Mech. Sci., 17, 371–379, 2026
https://doi.org/10.5194/ms-17-371-2026

Mechanical Sciences
Open Access

RIL-YOLO: a lightweight real-time object detection model on mobile devices for kart racing

Ang Sha[1], Fuen Xue[2], Yong Zhang[2], Xiaolin Zang[3], and Jinying Zhao[4]

[1]Innovation Institute for Sustainable Maritime Architecture Research and Technology, Qingdao University of Technology, Qingdao, China
[2]School of Information and Control Engineering, Qingdao University of Technology, Qingdao, China
[3]School of Architecture and Urban Planning, Qingdao University of Technology, Qingdao, China
[4]Qingdao Cyber Racing Technology Co., Ltd., Qingdao, China

Correspondence: Xiaolin Zang (zangxiaolin@qut.edu.cn)

Received: 12 January 2026 – Revised: 12 March 2026 – Accepted: 25 March 2026 – Published: 9 April 2026

Abstract. The kart is a high-speed mechanical system that requires real-time and reliable visual perception, while motion blur, occlusion, and limited mobile computing resources pose significant challenges. To address these issues, we propose RIL-YOLO, a lightweight object detection framework based on YOLOv8 and optimized for mobile deployment. The method incorporates motion blur data augmentation, a re-parameterized shared convolutional detection head architecture, an inner-CIoU (complete intersection over union) loss, and LAMP (layer-adaptive sparsity for the magnitude-based pruning)-based pruning to improve robustness, localization accuracy, and inference efficiency. Experimental results show that, compared with YOLOv8n, RIL-YOLO improves mAP@0.5 and mAP@[0.5 : 0.95] by 2.8 % and 2.2 %, respectively, while reducing parameters by 83 %, lowering FLOPs (floating point operations per second) by 53 %, and increasing inference speed by approximately 25 %. The proposed method achieves a favorable balance between accuracy and real-time performance on resource-constrained mobile platforms.

1 Introduction

The kart is a high-speed mechanical system for racing. It imposes stringent requirements on real-time visual perception for applications including safety monitoring and race analysis (Anggrainy et al., 2024; Matsumura et al., 2011). In these scenarios, severe motion blur, rapid scale variations, frequent occlusion, and complex backgrounds significantly increase detection difficulty, while deployment on mobile or edge devices further constrains model complexity and computational cost. Achieving an effective balance between accuracy, real-time performance, and lightweight deployment therefore remains challenging in high-speed dynamic environments.

In recent years, visual perception has also become a core component of autonomous driving systems, where real-time object detection supports environment understanding, obstacle avoidance, and decision-making. Autonomous vehicles must operate reliably under dynamic conditions such as high-speed motion, illumination changes, occlusion, and motion blur while increasingly relying on on-board or edge computing platforms with limited computational resources. Therefore, improving detection robustness and efficiency under resource constraints has become an important research direction in intelligent transportation systems.

Among existing vision-based approaches, the YOLO (you only look once) family has been widely adopted due to its end-to-end design and high inference efficiency. Since its introduction by Redmon et al. (2016), the YOLO family has continuously evolved, achieving improved accuracy and speed. YOLOv8 further adopts an anchor-free paradigm and a decoupled detection head, enabling a favorable trade-off between accuracy and speed. However, YOLOv8 still exhibits degraded robustness under motion blur and limited real-time performance under strict resource constraints.

Recent studies have explored lightweight optimization of YOLO-based detectors through detection head redesign, la-

bel assignment optimization, and model compression. Chen et al. (2025) introduced a hierarchical decoupled prediction head and dynamic label assignment to enable low-latency detection of dense and occluded pedestrians on edge devices, while Lu and Liu (2022) combined attention mechanisms, depth-wise separable convolutions, and channel pruning to achieve efficient real-time detection in industrial environments, and Ren et al. (2024) improved YOLOv8 for autonomous driving by incorporating attention mechanisms and an enhanced IoU-based loss to boost road object detection performance. Although effective, these methods are not specifically designed for high-speed motion scenarios and are rarely validated on real mobile devices.

To address these issues, we propose RIL-YOLO, a lightweight object detection framework optimized for high-speed dynamic scenarios and mobile deployment. The acronym "RIL" represents the three core improvements introduced in this work: the re-parameterized shared convolutional detection head (RSCD), the inner-CIoU (complete intersection over union) loss, and the LAMP (layer-adaptive sparsity for the magnitude-based pruning) pruning strategy. Experimental results based on a kart-racing dataset and real mobile devices demonstrate that RIL-YOLO effectively balances detection accuracy and real-time efficiency under resource-constrained conditions.

2 Materials and methods

YOLOv8 is an object detection model framework released by Ultralytics in January 2023. Its architecture consists of three main components: a backbone network, a neck network, and a detection head, as illustrated in Fig. 1. The backbone is built upon a CSPDarknet53-based architecture and incorporates C2f and SPPF (spatial pyramid pooling–fast) modules to enhance feature extraction capability and multi-scale information aggregation. The neck network combines feature pyramid network (FPN) and path aggregation network (PANet) structures to achieve bidirectional fusion of multi-scale features, thereby improving detection performance for objects of varying sizes. The detection head adopts an anchor-free design with a decoupled structure, in which classification and regression tasks are handled separately, leading to improved detection accuracy and training stability.

