2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Lightweight Fisheye Object Detection Network with Transformer-based Feature Enhancement for Autonomous Driving

Hu Cao [1], Yanpeng Li [1], Yinlong Liu[2*], Xinyi Li [1], Guang Chen [3], Alois Knoll [1]

***Abstract*— Fisheye cameras, offering a wide field of view (FOV) of 360°, are extensively employed for surround-view perception in autonomous driving. Compared with the object detection on the standard images, it lacks studies for fisheye images. Moreover, efficient perception is crucial for autonomous vehicles with limited computational capability. In this work, we introduce a lightweight fisheye object detection network with transformer-based feature enhancement for autonomous driving. Specifically, we leverage ShuffleNet V2 as a feature extraction network to reduce computation complexity and develop a transformer-based feature enhancement module (TFEM) to integrate multi-level features. Notably, we observe that data augmentation methods like mix-up and mosaic, effective on standard images, do not yield positive results on fisheye images. The results on the WoodScape dataset demonstrate that our method can achieve better performance with fewer parameters and floating-point operations per second (FLOPs). Extending our evaluation to the Microsoft Common Objects in Context (MS COCO) dataset shows that the proposed method has excellent generalization capability.**

I. Introduction

In an autonomous driving system, a comprehensive sensor network comprising multiple cameras, lidar, and radar components is employed to perceive the environment across various fields of view and ranges [1], [2], [3], [4], [5]. As illustrated in Fig. 1, the surround-view camera system is a pivotal component, typically composed of four fisheye cameras. This setup enables a 360° near-field perception, playing a crucial role in diverse applications, including automated parking, low-speed maneuvering, and emergency braking [6].

Current research efforts in object detection predominantly concentrate on developing innovative algorithms for standard images, with comparatively less attention given to fisheye images. Among the popular methodologies, two-stage detectors such as Faster RCNN [7] and Mask RCNN [8] achieve high detection accuracy but may exhibit slower running speeds. On the other hand, one-stage methods like YOLO [9], YOLOX [10], SSD [11], and FCOS [12] strike a better balance between performance and efficiency.

In the context of fisheye object detection, existing efforts such as [14], [15] introduce orientation-aware detection methods based on YOLO v3 [16] for overhead fisheye people detection. Nevertheless, the absence of a public fisheye dataset for autonomous driving hinders progress in this field. To address this gap, the WoodScape [13] dataset is created, providing a valuable resource for the development of fisheye perception algorithms. Leveraging this dataset, FisheyeYOLO [17] and OmniDet [18] were introduced for fisheye object detection and multi-task perception, respectively. Nonetheless, with the inherent computational constraints of autonomous vehicles, there is a growing need to explore novel models that strike a balance between detection performance and efficiency. This work aims to explore the design of a novel model that strikes a balance between detection performance and efficiency, catering to the specific needs of autonomous vehicles with limited computational capability.

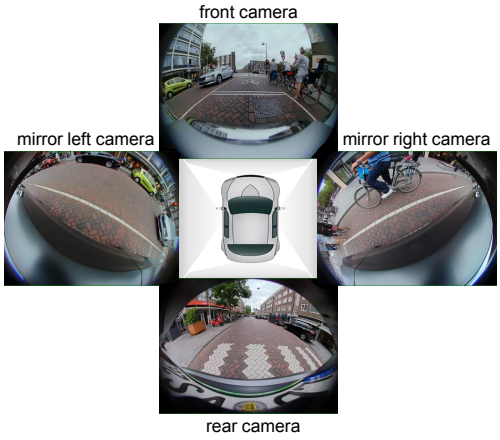

Fig. 1. The illustration for the surround-view camera system. Sample images are from the WoodScape [13] dataset.

Transformer-based models have achieved widespread success in diverse domains, such as natural language processing (NLP) and computer vision (CV) [19], [20], [21], [22]. Leveraging the self-attention mechanism [19], the transformer model has demonstrated superior performance in object perception tasks. In contrast to traditional convolution operations that rely on sliding window operations across the entire input image to extract feature maps, the transformer model based on self-attention excels at capturing global context.

