

ZIRF: Zero-Initialized Residual Fusion for Small Object Detection

Seon-Min Kim

Sunchon National University

Seung-Hwi Choi

Sunchon National University

Sung-Wook Park

Mokpo National University

Chun-Bo Sim

Sunchon National University

Se-Hoon Jung

`sh.jung@scnu.ac.kr`

Sunchon National University

Article

Keywords: Small Object Detection, Feature Pyramid, Cross-layer Fusion, Zero Initialization, Drone Imagery

Posted Date: October 22nd, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11204539/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

ZIRF: Zero-Initialized Residual Fusion for Small Object Detection

Seon-Min Kim¹, Seung-Hwi Choi², Sung-Wook Park³, Chun-Bo Sim^{2,*} & Se-Hoon Jung^{1,*}

¹ Sunchon National University, Department of Computer Engineering, Suncheon, 57922, Korea

² Sunchon National University, Interdisciplinary Program in IT-Bio Convergence System, Suncheon, 57922, Korea

³ Mokpo National University, Department of AI Computing and Security, Muan, 58554, Korea

* Corresponding authors: cbsim@scnu.ac.kr & shjung@scnu.ac.kr

Abstract

Object detection identifies the locations and classes of objects of interest in an image. It is widely used for drone and road imagery, where most objects are small and densely packed, making detection difficult. In small object detection, the bottleneck lies not in recognizing that an object exists but in precisely regressing its boundary. Existing work addresses this with high-resolution detection heads and feature pyramid necks, but Feature Pyramid Network- and Path Aggregation Network-style necks sequentially fuse features only between adjacent scales. Thus, high-level semantic information is diluted before it reaches the highest-resolution output. To compensate for this dilution, we propose Zero-Initialized Residual Fusion (ZIRF), a lightweight cross-layer fusion module built on a zero-initialized residual. ZIRF directly fuses the features of the other three levels into each of the four detection outputs through a single transformation. A gate computed from the target output selects the incoming information at each spatial location, and a residual scalar initialized to zero controls the contribution of the entire fusion path. Immediately after insertion, the model produces exactly the same output as before insertion, and fusion is incorporated during training only to the extent that it is needed. ZIRF does not replace the existing neck and is added as a residual to the neck output. The proposed model improved mean Average Precision @0.5:0.95 by +2.5%p over YOLOv11s on the SODA-D dataset, of which +0.66%p was attributable to ZIRF. On the VisDrone dataset, the average ZIRF gain was larger in the tile form than in the whole-image form, showing that the ZIRF effect depends on object size at the 640×640 input. These results indicate that zero-initialized cross-layer fusion can augment an existing detection neck with a small parameter cost while leaving the baseline output unchanged at insertion.

Keywords: Small Object Detection, Feature Pyramid, Cross-layer Fusion, Zero Initialization, Drone Imagery

Introduction

Small object detection is a core function in intelligent transportation systems, urban monitoring, and search and rescue. Drone and autonomous-driving imagery contains small objects at high densities, and as small objects become denser, precise estimation of bounding box coordinates becomes difficult. The smaller the object, the more sharply the Intersection over Union (IoU) drops even under a slight localization error, which widens the gap between the mean Average Precision (mAP)@0.5, measured at an IoU threshold of 0.5, and mAP@0.5:0.95, averaged over IoU thresholds from 0.5 to 0.95.

Precise localization requires two components. The first is high-resolution features that preserve the fine spatial information of small objects, and the second is multiscale fusion that delivers high-level semantic information contained in P5 down to the high-resolution features of P3 and P2. Prior work has developed the former into a stride-4 P2 head and the latter into necks ranging from Feature Pyramid Network (FPN)¹ and Path Aggregation Network (PAN)² to Bidirectional Feature Pyramid Network (BiFPN)³ and Reparameterized Generalized Feature Pyramid Network (RepGFPN)⁴. We adopt YOLOv11s⁵ combined with a P2 head and RepGFPN as the baseline and denote it P2RG. However, P2RG has a limitation in its information propagation path. PAN-style necks propagate information in sequence only between adjacent levels. The features of P5 reach the P2 output only after passing through the fusion stages of P4 and P3, and they are mixed with neighboring levels at each stage through upsampling and convolution, so the original information fades. Because small objects are detected at the P2 output, high-level semantic information carried by the low-resolution P5 features arrives weakened at exactly the place where it is needed.

Reducing the information loss caused by sequential propagation requires fusion that directly connects non-adjacent levels. If each output directly fuses the features of all other levels, P5 features can be delivered to P2 without passing through P4 and P3. Adaptively Spatial Feature Fusion (ASFF)⁶, Gated Fully Fusion (GFF)⁷, and Gold-YOLO⁸ are representative examples. However, these three methods have two problems when used to augment existing necks. First, they replace the existing fusion structure, so the neck must be retrained from scratch. Second, fusion is active from the start of training, so information from other levels whose usefulness has not yet been verified is mixed into the neck features. To avoid both replacing the existing neck and activating cross-layer fusion from the beginning of training, we propose a Zero-Initialized Residual Fusion (ZIRF) module. For each of the four detection outputs produced by P2RG, ZIRF directly fuses the features of the remaining three levels and adds the fused result to the existing output, scaled by a residual scalar γ initialized to zero.

In the initial state with $\gamma = 0$, the P2RG output before ZIRF insertion is preserved, and as training proceeds, the residual scalar is learned so that the necessary cross-layer information is gradually incorporated.

Inserting ZIRF into P2RG improves mAP@0.5:0.95 by +0.66%p on the SODA-D⁹ dataset. When the original VisDrone¹⁰ dataset images are organized in the whole-image and tile forms, the median object size at the 640×640 input differs (11.3 px versus 23.0 px), and the average ZIRF gain is larger in the tile form, where objects remain larger. This shows that the ZIRF effect is related to object size at the 640×640 input rather than to the dataset.

The contributions of this paper are as follows. First, we analyze the sequential propagation of high-level semantic information from P5 to P2 in PAN-style necks as a structural bottleneck for small object localization. Second, we propose ZIRF, which directly fuses non-adjacent pyramid features without replacing the existing neck and preserves the existing output through a zero-initialized residual. Third, through a controlled experiment that organizes the original VisDrone images in the whole-image and tile forms and cross-compares P2RG and ZIRF, we analyze the relationship between object size and the ZIRF gain with the dataset held fixed.

Related Work

Small Object Detection

Small object detection differs from general object detection in that the object occupies few pixels and therefore offers little discriminative appearance. Small objects are defined by the COCO¹¹ convention as those with an area of 32^2 px or less. They shrink to a few pixels in low-resolution feature maps, such as P5, where downsampling accumulates and object information vanishes. The difficulty is compounded by low contrast against backgrounds, clustering, occlusion, and large-scale variation. The small object problem is most pronounced in drone and autonomous-driving imagery. The SODA-D and VisDrone datasets are the representative datasets used in small object studies.