2.1 Improved YOLOv8

2.1.1 Data augmentation

We adopt a combination of online and offline data augmentation strategies to enhance the generalization capability of the model. The online augmentation techniques employed in this work include hue adjustment, saturation adjustment, brightness adjustment, image rotation, image translation, image scaling, and mosaic augmentation.

Offline data augmentation is conducted prior to training by preprocessing the dataset and generating augmented images that are stored locally (Kaur et al., 2021). Its primary objective is to simulate motion blur caused by camera shake or high-speed target motion in real kart-racing scenarios. Such motion blur often leads to indistinct object boundaries and loss of fine-grained details, thereby increasing detection difficulty. To address this issue, motion-blur-augmented samples are incorporated into the training set, improving the robustness and recognition performance of the model on blurred images.

2.1.2 Rep shared convolutional detection head

In YOLOv8, the detection head processes multi-scale feature maps using three independent decoupled branches, which introduces considerable parameter redundancy and limits inference efficiency on resource-constrained devices.

To address this issue, we propose a lightweight shared convolutional detection head (LSCD), which shares convolutional layers across different feature scales to reduce parameters and computational cost (Wang et al., 2024c). As illustrated in Fig. 2, feature maps from P3, P4, and P5 are first aligned by a 1×1 convolution, followed by two shared 3×3 convolutional layers for joint feature extraction. The shared features are then fed into separate classification and regression branches, where a scale layer preserves scale-specific adaptability.

To further enhance representation capability without increasing inference overhead, a diverse branch block (DBB) is incorporated into the shared convolutional layers. DBB introduces multi-branch structures during training to enrich feature representation and is re-parameterized into a single convolution during inference (Ding et al., 2021). The resulting re-parameterized shared convolutional detection head (RSCD) achieves improved detection performance while maintaining high inference efficiency, as shown in Figs. 3 and 4.

2.1.3 Improved loss function

Bounding-box regression plays a key role in detection accuracy. In YOLOv8, localization is optimized using CIoU loss, which combines overlap, center distance, and aspect ratio penalties. However, CIoU may converge slowly and shows limited adaptability because gradient updates for high-IoU and low-IoU samples are often unbalanced. The calculation formula is shown in Eq. (1):

$$L_{\mathrm{CIoU}} = 1 - \mathrm{IoU} + \frac{\rho^2(b, b^{\mathrm{g}})}{c^2} + \alpha\upsilon, \tag{1}$$

where IoU represents the intersection-over-union between the predicted box and the ground-truth box, measuring the degree of overlap between the two boxes; $\rho^2(b, b^{\mathrm{g}})$ is the squared Euclidean distance between the center point b of the