In this study, we introduce a lightweight fisheye object detection network with transformer-based feature enhancement tailored for autonomous driving. Specifically, ShuffleNet V2 serves as the feature extraction network, effectively capturing multi-level features with lower computation complexity. Furthermore, we employ the transformer encoder as a refinement module to enhance these multi-level features. Notably, our

*Yinlong Liu is the corresponding author of this work (YinlongLiu@um.edu.mo)

Authors Affiliation: [1]Chair of Robotics, Artificial Intelligence and Real-time Systems, Technische Universität München, München, Germany, [2]State Key Laboratory of Internet of Things for Smart City (SKL-IOTSC), University of Macau, Macau 999078, China, [3]Tongji University, Shanghai, China.

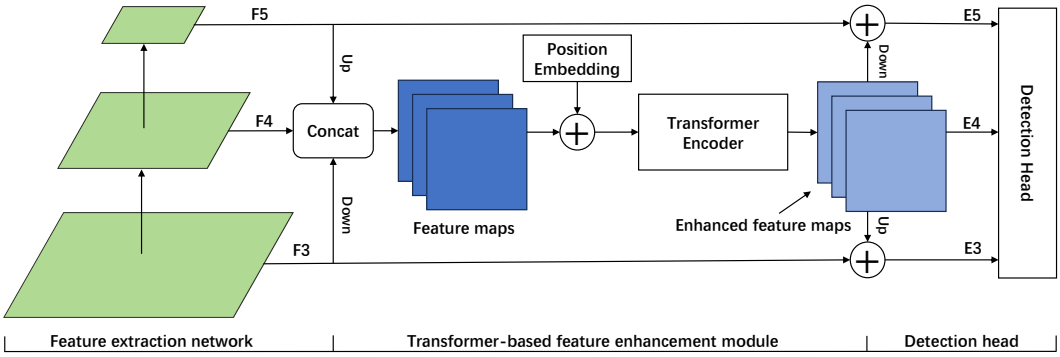

Fig. 2. The overall structure of the proposed lightweight fisheye object detection network. The model consists of a feature extraction network, a transformer-based feature enhancement module, and a detection head.

experiments reveal that data augmentation methods like mix-up and mosaic, proven effective on standard images, do not yield positive results on fisheye images.

Extensive experiments conducted on the WoodScape [13] and MS COCO [23] datasets indicate that the proposed method strikes a favorable balance between accuracy and efficiency. It achieves excellent generalization capability, emphasizing its potential for real-world deployment in autonomous driving scenarios. Our contributions can be summarized as follows:

- We present a lightweight framework for fisheye object detection that strikes a balance between performance and efficiency.
- A transformer-based feature enhancement module (TFEM) is proposed to refine the extracted multi-level features.
- Experimental evaluations on the WoodScape dataset show that the proposed method achieves excellent detection performance with fewer parameters and FLOPs. The results obtained on the MS COCO dataset confirm the excellent generalization ability of our method.

II. Related Work

A. Object detection based on standard camera

Deep learning-based methods for object detection on standard cameras are broadly categorized into two groups: two-stage detectors and one-stage detectors. The notable two-stage detectors include the RCNN series, exemplified by Fast RCNN [24], Faster RCNN [7], and Mask RCNN [8]. However, a drawback of two-stage detectors is their relatively slower running speed. In contrast, one-stage detectors aim to strike a balance between accuracy and efficiency. This category encompasses the YOLO series [9], [16], [10], RetinaNet [25], CenterNet [26], and FCOS [12], offering alternatives that exhibit improved speed without compromising accuracy.

B. Object perception based on fisheye camera

Compared to standard cameras, there has been limited research on fisheye-based object perception, primarily due to the scarcity of public datasets dedicated to fisheye-based object perception. Early works have focused on overhead fisheye people detection for intelligent building applications [14], [15]. To bridge this gap, the WoodScape [13] dataset was collected, providing a valuable resource for fisheye perception research in the context of autonomous driving. Several methods have been proposed for depth estimation in fisheye images, including FisheyeDistanceNet [27], UnRectDepthNet [28], and SVDistNet [29]. Additionally, a self-supervised approach for distance estimation, combined with semantic segmentation, is introduced in [30]. In the field of fisheye object detection, novel representations have emerged to address the challenges of generalized fisheye object detection, as exemplified by FisheyeYOLO [17]. Furthermore, OmniDet [18] has been proposed to cater to surround-view camera-based multi-task visual perception. These contributions collectively contribute to advancing the field of fisheye-based object perception.