Small object detection research is divided into approaches that modify networks and those that modify training objectives. Architecture-level approaches have developed along three lines: region partitioning and reinspection, sparse and query-based processing, and high-resolution heads and neck redesign. First, region partitioning and reinspection enlarges and redetects subregions with high object density to raise the effective resolution. ClusDet¹² estimates cluster regions, and DMNet¹³ determines partition regions from density maps. UFPMP-Det¹⁴ processes foreground jointly with multiple proxies, and SAHI¹⁵ presents general-purpose tiling applied to both training and inference. Tiling cuts the input image into a grid and processes each tile as an individual input, increasing the number of pixels an object occupies at a fixed input resolution. However, region partitioning and reinspection requires repeated inference or a separate region-estimation step, sacrificing the real-time capability of a single forward pass. Second, sparse and query-based processing reduces the computational burden of high-resolution features. QueryDet¹⁶ first predicts candidate locations of small objects on low-resolution feature maps and then performs detection only on the high-resolution features at those locations, and CEASC¹⁷ uses adaptive sparse convolution. Third, high-resolution heads and neck redesign handles small objects in a single pass by adding a high-resolution detection head or redesigning the neck. Following TPH-YOLOv5¹⁸, MASF-YOLO¹⁹, SL-YOLO²⁰, and SOD-YOLOv8²¹ combine a stride-4 P2 head with an enhanced fusion neck. In an ablation study of MASF-YOLO, adding a P2 head yielded the largest gain among the individual components.

Objective-level approaches keep the network architecture and redefine the label assignment criterion and loss function for small objects. To mitigate the sensitivity of IoU to localization error on very small bounding boxes, the normalized Wasserstein distance²² and Gaussian receptive-field-based label assignment²³ have been proposed. CFNet²⁴, which refines initial candidate regions in a coarse-to-fine manner, performed well on SODA-D.

This study applies the high-resolution head and neck redesign line and retains a single forward pass without region partitioning or repeated inference. In contrast with prior studies that redesign the neck architecture, we leave the RepGFPN neck of the P2RG baseline unchanged and insert only a residual path at its output. The output immediately after module insertion is kept identical to that of the baseline, and the contribution of cross-layer information is incorporated gradually during training. We also note that the tiling used by region partitioning and reinspection increases the number of pixels an object occupies at a fixed input, and analyze the relationship between object size and the ZIRF gain on the original VisDrone images and bounding box labels.

Feature Pyramids and Multiscale Fusion

The neck, which combines features of different scales, is a core component of a detection model, and its connection topology has become progressively more elaborate. FPN established the standard of delivering the high-level semantic information of P5 to the high-resolution features of P3 and P2 through a top-down path, and PAN added a bottom-up path so that the fine localization information retained in high-resolution features propagates up to P5. NAS-FPN²⁵ used neural architecture search to automatically design a topology that includes non-adjacent connections, and BiFPN repeatedly stacks bidirectional paths and regulates each contribution with learnable normalized weights. The RepGFPN of DAMO-YOLO improves the balance between accuracy and speed through reparameterization and efficient aggregation. However, the pyramid structures from FPN to RepGFPN propagate information mainly between adjacent levels in sequence, and this study aims to resolve the semantic dilution caused by this sequential propagation.

Research on directly connecting all levels beyond adjacency has progressed. Gold-YOLO gathers all levels into a single global representation and redistributes it to each scale, Hyper-YOLO²⁶ models high-order relations between levels with hypergraph convolution, and CFPT²⁷ lets all levels interact directly through channel and spatial attention alone without upsampling, bypassing the semantic gap of sequential propagation. CFPT replaces the neck with an attention structure, and

the interaction between levels is active from the start of training. ZIRF is inserted at the RepGFPN outputs as a zero-initialized residual, so it preserves the existing structure, and its cost is limited to 0.60M parameters because it does not redesign the entire pyramid.

Attempts to adaptively regulate the contribution of each feature during fusion have proceeded in parallel. ASFF learns position-wise spatial weights on size-matched inputs for each output level, AugFPN²⁸ enhances feature representation to compensate for the information loss at the top-level P5 feature of FPN, and GFF selectively passes features of all levels through gates. All three methods replace the existing fusion structure. GFF is directly comparable to ZIRF in that it regulates the contribution of features from multiple levels using gates, but the two methods differ in three respects. First, GFF computes a gate at every level and admits more information from other levels as the target gate decreases, whereas ZIRF computes per-source gates only from the target output feature and regulates the inflow of each source. Second, the GFF sigmoid gate does not remove the initial influence of the added fusion path, whereas ZIRF initializes the residual scalar to zero and keeps the output immediately after module insertion identical to that of the baseline. Third, GFF replaces the entire pyramid of the segmentation backbone, whereas ZIRF is inserted as a residual at the output of the detection neck. In contrast with prior studies that redesign the neck, ZIRF adds cross-layer information while retaining the fusion weights acquired through training.

Residual Scaling and Identity Initialization

Techniques that start a residual branch from zero, so that a newly added branch does not disturb the existing signal propagation early in training, have been studied extensively in deep networks. Initializing the scale factor γ of the last batch normalization in a residual block to zero makes the block resemble an identity mapping at the start and stabilizes large-scale training²⁹⁻³⁰. Fixup³¹ achieves similar stability without normalization by initializing the branch weights to zero. ReZero³² and SkipInit³³ achieve the same effect simply by multiplying the residual branch by a single scalar initialized to zero, and LayerScale³⁴ extends this to per-channel fine-grained weighting to stabilize the training of deep vision transformers. The zero-convolution of ControlNet³⁵ follows the same principle in that it starts the control branch from zero and gradually couples it to a pretrained model.

These methods limit the effect of a newly added path on the existing signal propagation early in training and adjust the contribution of the path through training. ZIRF applies the zero-initialization principle to the cross-layer residual path after the detection neck. The implementation is described in the formulation of the proposed module.

Method

3.1 Overall Architecture

The architecture of the proposed model is shown in Fig. 1. The backbone extracts multiscale features from the input, the RepGFPN neck fuses the multiscale features, and the ZIRF module enhances each of the four outputs with cross-layer information. The enhanced features are passed to the four detection heads.