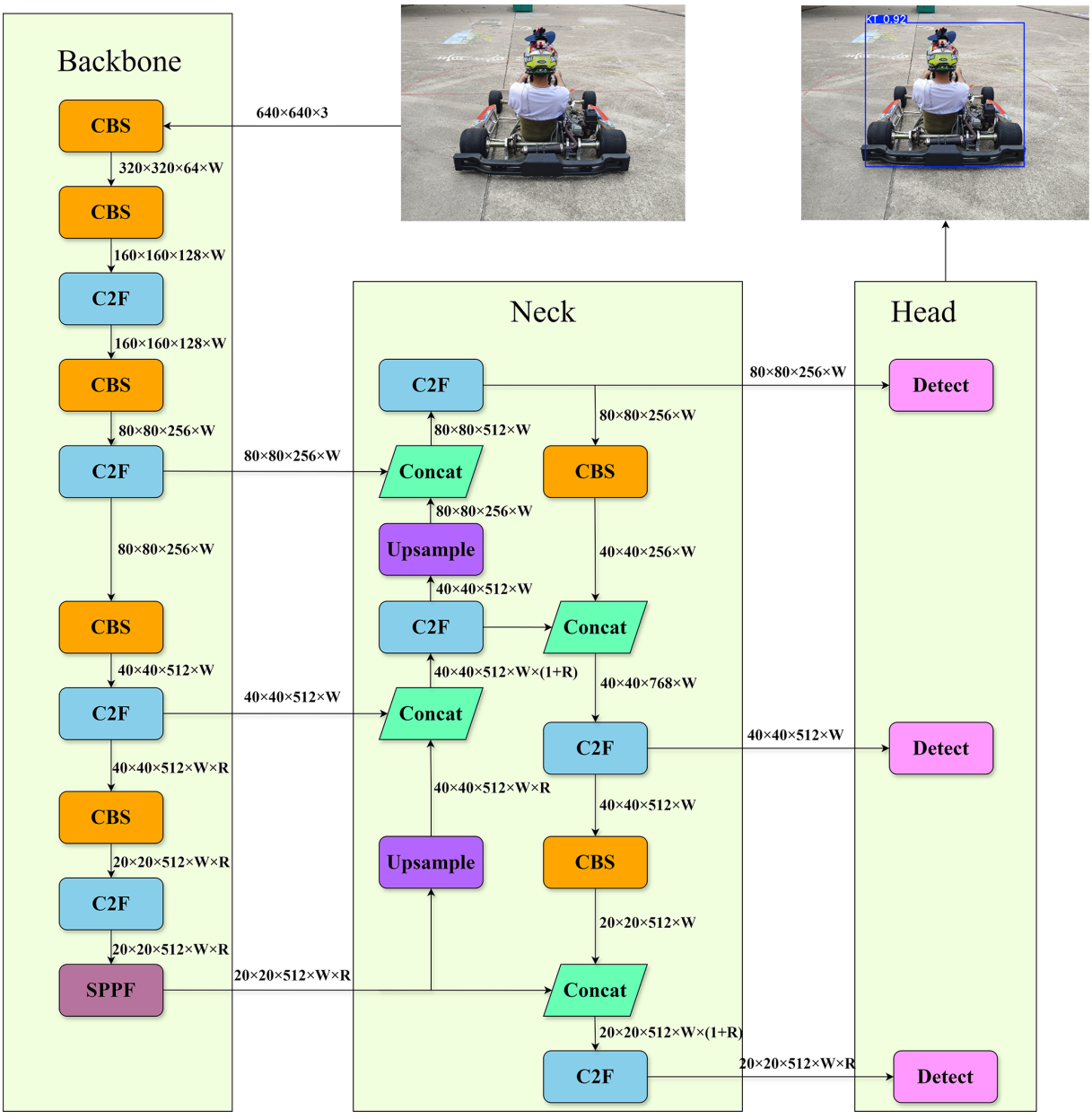

Figure 1. Structure of YOLOv8.

predicted box and the center point b^{g} of the ground-truth box; c is the diagonal length of the smallest enclosing rectangle covering both the predicted box and the ground-truth box; α is a weighting factor used to balance the consistency between IoU and aspect ratio; and υ is a penalty term used to measure the consistency of width-to-height ratios.

To improve convergence and generalization, we introduce inner-IoU, where auxiliary bounding boxes are generated by scaling the original predicted and ground-truth boxes with a factor ratio (Zhang et al., 2023). Specifically, smaller auxiliary boxes are used for high-IoU samples to strengthen gradients, while larger auxiliary boxes are used for low-IoU samples to enlarge the effective optimization region and avoid vanishing gradients. Intuitively, inner-IoU improves gradient stability by adaptively adjusting the effective overlap region used for regression. For high-IoU samples, shrinking the auxiliary bounding boxes increases the sensitivity of the loss to small localization errors, preventing gradient vanishing. For low-IoU samples, enlarging the auxiliary region smooths the optimization landscape and avoids excessively large or unstable gradient updates. As a result, gradient magnitudes become more balanced across samples of different qualities, leading to more stable convergence during training. Replacing IoU in CIoU with inner-IoU yields the proposed inner-

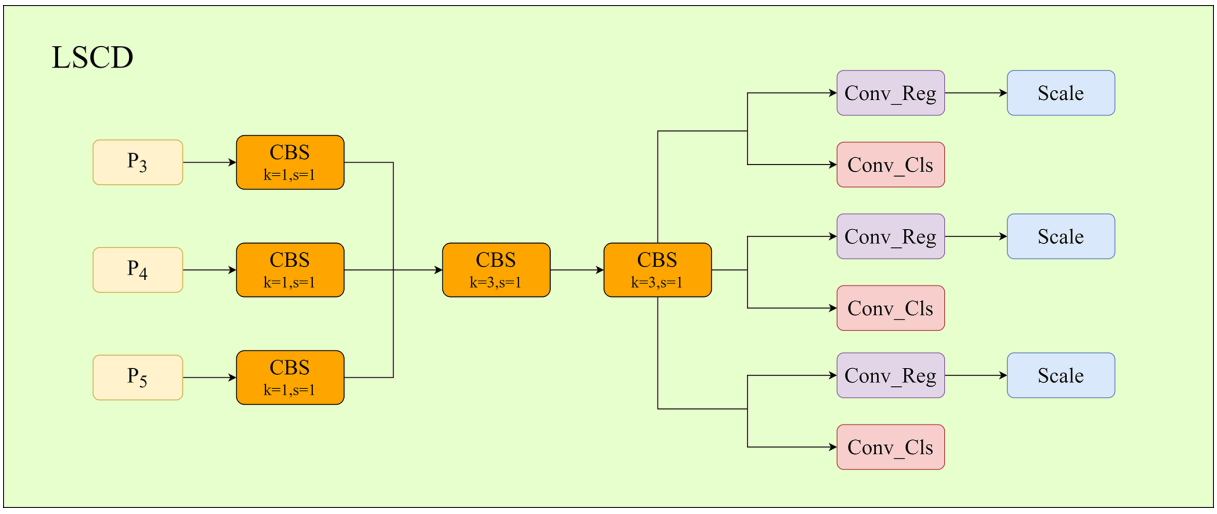

Figure 2. LSCD module.

