.; Girshick, R. Training Region-based Object Detectors with Online Hard Example Mining. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 761–769.
26. Akyon, F.C.; Altinuc, S.O.; Temizel, A. Slicing Aided Hyper Inference and Fine-Tuning for Small Object Detection. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Bordeaux, France, 16–19 October 2022; pp. 966–970.
27. Zhang, R. Making Convolutional Networks Shift-Invariant Again. In Proceedings of the 36th International Conference on Machine Learning (ICML), Long Beach, CA, USA, 9–15 June 2019; pp. 7324–7334.
28. Azulay, A.; Weiss, Y. Why do deep convolutional networks generalize so poorly to small image transformations? *J. Mach. Learn. Res.* **2019**, *20*, 1–25.
29. Shi, W.Z.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A.P.; Bishop, R.; Rueckert, D.; Wang, Z.H. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1874–1883.
30. Tarvainen, A.; Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 1195–1204.
31. Sohn, K.; Berthelot, D.; Li, C.-L.; Zhang, Z.; Carlini, N.; Cubuk, E.D.; Kurakin, A.; Zhang, H.; Raffel, C. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. In Proceedings of the Advances in Neural Information Processing Systems 33 (NeurIPS 2020), Vancouver, BC, Canada, 6–12 December 2020; pp. 596–608.
32. Liu, Y.C.; Ma, C.Y.; Kira, Z. Unbiased Teacher v2: Semi-supervised Object Detection for Anchor-free and Anchor-based Detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 9809–9818.

33. Xu, M.D.; Zhang, Z.; Hu, H.; Wang, J.F.; Wang, L.J.; Wei, F.Y.; Bai, X.; Liu, Z.C. End-to-End Semi-Supervised Object Detection with Soft Teacher. In Proceedings of the 18th IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 11–17 October 2021; pp. 3040–3049.
34. Wei, X.S.; Song, Y.Z.; Aodha, O.M.; Wu, J.; Peng, Y.; Tang, J.; Yang, J.; Belongie, S. Fine-Grained Image Analysis with Deep Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 8927–8948. [CrossRef]
35. Deng, J.K.; Guo, J.; Xue, N.N.; Zafeiriou, S. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In Proceedings of the 32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 4685–4694.
36. Su, Y.K.; Sun, R.Z.; Lin, G.S.; Wu, Q.Y. Context Decoupling Augmentation for Weakly Supervised Semantic Segmentation. In Proceedings of the 18th IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 11–17 October 2021; pp. 6984–6994.
37. Song, H.X.; Liang, L. Causal-YOLO: Enhancing object detection in remote sensing imagery through causal feature alignment. *Opt. Laser Technol.* **2026**, *200*, 115167. [CrossRef]
38. Devries, T.; Taylor, G. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv* **2017**, arXiv:1708.04552. [CrossRef]
39. Huang, Z.Y.; Wang, H.H.; Xing, E.P.; Huang, D. Self-challenging Improves Cross-Domain Generalization. In Proceedings of the 16th European Conference on Computer Vision-ECCV-Biennial, Glasgow, UK, 23–28 August 2020; pp. 124–140.
40. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In Proceedings of the 13th European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2014; pp. 740–755.
41. He, K.M.; Zhang, X.Y.; Ren, S.Q.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
42. Jocher, G.; Chaurasia, A.; Qiu, J. YOLOv8. 2023. Available online: https://github.com/ultralytics/ultralytics (accessed on 1 September 2026).
43. Wang, A.; Chen, H.; Liu, L.; Chen, K.; Lin, Z.; Han, J.; Ding, G. YOLOv10: Real-Time End-to-End Object Detection. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Vancouver, BC, Canada, 10–12 December 2024; pp. 107984–108011.
44. Jocher, G.; Qiu, J. Ultralytics YOLO11. 2024. Available online: https://github.com/ultralytics/ultralytics (accessed on 1 September 2026).
45. Tian, Y.J.; Ye, Q.X.; Doermann, D. YOLOv12: Attention-Centric Real-Time Object Detectors. In Proceedings of the Advances in Neural Information Processing Systems 38 (NeurIPS), Vancouver, BC, Canada, 1–7 December 2025; pp. 78433–78457.
46. Jocher, G.; Qiu, J.; Liu, M.Y.; Lyu, S.; Akyon, F.C.; Kalfaoglu, M.E. Ultralytics YOLO26: Unified Real-Time End-to-End Vision Models. *arXiv* **2026**, arXiv:2606.03748. [CrossRef]
47. Li, H.L. Rethinking Features-Fused-Pyramid-Neck for Object Detection. In Proceedings of the 18th European Conference on Computer Vision (ECCV), Milan, Italy, 29 September–4 October 2024; pp. 74–90.
48. Li, X.F.; Li, H.Y.; Zhou, H.; Yu, M.M.; Chen, D.; Li, S.H.; Zhang, J. Camouflaged object detection with counterfactual intervention. *Neurocomputing* **2023**, *553*, 126530. [CrossRef]
49. Wojtalik, M. Mechanical Parts Detection 2. 2026. Available online: https://platform.ultralytics.com/mateusz-wojtalik/datasets/mechanical-parts-detection-2 (accessed on 1 September 2026).
50. Zhu, P.F.; Wen, L.Y.; Du, D.W.; Bian, X.; Fan, H.; Hu, Q.H.; Ling, H. Detection and Tracking Meet Drones Challenge. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *44*, 7380–7399. [CrossRef]
51. Robinson, I.; Robicheaux, P.; Popov, M.; Ramanan, D.; Peri, N. RF-DETR: Neural architecture search for real-time detection transformers. *arXiv* **2025**, arXiv:2511.09554. [CrossRef]
52. Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs beat YOLOs on real-time object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024), Seattle, WA, USA, 17–21 June 2024; pp. 16965–16974.
53. Chen, N.; Liu, S.; Shen, X.; Zhang, Y. CSDETR: A lightweight and efficient real-time traffic light detection model based on improved RT-DETR. *J. Real-Time Image Process.* **2026**, *23*, 67. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.