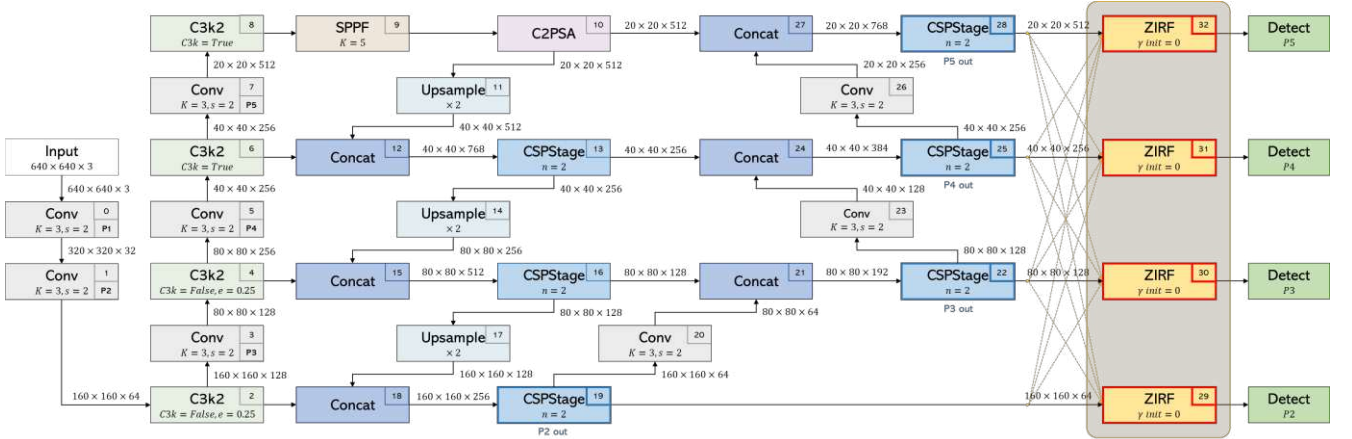


Figure 1. Overall architecture of the proposed model. The ZIRF blocks attached to the four neck outputs are the contribution of this work, and all remaining components form the baseline.

The backbone is YOLOv11s, which produces four pyramid features, $\{P_2, P_3, P_4, P_5\}$, from the input. The strides of the levels are 4, 8, 16, and 32, corresponding to spatial resolutions of 160^2 , 80^2 , 40^2 , and 20^2 for a 640×640 input. With the s scale, whose width multiplier is 0.50, the channel widths of the four levels are 64, 128, 256, and 512. Unlike the standard YOLO, which uses only three heads from P3 to P5, we add a stride-4 P2 head to preserve the fine spatial information of small objects.

For the fusion stage, we adopt RepGFPN of DAMO-YOLO and extend it to the P2 level. The neck propagates semantics from P5 to P2 along the top-down path and then passes localization information from P2 to P5 along the bottom-up path, refining the features at each fusion point with six reparameterized CSPStages, three on each path. Each stage consists of two blocks with SiLU activation and a hidden ratio of 1.0. This yields the output features $F_l, l \in \{2, 3, 4, 5\}$ of the P2RG baseline.

The RepGFPN neck sequentially fuses features of different scales only between adjacent levels. For a source level k to contribute to an output level l , the signal must pass through every level in between, so the number of stages it traverses

is proportional to the scale distance $|l - k|$. At each stage, the feature map is resized by a factor of two and fused with the neighboring feature by convolution. The effect of sequential propagation is greatest at P2, where small objects are decoded. Because the semantic information of P5 must pass through P4 and P3 to reach P2, the scale distance between P5 and P2 is the largest of the four levels. In the RepGFPN neck, the P5 features pass through three upsampling steps and three fusion blocks before reaching the P2 output, and at each block they are mixed with features of other levels and diluted.

After computing all four outputs, one ZIRF block from Fig. 2 is applied to each output F_l . Each block takes as input its own target F_l , shown as a black solid line in Fig. 1, and the remaining three levels, shown as yellow dashed lines, and produces the enhanced feature $\tilde{F}_l$. The set $\{\tilde{F}_2, \tilde{F}_3, \tilde{F}_4, \tilde{F}_5\}$ is passed to the detection heads. Each of the four blocks has an independent residual scalar γ_l , initialized to zero. Behind the neck output, each block can receive the outputs of all four levels as input, and because the fusion path of the RepGFPN neck is left unchanged, the connection topology of PAN is also preserved.

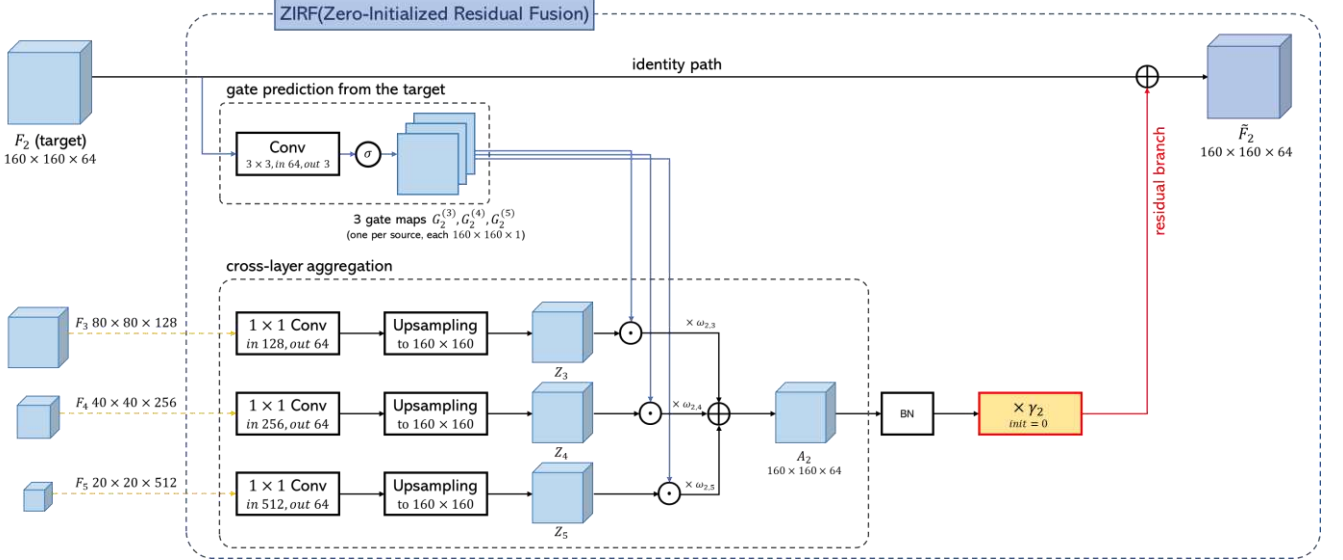


Figure 2. Internal structure of the ZIRF block for target P2 with sources P3 to P5.

3.2 Zero-Initialized Residual Fusion

Let the multiscale outputs of the RepGFPN neck be $F_l \in \mathbb{R}^{C_l \times H_l \times W_l}$, $l \in \{2, 3, 4, 5\}$. For each output level l , ZIRF directly fuses the remaining $n = 3$ levels to produce the enhanced feature $\tilde{F}_l$. Fig. 2 shows the pipeline using the block for target P2 as an example. Each source $k \neq l$ is projected to the target channel width C_l by a 1×1 convolution $W_{l,k}^p$ and then bilinearly interpolated to the target resolution.

$$\hat{F}_{l \leftarrow k} = U_l(W_{l,k}^p * F_k), \quad k \neq l \quad (1)$$

Unlike the sequential adjacent fusion of PAN, Equation (1) delivers the features of source level k to output level l with a single projection and interpolation, avoiding fusion that repeatedly passes through intermediate levels. Z_k in Fig. 2 corresponds to $\hat{F}_{l \leftarrow k}$ in Equation (1). The relative importance between sources is regulated by the per-source-level weight $\omega_{l,k}$, and the position-wise inflow by the content gate G_l .