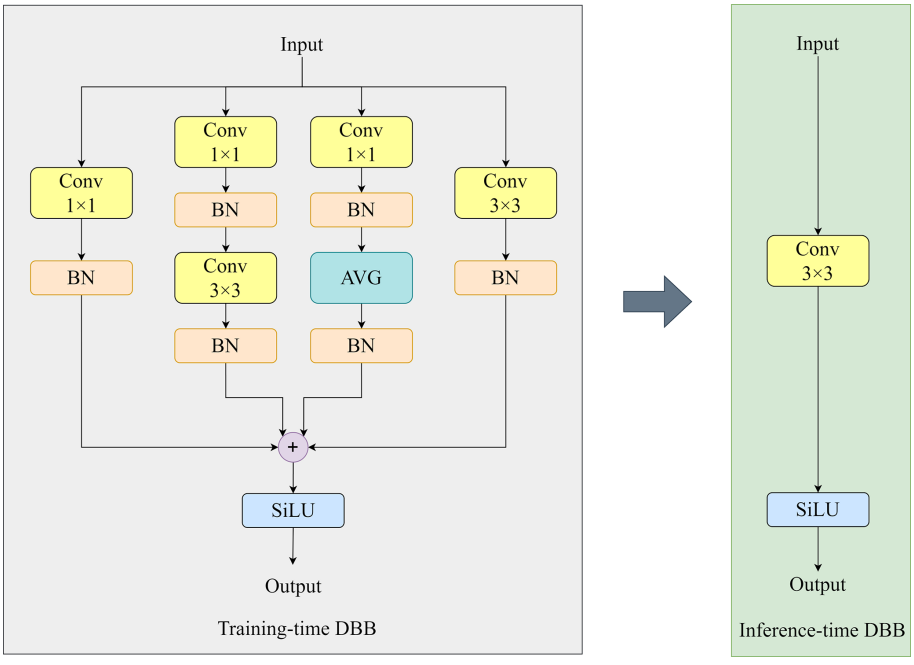

Figure 3. DBB module.

CIoU loss, leading to more stable training and improved localization performance. The mechanism of inner-IoU is illustrated in Fig. 5, and the calculation formulas are shown as follows.

$$b_l^g = x_c^g - \frac{w^g \times \text{ratio}}{2}, \quad b_r^g = x_c^g + \frac{w^g \times \text{ratio}}{2} \tag{2}$$

$$b_t^g = y_c^g - \frac{h^g \times \text{ratio}}{2}, \quad b_b^g = y_c^g + \frac{h^g \times \text{ratio}}{2} \tag{3}$$

$$b_l = x_c - \frac{w \times \text{ratio}}{2}, \quad b_r = x_c + \frac{w \times \text{ratio}}{2} \tag{4}$$

$$b_t = y_c - \frac{h \times \text{ratio}}{2}, \quad b_b = y_c + \frac{h \times \text{ratio}}{2} \tag{5}$$

$$\begin{aligned} \text{inter} = &(\min(b_r^g b_r) - \max(b_l^g b_r)) \\ &\times (\min(b_b^g b_b) - \max(b_t^g b_t)) \end{aligned} \tag{6}$$

$$\begin{aligned} \text{union} = &(w^g \times h^g) \times (\text{ratio})^2 + (w \times h) \\ &\times (\text{ratio})^2 - \text{inter} \end{aligned} \tag{7}$$

$$\text{IoU}^{\text{inner}} = \frac{\text{inter}}{\text{union}} \tag{8}$$

$$L_{\text{inner-CIoU}} = 1 - \text{IoU}^{\text{inner}} + \frac{\rho^2(b, b^{\text{g}})}{c^2} + \alpha v \tag{9}$$

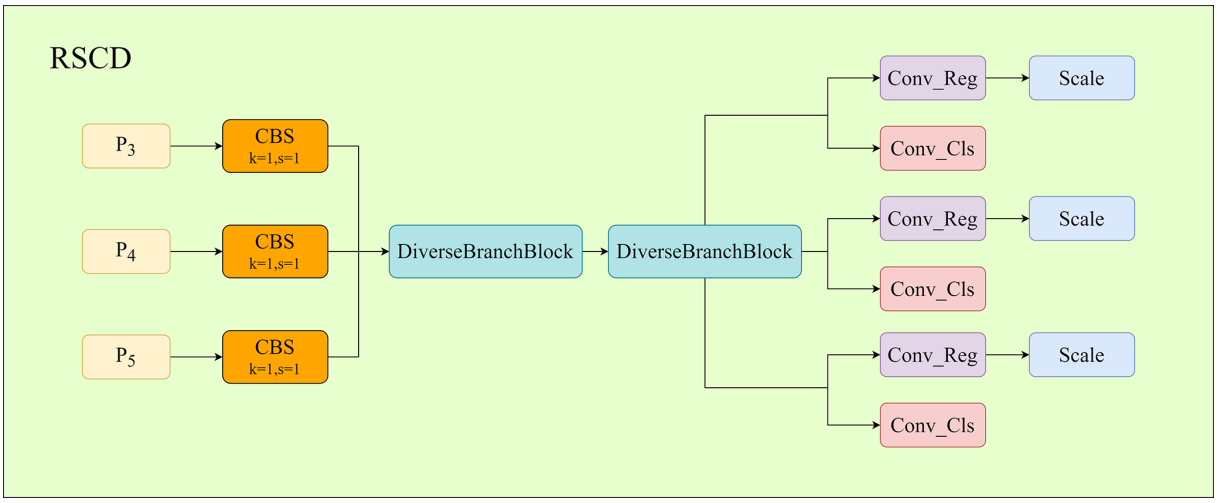

Figure 4. RSCD module.

2.1.4 Model pruning using the LAMP algorithm

To enhance the performance of YOLOv8n in the go-kart detection task while reducing computational cost, we introduce an adaptive pruning strategy known as layer-adaptive sparsity for the magnitude-based pruning (LAMP). LAMP dynamically prunes parameters across different layers based on layer-wise characteristics and importance instead of applying a uniform global pruning ratio (Lee et al., 2020). The pruning ratio of each layer is determined according to its contribution to overall model performance, enabling effective model compression while preserving detection accuracy.

LAMP proposes a novel global pruning-importance scoring method, which quantitatively measures the contribution of each layer to the network. In the LAMP, each layer of the model is first analyzed to evaluate the importance of its parameters and neurons. The formulation is presented as Eq. (10). Subsequently, the weights are ranked according to their LAMP scores, and pruning begins with those having the lowest scores. After each pruning iteration, the LAMP scores of the remaining weights are recalculated to ensure that parameters contributing less to model performance are progressively removed. This iterative process continues until the weights in each layer are compressed to a predefined pruning ratio.

$$\text{Score}(u; W) = \frac{(W[u])^2}{\sum_{v \geq u}(W[v])^2} \tag{10}$$

In the above, $W[u]$ denotes the weight of the uth connection. The numerator represents the square of the connection weight, while the denominator corresponds to the sum of the squared weights of all connections with lower importance.

2.2 Mobile deployment

To meet the application requirements of real-time monitoring and object detection on mobile platforms in high-speed dynamic scenarios, the optimized YOLOv8 model is deployed on a smartphone platform for validation. Considering the limited computational resources of mobile devices, the model is optimized using the NCNN (neural computing neural network) framework in combination with model quantization techniques. NCNN is a high-performance open-source neural network inference framework specifically designed for low-power and resource-constrained devices, providing fast and low-latency inference across multiple platforms through optimized memory management and computational pipelines. Ultimately, the optimized YOLOv8 model is successfully deployed on mobile devices, enabling a real-time monitoring application tailored for kart-racing scenarios.

3 Results and discussions

3.1 Dataset

To evaluate the performance of the proposed method in high-speed dynamic scenarios, a kart-racing object detection dataset was constructed using both online image collection and on-site video recordings. Key frames were extracted from recorded videos, and the final dataset consists of 2337 images. The constructed kart-racing dataset will be made available upon reasonable request to the corresponding author for research purposes.

All images were manually annotated using the LabelImg tool, covering nine object categories relevant to kart-racing scenes: whole kart body (KT), front head of the ego kart (KHF), left apex lane marking (ApexL), finish line (FL), right apex lane marking (ApexR), approaching kart facing