$$\omega_{l,k} = \frac{\text{ReLU}(\alpha_{l,k})}{\sum_{k'} \text{ReLU}(\alpha_{l,k'}) + \epsilon}, \quad G_l = \sigma(W_l^g * F_l) \in (0,1)^{n \times H_l \times W_l} \quad (2)$$

In Equation (2), $\omega_{l,k}$ is the per-source-level weight that represents how much source level k contributes to output level l , constrained to non-negative values summing to 1 through ReLU and summation as in the fast normalization of BiFPN. ReLU stands for rectified linear unit. $\epsilon = 10^{-4}$ prevents the denominator from becoming zero. G_l is the set of n gate maps obtained by applying a 3×3 convolution W_l^g and a sigmoid to the target feature F_l , one per source. Because the gate is computed from the target output F_l rather than from the sources, each output modulates the incoming information position by position according to its own content. The weighted and gated sources are summed, batch-normalized, scaled by a single scalar γ_l initialized to zero, and added to the target as a residual. The affine transform of batch normalization rescales the per-channel distribution, and the residual contribution of the entire block is controlled by the single scalar γ_l .

$$\tilde{F}_l = F_l + \gamma_l \cdot \text{BN}(\sum_{k \neq l} \omega_{l,k} (G_l^{(k)} \odot \hat{F}_{l \leftarrow k})), \quad \gamma_l \leftarrow 0 \quad (3)$$

Equation (3) guarantees the preservation of the initial output. In the initial state $\gamma_l = 0$, $\tilde{F}_l = F_l$ and the model is numerically identical to P2RG. Let A_l denote the weighted sum in Equation (3) before batch normalization and δ_l the gradient of the loss L with respect to $\tilde{F}_l$. Then $\frac{\partial L}{\partial \gamma_l} = \langle \delta_l, BN(A_l) \rangle$, and because this gradient is generally non-zero, γ_l immediately departs from zero. From the moment $\gamma_l \neq 0$, the parameters of Equations (1) and (2) receive gradients. The additional parameters per block consist of bias-free 1×1 projections for each source, a 3×3 gate convolution, n scalar weights $\alpha_{l,k}$, affine batch normalization, and a single γ_l .

Experiments

4.1 Training and Test Details

All experiments were conducted in the environment listed in Table 1, and the training hyperparameters are given in Table 2. For optimization, we used MuSGD, which ultralytics³⁶ selects automatically at this training scale, updating matrix parameters with Muon³⁷ and the remaining parameters with momentum SGD. For a fair comparison, all variants used the same training configuration and module insertion position. Mosaic augmentation was disabled for the last 10 epochs. All reported performance and post-hoc analyses were based on the checkpoint with the highest ultralytics fitness among the EMA weights maintained during training.

Table 1. Experimental environment

Hardware Specifications		Software Versions	
CPU	Intel Core Ultra 7 265K	OS	Ubuntu 24.04.3 LTS
GPU	NVIDIA GeForce RTX 5090 32GB	Python	3.12.3
		PyTorch	2.8.0
RAM	G.Skill DDR5-4800 64GB	CUDA / cuDNN	12.9 / 9.10.2
		Ultralytics	8.4.42

Table 2. Training hyperparameters

Parameter	Value	Parameter	Value
Optimizer	MuSGD	Batch size	16
Momentum	0.937	Input resolution	640×640
Initial learning rate	0.01	Loss weights box, cls, dfl	7.5, 0.5, 1.5
Weight decay	5×10^{-4}	Mosaic	1.0
Warmup epochs, momentum	3, 0.8	Mixup	0.4
Epochs	100	Close mosaic	10

4.2 Datasets

We used SODA-D, a dataset dedicated to small objects, as the main benchmark. SODA-D consists of 24,828 high-quality images from driving scenarios and provides horizontal bounding box labels for 278,433 instances across 9 classes. Objects are subdivided by area into four ranges: 144 px² or less, 144 to 400, 400 to 1024, and 1024 to 2000, denoted eS, rS, gS, and Normal, respectively. Objects with an area exceeding 2000 px² were excluded from labeling and treated as ignore regions. Following the official devkit, images were split into 800×800 patches with an overlap of 150 px, yielding 56,225 training patches and 109,126 validation patches. The training split excluded patches without objects, whereas the validation split retained all patches, including empty ones for evaluation after merging back to the original images, so the validation split contained more patches than the training split.

For the control experiment that verifies the mechanism, we used VisDrone. Bounding box labels for 10 classes are provided on 6,471 training and 548 validation images, which we organized in two forms: the whole-image and the 640×640 tile forms. The tile form split the same images into 640×640 tiles with an overlap of 128 px, and each bounding box was retained when at least 40% of its area lay inside the tile. The 40% criterion is identical to that of the SODA-D devkit, and the split yielded 44,205 training and 2,842 validation tiles. The whole-image form downsampled the original image to fit the 640×640 input, making objects smaller, whereas in the tile form the same object occupied more pixels.

4.3 Evaluation Metrics and Protocols

We report the COCO-style mAP@0.5 and mAP@0.5:0.95 on the validation split, together with Average Precision (AP) per area range for analysis by object size. Per-class AP is the area under the Precision–Recall curve, and mAP is the average of AP over all classes. The number of seed repetitions was set according to the magnitude of the performance difference between the two configurations being compared. YOLOv11s and the P2-head configuration results are reference results from constructing the baseline, so only seed 0 is reported. The comparison between ZIRF and P2RG, the fusion module comparison, and the controlled tiling experiment involve small differences, so they are reported as the mean and standard deviation over seeds 0 to 2, fixed in advance.

Two evaluation protocols were used together on SODA-D. Internal comparisons and the gate variant ablation are reported with ultralytics validation results computed at the 800×800 patch level, and external comparisons and the size-range analysis are reported with the official SODA-D devkit evaluation. The official evaluation merged patch predictions back to the original image coordinates and computed AP on the validation split with ignore regions excluded. Evaluation in the VisDrone tile form was performed at the tile level without merging back to the original images. Within each form, the performance difference between ZIRF and the baseline was compared under identical evaluation protocols.

Inference latency was measured under a batch size of 1 and FP16. All configurations were measured three times in a single session in round-robin order. Each run took the average of 200 forward passes after 200 warm-up passes, and the reported value is the mean over three runs. Only the model inference time, excluding preprocessing and NMS, is reported. Because the latency in every table was obtained in the same single session, configurations that appear in multiple tables have identical values.

Experimental Results

We first confirm the improvement over the P2RG baseline on SODA-D, then analyze the relationship between object size and gain through a comparison of input forms on the original VisDrone data, and finally present a comparison with existing fusion modules transplanted into the same slot, a cost analysis, component ablation, and analysis of the learned parameters.