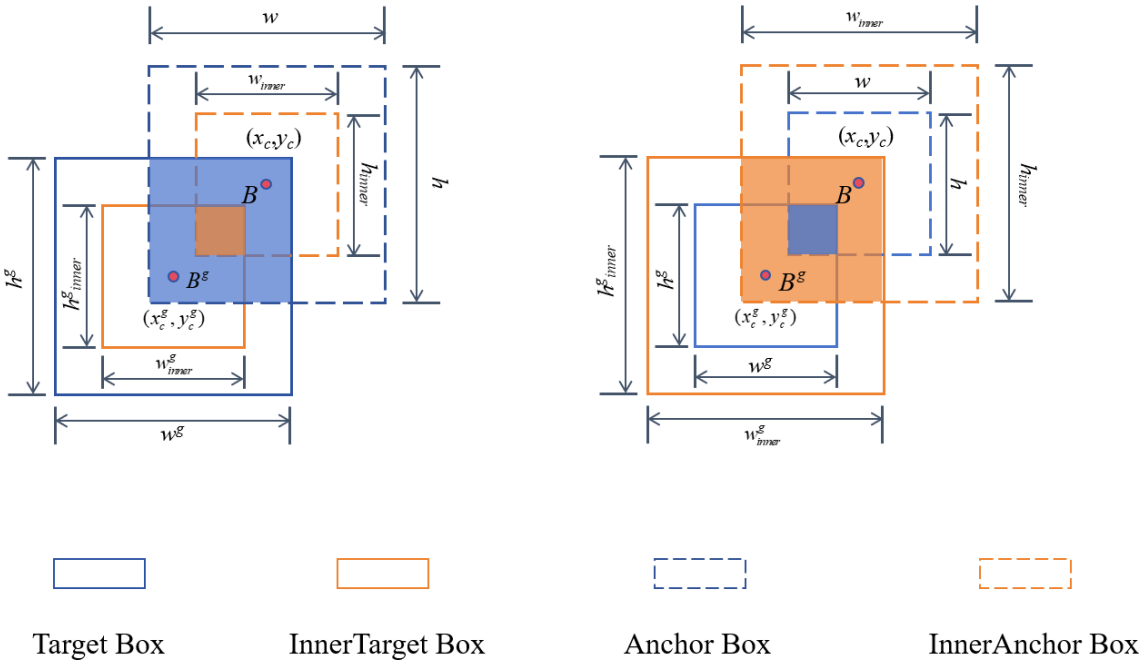

Figure 5. Inner-IoU schematic.

Table 1. Experimental environment configuration.

Device	Experimental environment	Version
PC	Operating system	Windows11
	CPU	Intel i7–14700K
	GPU	Nvidia RTX 4060ti 16G
	Memory	64G
Smartphone	SOC	Snapdragon 8 Gen 3

the camera (KHInv), half kart head on the left side (KHL), half kart head on the right side (KHR), pedestrian (P), and gantry structure (LMJ).

The distribution of samples across categories is moderately imbalanced. The whole kart body (KT) and ego front head (KHF) contain relatively more instances due to their frequent occurrence in racing scenarios, while other categories such as pedestrians and gantry structures appear less frequently but remain sufficiently represented for training and evaluation.

The dataset includes images captured under different lighting conditions, primarily during daytime and late afternoon, thereby introducing moderate illumination variation.

The dataset was divided into training, validation, and test sets with a ratio of 8 : 1 : 1.

3.2 Experimental environment configuration

The experimental environment required is shown in Table 1.

3.3 Experimental parameter

All models were trained using the SGD (stochastic gradient descent) optimizer with an initial learning rate of 0.01, a momentum of 0.937, and a weight decay of 0.01. A cosine annealing learning-rate schedule was adopted during training. Input images were resized to 640×640, and training was conducted for 150 epochs with a batch size of 64. All models were trained from scratch without using pre-trained weights to ensure fair comparison across different configurations.

3.4 Evaluation metrics

To comprehensively evaluate the performance of the RIL-YOLO model, multiple evaluation metrics were adopted. These include precision (P), recall (R), F1 score, mAP, Params, FLOPs (floating point operations per second), and FPS. Precision, recall, and F1 score evaluate classification performance, while mAP measures overall detection accuracy by computing the area under the precision–recall curve. Params represent the total number of learnable parameters in the model, reflecting model size and memory requirements. FLOPs indicate the computational complexity required for a single forward pass. FPS evaluates inference efficiency and real-time performance. Their calculation formulas are given as follows:

$$P = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, R = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{11}$$

$$\mathrm{F1} = 2 \times \frac{P \times R}{P+R}, \mathrm{mAP} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{AP}_i, \tag{12}$$

$$\mathrm{FPS} = \frac{1}{T_{\mathrm{inf}}}, \tag{13}$$

where TP, FP, and FN are the true positive, false positive, and false negative, respectively; AP is numerically equal to the area enclosed by the $P-R$ curve and the coordinate axis; and T_{inf} is the average inference time per image. FPS values were measured on the mobile device by continuously processing 500 frames. The reported FPS corresponds to the mean ± standard deviation across repeated measurements. Other metrics were obtained on the PC platform.

3.5 Ablation experiment

To systematically analyze the contribution of each improvement, all optimization modules were incrementally integrated into the experiments. The experimental results are presented in Table 2, which provides a detailed comparison of the performance gains contributed by each module. Here, A, B, C, and D denote motion blur augmentation, inner-CIoU loss, the RSCD head, and LAMP pruning(ratio = 1.5), respectively.

The experimental results indicate that motion blur augmentation enhances robustness in relation to dynamic scenes, achieving a recall of 90.7 % when used alone. When combined with inner-CIoU and RSCD, precision and recall become more balanced, leading to a notable improvement in mAP@0.5. Inner-CIoU alone increases precision to 90.3 % but reduces recall to 85.1 %; however, introducing RSCD restores recall to above 88 % while maintaining high mAP@0.5, highlighting the importance of coordination between localization optimization and detection head design. RSCD further reduces model parameters by 21.6 % with minimal performance degradation, providing a solid basis for lightweight deployment. After applying LAMP pruning, the parameter count is further reduced to 0.51 M, resulting in a 25 % increase in inference speed.

The combination of RSCD and inner-CIoU exhibits a synergistic effect. RSCD focuses on structural optimization by sharing convolutional parameters across feature scales, which improves feature consistency and reduces overfitting under limited model capacity. In contrast, Inner-CIoU addresses optimization dynamics by balancing gradient magnitudes for high- and low-quality samples. When combined, RSCD provides stronger and more compact feature representations, while inner-CIoU ensures stable and balanced regression updates. This complementary interaction leads to more robust convergence and improved overall performance.

3.6 Comparative experiment

To validate the overall performance advantages of the proposed improved algorithm, several representative lightweight models were selected for comparison, including YOLOv8-fastnet, YOLOv8-mobilenetv4, YOLOv5n, YOLOv9t (Wang et al., 2024b), YOLOv10n (Wang et al., 2024a), and YOLOv11n. Among them, YOLOv8-fastnet and YOLOv8-mobilenetv4 replace the original backbone of YOLOv8 with FastNet (Chen et al., 2023) and MobileNetV4 (Qin et al., 2024), respectively. Since the proposed method is built upon the YOLOv8 framework and primarily targets lightweight real-time deployment on mobile devices, the comparative analysis focuses on representative lightweight one-stage detectors. Two-stage detectors (e.g., faster R-CNN) are generally more computationally intensive and less suitable for mobile real-time applications and therefore were not included in this comparison. For fairness, all comparison models, including YOLOv11n, were implemented using the official Ultralytics implementation (version 8.3.50). The comparative experimental results are summarized in Table 3.

The experimental results demonstrate that our method outperforms all comparison models on both mAP@0.5 and mAP@[0.5 : 0.95], achieving improvements of 1.2 % and 2.1 % over the second-best model, YOLOv9t, respectively, which verifies its strong adaptability to complex scenarios. With respect to model efficiency, the parameter count of our method is only 25.9 % of that of YOLOv9t, while the FLOPs are reduced by 39.7 % compared with YOLOv11n, demonstrating an effective balance between high accuracy and low computational cost.

4 Conclusions

We propose RIL-YOLO, a lightweight object detection method for real-time visual perception in high-speed mechanical systems, using kart racing as a representative dynamic scenario. Based on the YOLOv8 framework, the model is systematically optimized through motion blur data augmentation, an RSCD head, an inner-CIoU loss function, and LAMP-based pruning. Experimental results show that, relative to the YOLOv8n baseline, the optimized model improves mAP@0.5 and mAP@[0.5 : 0.95] by 2.8 % and 2.2 %, respectively, while reducing parameters by approximately 83 %, lowering computational cost by about 53 %, and increasing mobile inference speed by around 25 %, achieving a favorable balance between accuracy and real-time performance on resource-constrained platforms.

Furthermore, in practical mobile deployment scenarios, reduced model complexity contributes not only to faster inference but also to lower power consumption and improved thermal stability during prolonged operation. By alleviating computational load, the proposed method helps maintain stable latency and mitigate performance degradation caused by thermal throttling on smartphones or embedded devices. During practical testing under continuous real-time monitoring, no obvious degradation in inference speed was observed, indicating stable runtime behavior under typical operating conditions. These characteristics enhance its suitability for continuous real-time monitoring applications in high-speed dynamic environments.

Table 2. Ablation experiment.

Model	P (%)	R (%)	F1 (%)	mAP@0.5 (%)	mAP@[0.5 : 0.95] (%)	Params (M)	FLOPs (G)	FPS (mean ± SD)
YOLOv8n	83.4	90.7	86.9	90.9	58.8	3.01	8.1	19.6 ± 1.7
$+A$	90.3	85.1	87.6	91.4	59.9	3.01	8.1	19.1 ± 1.0 (−2 %)
$+B$	88.9	86.4	87.6	93.8	58	3.01	8.1	19.9 ± 0.9 (+2 %)
$+C$	88.8	82.9	85.7	88.7	57.1	2.36	6.5	21.8 ± 0.6 (+12 %)
$+A+B$	86	90.4	88.1	92.4	60.9	3.01	8.1	19.6 ± 0.8 (0 %)
$+A+C$	83.4	89.7	86.4	90.1	60	2.36	6.5	20.9 ± 0.8 (+7 %)
$+B+C$	88	83.5	85.7	91.4	55.9	2.36	6.5	21.1 ± 1.2 (+8 %)
$+A+B+C$	85.1	88.1	86.6	90.4	58.1	2.36	6.5	21.4 ± 1.3 (+9 %)
$+A+B+C+D$	85.6	88.2	86.9	93.7	61	0.51	3.8	24.4 ± 0.5 (+25 %)

Table 3. Comparative experiment.