5.1 Primary Results on SODA-D

Table 3 shows the results of adding components step by step to YOLOv11s. The upper two rows show seed-0 results, and the lower two rows show the mean and standard deviation over three seeds. On the three-seed mean, $mAP@0.5:0.95$ improved from 0.2756 for P2RG to 0.2822 after ZIRF insertion, a gain of +0.0066, and $mAP@0.5$ improved from 0.5404 to 0.5505, a gain of +0.0101. The single largest contribution came from the high-resolution P2 head, and the additional ZIRF gain was of the same order as the RepGFPN contribution. Considering the cost of each step, the P2 head yielded +0.0110 with 0.17M parameters and 8.0 GFLOPs, RepGFPN yielded +0.0076 with 2.99M parameters and 9.0 GFLOPs, and ZIRF yielded +0.0066 with 0.60M parameters and 5.3 GFLOPs. ZIRF achieved a gain of similar size with one-fifth of the parameters of RepGFPN. The parameters and FLOPs were computed with an 80-class detection head, making them comparable to those of published YOLO models.

Table 3. Cumulative ablation from YOLOv11s on SODA-D patch-level validation

Configuration	Params. (M)	GFLOPs	Latency (ms)	$mAP@0.5$	$mAP@0.5:0.95$
YOLOv11s	9.46	21.7	2.24	0.5238	0.2570
+ P2 head	9.63	29.7	2.62	0.5322	0.2680
+ RepGFPN (P2RG)	12.62	38.7	3.81	0.5404 ± 0.0022	0.2756 ± 0.0011
+ ZIRF (Ours)	13.22	44.0	4.89	0.5505 ± 0.0023	0.2822 ± 0.0013

5.2 Effective Object Size and the Performance Gain

The gain from cross-layer fusion depends on how many pixels an object retains at the 640×640 input. In the size-range results of Table 4, the gain of ZIRF in the eS range at +0.0022 was the smallest and increased with range size to +0.0050 for rS, +0.0059 for gS, and +0.0123 for Normal. Since even the Normal range of SODA-D covers objects of only 32 to 45 px on a side, the increase across ranges shows that the effect of ZIRF varies with pixel count within the small object regime rather than reflecting a transition to normal-sized objects.

Table 4. Size-degree AP on SODA-D with the official evaluation. Increments over P2RG are given in parentheses.

Method	AP_{eS}	AP_{rS}	AP_{gS}	AP_N
P2RG	0.1122 ± 0.0011	0.2318 ± 0.0005	0.3179 ± 0.0007	0.3347 ± 0.0022
+ ZIRF (Ours)	0.1144 ± 0.0027 (+0.0022)	0.2369 ± 0.0011 (+0.0050)	0.3238 ± 0.0011 (+0.0059)	0.3470 ± 0.0028 (+0.0123)

Tiling changes two factors at once. One is the intended variable, the effective object size, and the other is an unintended variable, the sliced inference pipeline. Because the two forms differ even in their evaluation coordinate systems, directly comparing absolute values would mix the two effects. To separate them, we used a 2×2 control design that crossed the whole-image and tile forms with the baseline and ZIRF attachment, and measured the ZIRF gain as the difference between ZIRF and the baseline within each form. Since the baseline and ZIRF shared identical training and evaluation pipelines within each form, the performance difference between the two models was compared under the same conditions.

Both forms performed training and inference in the same form. The whole-image form took one original VisDrone image as a sample and downsampled it to the input size, and since the originals were 1400 to 2000 px wide, the objects shrank to less than half their size in this process. The tile form split the same image into 640×640 tiles with an overlap of 128 px and took one tile as a sample, ensuring that the input resolution matched the original tile size and no downscaling occurred. Inference also followed the training sample unit of each form, and evaluation in the tile form was performed at the tile level without merging back to the original coordinates. Both forms were constructed from the same original VisDrone images and bounding box labels, and the network input size was also identical. However, because tiling changes not only the object size

at the input but also the image extent covered by one sample, we interpret this comparison as an experiment that confirms the association between object size and the gain of ZIRF with dataset differences excluded.

To show how much tiling changes the object size at the input, Fig. 3 shows the cumulative distribution³⁸ of effective object size³⁹, defined as the square root of box area, over the entire validation split for the four data forms. Except for the native-resolution size, the three forms were converted to the input scale. The native-resolution object size of the whole-image and tile forms has the same median of 22.8 px, and the curve of the tile form almost overlaps the native-resolution curve, showing that tiling preserves the original size distribution. When the large images of the whole-image form are downsampled to fit the input, objects shrink to a median of 11.3 px and 31% fall below 8 px, whereas the SODA-D patch distribution and tiling preserve objects at medians of 19.3 px and 23.0 px, respectively, with only about 5% below 8 px. The fractions below 32 px are 92.4% for the whole-image form, 68.2% for the tile form, and 91.5% for SODA-D. Because the whole-image and tile forms use the same VisDrone images, the difference between the two forms arises from the object size difference produced by the input transformation, not from the dataset.

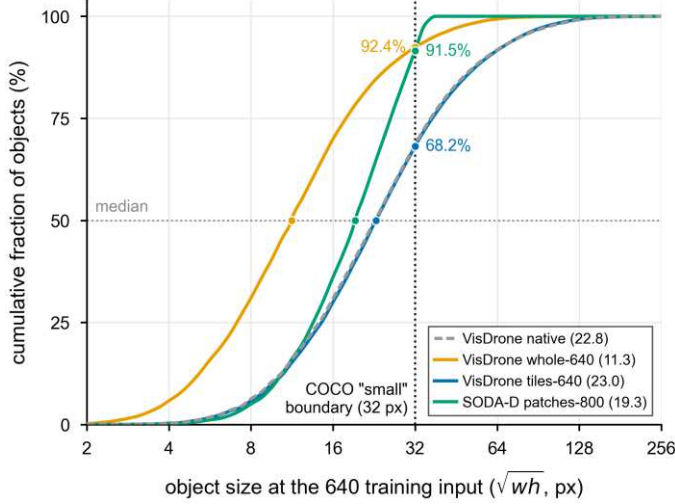


Figure 3. Cumulative distribution of effective object size at the 640×640 training input for the four data forms. Markers indicate the medians and the fractions below the COCO small boundary of 32 px.

Table 5 shows the result of the controlled experiment. In the whole-image form, the average gain of ZIRF, $+0.0025$, is of the same magnitude as the seed standard deviation of ZIRF, ± 0.0017 . Training the same images as tiles, which enlarges the object size at the input from 11.3 px to 23.0 px, raised the average gain to $+0.0036$. The direction agrees with the range-wise increase on SODA-D observed in Table 4. Since the average gain increases in the tile form, where objects remain larger at the input, the gain of ZIRF is related to object size at the input rather than to dataset differences.

Table 5. Control experiment on VisDrone with the P2RG baseline in $\text{mAP}@0.5:0.95$. Increments in parentheses are measured within each form.

Form	Eff. size @640	P2RG	+ ZIRF (Ours)
Whole image	11.3 px	0.2514 ± 0.0007	$0.2539 \pm 0.0017 (+0.0025)$
640 px tile	23.0 px	0.3784 ± 0.0020	$0.3820 \pm 0.0005 (+0.0036)$

5.3 In-slot Comparison of Fusion Modules

To determine whether the ZIRF gain stems from its specific fusion design or is obtained merely by inserting a fusion path after the completed neck, we transplanted existing fusion mechanisms into the four output residual slots occupied by ZIRF and compared the results as shown in Table 6. The compared modules are BiFPN with fast normalized weighted fusion, ASFF with softmax spatial weighted fusion, GFF with gated fusion, and Gold-YOLO with gather-and-distribute injection. Each module reproduced the fusion mechanism of its original paper, checked against the official implementation, and the slot, inputs, and training configuration were kept identical so that only the fusion design was isolated. Because the original designs of the four modules replaced or restructured the neck topology, this comparison was a mechanism-level transplant into the same slot rather than a reproduction of the original architectures.

The results in Table 6 show two things. First, all four compared modules improved performance over P2RG, which means that the residual slot after the completed neck can serve as an integration point for a variety of fusion mechanisms. Second, in terms of accuracy alone, ZIRF was not the top performer. All four modules achieved higher mean accuracy than ZIRF. However, the gap was 0.0009 to 0.0026, less than half of the 0.0066 gain of ZIRF, and the parameter and FLOP increases of ZIRF were the smallest of the five modules. ZIRF secured 72% of the best gain with a parameter increase of only 18% to 47% of that of the four compared modules. This cost advantage is limited to parameters and FLOPs. Latency differed little between modules and did not follow the parameter order, and ZIRF remained in the middle despite having the fewest parameters. In addition, none of the compared modules possesses the property of preserving the baseline output immediately after module insertion through $\gamma = 0$ zero initialization. If the goal in the residual slot is the accuracy ceiling, a heavier fusion is more appropriate, whereas ZIRF is the appropriate choice under cost constraints.

Table 6. In-slot comparison of fusion modules on SODA-D patch-level validation.

Slot module	Params. (M)	GFLOPs	Latency (ms)	mAP@0.5	mAP@0.5:0.95
P2RG	12.62	38.7	3.81	0.5404 ± 0.0022	0.2756 ± 0.0011
BiFPN	13.90 (+1.28)	45.6 (+6.9)	4.86	0.5523 ± 0.0016	0.2834 ± 0.0005
ASFF	13.96 (+1.34)	46.0 (+7.3)	5.06	0.5537 ± 0.0012	0.2843 ± 0.0008
GFF	13.90 (+1.28)	45.6 (+6.9)	5.08	0.5552 ± 0.0016	0.2848 ± 0.0014
Gold-YOLO	15.99 (+3.37)	50.9 (+12.2)	4.78	0.5545 ± 0.0018	0.2831 ± 0.0012
ZIRF (Ours)	13.22 (+0.60)	44.0 (+5.3)	4.89	0.5505 ± 0.0023	0.2822 ± 0.0013

5.4 Efficiency Analysis

The ZIRF cost is regarded as the increment added to the P2RG baseline. Inserting ZIRF adds 0.60M parameters and 5.3 GFLOPs. Looking at the internal distribution of FLOPs, the additional computation is concentrated in projecting the high-resolution P2 feature to each target level. The multiply-accumulate count per target level is approximately 0.14G for the P2 block, 0.31G for P3, 0.69G for P4, and 1.47G for the P5 block, increasing rather than decreasing for lower-resolution targets. This is because the P2 source feature retains its 160×160 grid while being projected to match the wide channel width of low-resolution targets. Therefore, most of the cost lies not in the high-resolution output but in the projection that carries the high-resolution source to low-resolution targets.

Inference latency does not scale with FLOPs. The latency increase from inserting ZIRF is 1.08 ms. When the components are added in turn, the step-wise latency increases are 0.38 ms for the P2 head, 1.19 ms for RepGFPN, and 1.08 ms for ZIRF. Converting these to latency per additional GFLOP gives 0.048, 0.132, and 0.204 ms/GFLOP, respectively. Thus, the ZIRF step has the highest latency per FLOP. The ZIRF step consists of small kernels, resizing, and gating spread across four blocks, and at a batch size of 1, the number of kernel launches is presumed to affect latency more than FLOPs. The latency of ZIRF did not follow the parameter order in the fusion module comparison above for the same reason. Consequently, the FLOP reduction is an advantage in model capacity and memory and does not translate directly into a latency reduction. ZIRF is light in terms of parameters and FLOPs.

5.5 Ablation and Analysis

To isolate the contribution of each component, we compared three variants in the tile form, where the average ZIRF gain is larger, as shown in Table 7. The variants remove the content gate, initialize the residual scalar to $\gamma = 1$, and condition the gate on the sources instead of the target, respectively. All three variants exceeded the baseline by 0.0028 to 0.0042, so the gain through the residual slot remained regardless of the component altered. However, the mean differences between the variants were of the same magnitude as the standard deviations, so the components cannot be ranked. Therefore, the ablation shows not a ranking of the components but that the gain of ZIRF persists under gate removal, a change of initialization, and a change of conditioning direction. This is in the same direction as in Table 6, where all four fusion modules transplanted into the same slot improved performance.

The rationale for adopting zero initialization lies not in final accuracy but in the identity of the initial output, which Equation (3) guarantees structurally. The $\gamma = 1$ variant lacks initial output preservation, yet its mean difference from the final configuration is of the same magnitude as the standard deviation, so zero initialization is a choice that obtains initial output preservation at no cost in accuracy.

Table 7. Robustness of ZIRF to component-level variations on VisDrone in the tile form.

Content gate	Residual init	Gate condition	mAP@0.5:0.95
X			0.3784 ± 0.0020
X	$\gamma = 0$	target	0.3812 ± 0.0007 (+0.0028)
✓	$\gamma = 1$	target	0.3825 ± 0.0010 (+0.0042)
✓	$\gamma = 0$	source	0.3818 ± 0.0014 (+0.0034)
✓	$\gamma = 0$	target	0.3820 ± 0.0005 (+0.0036)

The role of each component becomes clearer in the learned parameters. The final residual scalars per level in Table 8 show that the cross-layer path is strongly activated at the high-resolution outputs where small objects are decoded and remains near identity at the low-resolution outputs. P2 is 0.67, and P3 is 0.54, whereas P4 and P5 are 0.02, consistently across the three seeds. This means that the residual, starting from zero, is selectively activated only at the levels where it is needed. The source contributions also decomposed stably. Half of the signal received by the P2 block comes from the adjacent level P3, and the other half from P4 and P5. No direct connection from P4 and P5 to P2 exists in the PAN topology. This means that the direct path was used in training and secured a contribution comparable to adjacent re-fusion, consistent with the design motivation of ZIRF to directly deliver non-adjacent features.