Model	P (%)	R (%)	F1 (%)	mAP@0.5 (%)	mAP@[0.5 : 0.95] (%)	Params (M)	FLOPs (G)
YOLOv8-mobilenetv4	80.9	84.9	82.9	84	51.6	5.70	22.6
YOLOv8-fastnet	87.7	84.3	86.0	88.7	56.7	4.17	10.7
YOLOv5n	85.5	80.7	83.0	88.1	56.2	2.50	7.1
YOLOv9t	88.9	88	88.4	92.5	58.9	1.97	7.6
YOLOv10n	75.4	77.1	76.2	85.7	53.9	2.70	8.2
YOLOv11n	86.5	85.4	85.9	92.4	57.6	2.58	6.3
RIL-YOLO(ours)	85.6	88.2	86.9	93.7	61	0.51	3.8

Despite the demonstrated improvements, several limitations remain. First, the dataset size is relatively limited compared to large-scale benchmarks, which may restrict generalization to more diverse environments. Second, although mobile inference stability was verified under typical operating conditions, comprehensive long-duration thermal and power consumption analysis was not systematically conducted. In addition, extreme lighting variations or highly crowded racing scenarios may still pose challenges to detection robustness. Future work will focus on expanding the dataset diversity and conducting more extensive real-world deployment evaluations.

Code and data availability. All the code and data used in this paper can be obtained upon request to the corresponding author.

Author contributions. AS conceived the study; provided the key ideas; and performed the method design, the data collection, the paper revision, and the supervision. FX designed the RIL-YOLO and constructed the paper. YZ provided the test program. XZ provided the key suggestions on the experiments. JZ provided the key suggestions on the research questions.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Financial support. This research has been supported by the National Natural Science Foundation of China (grant no. 42301266) and the Natural Science Foundation of Shandong Province (grant no. ZR2023QD005).

Review statement. This paper was edited by Benliang Zhu and reviewed by Marco Aiello and two anonymous referees.

References

Anggrainy, R., Yoga, N. G., Wiyono, A., Putri, D. M., Wahyudi, Z. T., and Septiyan, Y. A.: Unveiling the Future of Safety: Cutting-Edge Simulation Testing of e-Kart Bumpers, J. Phys. Conf. Ser., 012096, https://doi.org/10.1088/1742-6596/2866/1/012096, 2024.

Chen, C., Li, J., Shuai, Z., Wang, Y., and Wang, Y.: A lightweight optimization framework for real-time pedestrian detection in dense and occluded scenes, Mech. Sci., 16, 877–886, https://doi.org/10.5194/ms-16-877-2025, 2025.

Chen, J., Kao, S.-h., He, H., Zhuo, W., Wen, S., Lee, C.-H., and Chan, S.-H. G.: Run, don't walk: chasing higher FLOPS for faster neural networks, arXiv [preprint], https://doi.org/10.48550/arXiv.2303.03667, 2023.

Ding, X., Zhang, X., Han, J., and Ding, G.: Diverse branch block: Building a convolution as an inception-like unit, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021, IEEE, 10881–10890, https://doi.org/10.1109/CVPR46437.2021.01074, 2021.

Kaur, P., Khehra, B. S., and Mavi, E. B. S.: Data augmentation for object detection: A review, in: 2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), 537–543, https://doi.org/10.1109/MWSCAS47672.2021.9531849, 2021.

Lee, J., Park, S., Mo, S., Ahn, S., and Shin, J.: Layer-adaptive sparsity for the magnitude-based pruning, arXiv [preprint], https://doi.org/10.48550/arXiv.2010.07611, 2020.

Lu, J. and Liu, Y.: A real-time and accurate detection approach for bucket teeth falling off based on improved YOLOX, Mech. Sci., 13, 979–990, https://doi.org/10.5194/ms-13-979-2022, 2022.

Matsumura, K., Yamakoshi, T., Yamakoshi, Y., and Rolfe, P.: The effect of competition on heart rate during kart driving: A field study, BMC Research Notes, 4, 342, https://doi.org/10.1186/1756-0500-4-342, 2011.

Qin, D., Leichner, C., Delakis, M., Fornoni, M., Luo, S., Yang, F., Wang, W., Banbury, C., Ye, C., and Akin, B.: MobileNetV4: Universal models for the mobile ecosystem, arXiv [preprint], https://doi.org/10.48550/arXiv.2404.10518, 2024.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A.: You only look once: Unified, real-time object detection, arXiv [preprint], https://doi.org/10.48550/arXiv.1506.02640, 2016.

Ren, H., Jing, F., and Li, S.: DCW-YOLO: road object detection algorithms for autonomous driving, IEEE Access, 13, 125676–125688, https://doi.org/10.1109/ACCESS.2024.3364681, 2024.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., and Han, J.: Yolov10: Real-time end-to-end object detection, arXiv [preprint], https://doi.org/10.48550/arXiv.2405.14458, 2024a.

Wang, C.-Y., Yeh, I.-H., and Mark Liao, H.-Y.: Yolov9: Learning what you want to learn using programmable gradient information, arXiv [preprint], https://doi.org/10.48550/arXiv.2402.13616, 2024b.

Wang, H., Liu, X., Song, L.,Zhang, Y., Rong, X., and Wang, Y.: Research on a train safety driving method based on fusion of an incremental clustering algorithm and lightweight shared convolution, Sensors, 24, 4951, https://doi.org/10.3390/s24154951, 2024c.

Zhang, H., Xu, C., and Zhang, S.: Inner-IoU: more effective intersection over union loss with auxiliary bounding box, arXiv [preprint], https://doi.org/10.48550/arXiv.2311.02877, 2023.