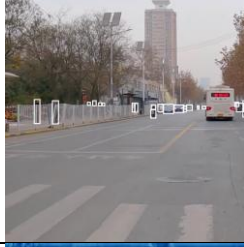
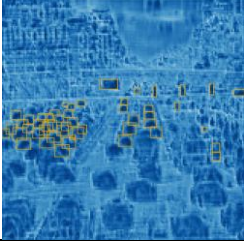
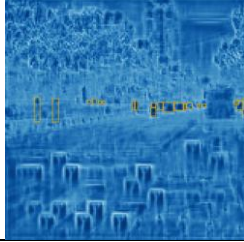
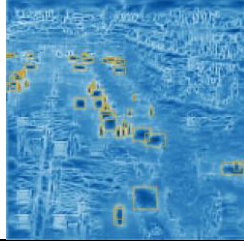
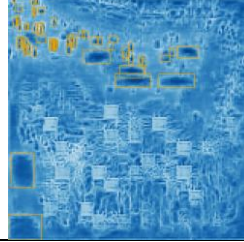
Table 8. Learned ZIRF parameters. Final $|\gamma|$ of the block at each output level and share of each level as a source to P2 block

Output level	Stride	Final $ \gamma $	Source share to P2
--------------	--------	------------------	--------------------

P2	4	0.67 ± 0.05	Target
P3	8	0.54 ± 0.03	0.50 ± 0.01
P4	16	0.02 ± 0.00	0.25 ± 0.01
P5	32	0.02 ± 0.00	0.25 ± 0.01

To examine what the gate responds to, Table 9 overlays ground-truth boxes and gate responses in the same coordinates. The scenes are two SODA-D validation patches, a dense intersection and a sparse road, and two VisDrone validation tiles, a daytime and a nighttime urban scene. The input and ground-truth boxes are placed in the same 640×640 space as the model input, the source-averaged gate map of the P2 block is aligned to the same coordinates, and the gates are extracted from the ZIRF seed-0 model trained on each dataset. The gate learns a spatial modulation that suppresses homogeneous backgrounds such as road surfaces and sky and activates in regions with structure. It also activates on high-contrast road markings such as crosswalks. However, it is not selective to object locations. Aggregating the mean response of the P2-block gate inside and outside ground-truth boxes over the entire validation split of each dataset gave a ratio of 1.04 for SODA-D and 1.10 for VisDrone. The gate responds to the presence of local structure rather than to objectness. The foreground large vehicles in the SODA-D scenes have no ground-truth boxes because SODA-D excludes objects with an area above 2000 px^2 from labeling, whereas in the VisDrone scenes, which are labeled without a size limit, large vehicles also have boxes.

Table 9. Spatial response of the P2-block gate on SODA-D and VisDrone validation scenes.

	SODA-D		VisDrone	
Input				
Gate response				

Conclusion

This paper proposes ZIRF, a lightweight module that compensates for the dilution of P5 semantic information as it passes through adjacent levels of the feature pyramid. ZIRF directly fuses all other pyramid levels into each detection output through a single transformation and adds the result while regulating the contribution of the fusion path with a residual scalar that starts from zero. ZIRF does not replace the RepGFPN neck of the P2RG baseline but augments its outputs, and the detection output immediately after module insertion is identical to that of the baseline. On SODA-D, ZIRF improves the $\text{mAP}@0.5:0.95$ of the P2RG baseline from 0.2756 to 0.2822. In the comparison with existing fusion modules transplanted into the same slot, ZIRF secures 72% of the best gain with a parameter increase of only 18% to 47% of those of the compared modules, at a cost of +0.60M parameters and +5.3 GFLOPs. On VisDrone, the ZIRF gain becomes more pronounced as objects retain more pixels at the 640×640 input. The learned parameters also support the ZIRF design motivation. The residual scalars increase substantially only at the P2 and P3 outputs where small objects are decoded, and half of the signal received by the P2 block comes from the direct paths from P4 and P5 that do not exist in the PAN topology.

This study has three limitations. First, the cost advantage of ZIRF holds in parameters and FLOPs but not in inference latency. Second, validation is limited to two small object benchmarks of drone and road imagery. Third, the gate responds to the presence of local structure rather than to the presence of objects, and how the gate response improves detection performance has not been established. Future work will improve the efficiency of the module implementation, whose latency was high relative to its GFLOPs. We intend to apply ZIRF to general-scene datasets such as COCO and to a range of model scales and resolution settings to determine its scope of applicability, and analyze the relationship between gate response and detection performance.

Data Availability

The datasets analyzed in the current study are available in the SODA-D and VisDrone repositories. SODA-D can be accessed from its official project page at <https://shaunyuan22.github.io/SODA/>, and VisDrone from its official repository at <https://github.com/VisDrone/VisDrone-Dataset>. Both datasets are publicly available and were used in compliance with their

respective licenses for non-commercial research purposes.

References

1. Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2117–2125. doi:10.1109/CVPR.2017.106
2. Liu S, Qi L, Qin H, Shi J, Jia J (2018) Path aggregation network for instance segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 8759–8768. doi:10.1109/CVPR.2018.00913
3. Tan M, Pang R, Le QV (2020) EfficientDet: Scalable and efficient object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 10781–10790. doi:10.1109/CVPR42600.2020.01079
4. Xu X, Jiang Y, Chen W, Huang Y, Zhang Y, Sun X (2022) DAMO-YOLO: A report on real-time object detection design. arXiv preprint arXiv:2211.15444. doi:10.48550/arXiv.2211.15444
5. Khanam R, Hussain M (2024) YOLOv11: An overview of the key architectural enhancements. arXiv preprint arXiv:2410.17725. doi:10.48550/arXiv.2410.17725
6. Liu S, Huang D, Wang Y (2019) Learning spatial fusion for single-shot object detection. arXiv preprint arXiv:1911.09516. doi:10.48550/arXiv.1911.09516
7. Li X, Zhao H, Han L, Tong Y, Tan S, Yang K (2020) Gated fully fusion for semantic segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence, 34(7):11418–11425. doi:10.1609/aaai.v34i07.6805
8. Wang C, He W, Nie Y, Guo J, Liu C, Wang Y, Han K (2023) Gold-YOLO: Efficient object detector via gather-and-distribute mechanism. In: Advances in Neural Information Processing Systems (NeurIPS) 36. doi:10.52202/075280-2224
9. Cheng G, Yuan X, Yao X, Yan K, Zeng Q, Xie X, Han J (2023) Towards large-scale small object detection: Survey and benchmarks. IEEE Transactions on Pattern Analysis and Machine Intelligence 45(11):13467–13488. doi:10.1109/TPAMI.2023.3290594
10. Zhu P, Wen L, Du D, Bian X, Fan H, Hu Q, Ling H (2022) Detection and tracking meet drones challenge. IEEE Transactions on Pattern Analysis and Machine Intelligence 44(11):7380–7399. doi:10.1109/TPAMI.2021.3119563
11. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft COCO: Common objects in context. In: European Conference on Computer Vision (ECCV), LNCS 8693, 740–755. doi:10.1007/978-3-319-10602-1_48
12. Yang F, Fan H, Chu P, Blasch E, Ling H (2019) Clustered object detection in aerial images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 8311–8320. doi:10.1109/ICCV.2019.00840
13. Li C, Yang T, Zhu S, Chen C, Guan S (2020) Density map guided object detection in aerial images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 190–191. doi:10.1109/CVPRW50498.2020.00103
14. Huang Y, Chen J, Huang D (2022) UFPMP-Det: Toward accurate and efficient object detection on drone imagery. In: Proceedings of the AAAI Conference on Artificial Intelligence, 36(1):1026–1033. doi:10.1609/aaai.v36i1.19986
15. Akyon FC, Altinuc SO, Temizel A (2022) Slicing aided hyper inference and fine-tuning for small object detection. In: Proceedings of the IEEE International Conference on Image Processing (ICIP), 966–970. doi:10.1109/ICIP46576.2022.9897990
16. Yang C, Huang Z, Wang N (2022) QueryDet: Cascaded sparse query for accelerating high-resolution small object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 13668–13677. doi:10.1109/CVPR52688.2022.01330
17. Du B, Huang Y, Chen J, Huang D (2023) Adaptive sparse convolutional networks with global context enhancement for faster object detection on drone images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 13435–13444. doi:10.1109/CVPR52729.2023.01291
18. Zhu X, Lyu S, Wang X, Zhao Q (2021) TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2778–2788. doi:10.1109/ICCVW54120.2021.00312
19. Lu L, He D, Deng Z, Liu C, Shen Z (2025) MASF-YOLO: An improved YOLOv11 network for small object detection on drone view. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC). doi:10.1109/SMC58881.2025.11342487
20. Chen D, Zhang L (2024) SL-YOLO: A stronger and lighter drone target detection model. arXiv preprint arXiv:2411.11477. doi:10.48550/arXiv.2411.11477
21. Khalili B, Smyth AW (2024) SOD-YOLOv8: Enhancing YOLOv8 for small object detection in aerial imagery and traffic scenes. Sensors 24(19):6209. doi:10.3390/s24196209
22. Wang J, Xu C, Yang W, Yu L (2021) A normalized Gaussian Wasserstein distance for tiny object detection. arXiv preprint arXiv:2110.13389. doi:10.48550/arXiv.2110.13389
23. Xu C, Wang J, Yang W, Yu H, Yu L, Xia GS (2022) RFLA: Gaussian receptive field based label assignment for tiny object detection. In: European Conference on Computer Vision (ECCV), LNCS 13669, 526–543. doi:10.1007/978-3-031-20077-9_31
24. Yuan X, Cheng G, Yan K, Zeng Q, Han J (2023) Small object detection via coarse-to-fine proposal generation and imitation learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 6317–6327. doi:10.1109/ICCV51070.2023.00581
25. Ghiasi G, Lin TY, Le QV (2019) NAS-FPN: Learning scalable feature pyramid architecture for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 7036–7045. doi:10.1109/CVPR.2019.00720
26. Feng Y, Huang J, Du S, Ying S, Yong JH, Li Y, Ding G, Ji R, Gao Y (2025) Hyper-YOLO: When visual object detection meets hypergraph computation. IEEE Transactions on Pattern Analysis and Machine Intelligence 47(4):2388–2401. doi:10.1109/TPAMI.2024.3524377
27. Du Z, Hu Z, Zhao G, Jin Y, Ma H (2025) Cross-layer feature pyramid transformer for small object detection in aerial images. IEEE Transactions on Geoscience and Remote Sensing 63:5625714. doi:10.1109/TGRS.2025.3572706
28. Guo C, Fan B, Zhang Q, Xiang S, Pan C (2020) AugFPN: Improving multi-scale feature learning for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 12595–12604.

doi:10.1109/CVPR42600.2020.01261

29. He T, Zhang Z, Zhang H, Zhang Z, Xie J, Li M (2019) Bag of tricks for image classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 558–567. doi:10.1109/CVPR.2019.00065
30. Goyal P, Dollár P, Girshick R, Noordhuis P, Wesolowski L, Kyrola A, Tulloch A, Jia Y, He K (2017) Accurate, large minibatch SGD: Training ImageNet in 1 hour. arXiv preprint arXiv:1706.02677. doi:10.48550/arXiv.1706.02677
31. Zhang H, Dauphin YN, Ma T (2019) Fixup initialization: Residual learning without normalization. In: International Conference on Learning Representations (ICLR).
32. Bachlechner T, Majumder BP, Mao H, Cottrell G, McAuley J (2021) ReZero is all you need: Fast convergence at large depth. In: Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI), PMLR 161:1352–1361.
33. De S, Smith SL (2020) Batch normalization biases residual blocks towards the identity function in deep networks. In: Advances in Neural Information Processing Systems (NeurIPS) 33, 19964–19975.
34. Touvron H, Cord M, Sablayrolles A, Synnaeve G, Jégou H (2021) Going deeper with image transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 32–42. doi:10.1109/ICCV48922.2021.00010
35. Zhang L, Rao A, Agrawala M (2023) Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 3836–3847. doi:10.1109/ICCV51070.2023.00355
36. Jocher G, Qiu J (2026) Ultralytics YOLO, version 8.4.42. <https://github.com/ultralytics/ultralytics>. Accessed 3 Sep 2026
37. Jordan K, Jin Y, Boza V, You J, Cesista F, Newhouse L, Bernstein J (2024) Muon: An optimizer for hidden layers in neural networks. <https://kellerjordan.github.io/posts/muon/>. Accessed 3 Sep 2026
38. Singh B, Davis LS (2018) An analysis of scale invariance in object detection: SNIP. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 3578–3587. doi:10.1109/CVPR.2018.00377
39. Yu X, Gong Y, Jiang N, Ye Q, Han Z (2020) Scale match for tiny person detection. In: Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), 1257–1265. doi:10.1109/WACV45572.2020.9093394

Acknowledgement

This work was supported by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) and Korea Smart Farm R&D Foundation (KosFarm) through the Smart Farm Innovation Technology Development Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA) and the Ministry of Science and ICT (MSIT), Rural Development Administration (RDA)(RS-2025-02306946)

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korean government (MSIT)(IITP-2026-RS-2020-II201489)

Funding

This work was supported by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) and Korea Smart Farm R&D Foundation (KosFarm) through the Smart Farm Innovation Technology Development Program, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA) and the Ministry of Science and ICT (MSIT), Rural Development Administration (RDA)(RS-2025-02306946)

Author Contributions Statement

S.M.K. conceived ZIRF, designed the zero-initialized residual fusion structure, implemented the training pipeline, and conducted the experiments on the SODA-D and VisDrone datasets. S.H.C. and S.W.P. contributed to the experiments and analysis. C.B.S. and S.H.J. supervised the project and contributed to manuscript revision. All authors have reviewed and approved the final manuscript.

Corresponding Author

Correspondence to Chun-Bo Sim and Se-Hoon Jung.

Competing Interests

The authors declare no competing interests.

Informed Consent

This study uses two publicly available datasets, SODA-D and VisDrone. These datasets consist of images collected and released by their respective authors, for which no additional informed consent is required. All datasets were used in accordance with their terms and conditions, which permit their use for research purposes and in academic publications.

Additional Information

Correspondence and requests for materials should be addressed to C.B.S. and S.H.J.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.