III. Method

The overall structure of the proposed lightweight fisheye object detection network is shown in Fig. 2. Our model is composed of three key components: a feature extraction network, a transformer-based feature enhancement module (TFEM), and a detection head. The initial step involves inputting the fisheye image into the feature extraction network to extract multi-level features. Following this, the TFEM is used to strengthen multi-level features. Finally, the detection head is employed to make precise predictions based on the enhanced multi-level features. The detailed process is outlined as follows:

A. Feature extraction network

To enhance the efficiency of our model, we employ ShuffleNet V2 [31] as the feature extraction network. ShuffleNet V2 is a computation-efficient convolutional neural network that is particularly suitable for edge devices with limited power. The key idea of ShuffleNet V2 is that it employs group pointwise convolutions and channel shuffle operations. The combination of group pointwise convolutions and channel shuffle operations forms the ShuffleNet unit, the basic component of ShuffleNet V2 is shown in Fig. 3.

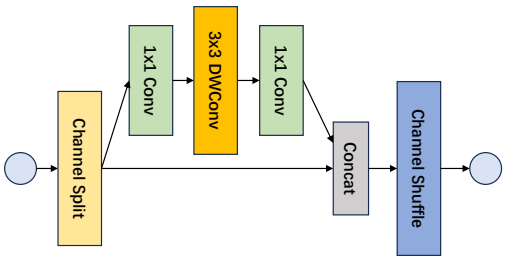

Fig. 3. The basic unit of ShuffleNet V2 [31].

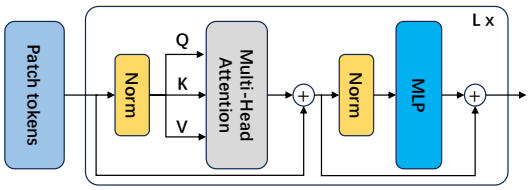

Fig. 4. The framework of the transformer encoder. It includes the norm layer, multi-head self-attention, and MLP layer.

The basic unit involves splitting the input feature channel into two branches with an equal number of channels. One branch remains identical, while the other undergoes corresponding convolution operations. ShuffleNet V2 offers different network sizes, 0.5x, 1x, 1.5x, and 2x, tailored to varying complexities. In this work, we strategically opt for ShuffleNet V2 1x as the backbone. This choice is made with the objective of minimizing both the model size and the number of FLOPS while maintaining accuracy, aligning with our commitment to efficiency-accuracy balance in neural network design.

B. Transformer-based feature enhancement module

High-resolution feature maps with shallow semantic information are beneficial for detecting small objects, while low-resolution feature maps with deep semantic information are more adept at detecting larger objects. Currently, FPN [32], ASFF [33], and PANet [34] are the most widely used multi-level feature enhancement methods, applying top-down pathways and lateral connections for multi-level feature fusion. These methods have demonstrated the complementarity of high-level and low-level features. However, these approaches tend to focus more on features within close proximity and less on those at more distant levels. Inspired by [35], we introduce a strategy of utilizing the transformer encoder to facilitate the seamless flow of information across various levels. This approach ensures a more equitable distribution of information across different feature levels, fostering a comprehensive integration of both high-level and low-level features.

Principle. As shown in Fig. 2, multi-level feature maps, $\boldsymbol{F_3}, \boldsymbol{F_4}, \boldsymbol{F_5}$, from the feature extraction network are fed into the TFEM. First, the channels of these feature maps are unified by pointwise convolution (PWConv), and the resolution of these feature maps is unified by interpolation into the resolution of $\boldsymbol{F_4}$. Second, the unified feature maps are flattened and then added to the position embedding (PE). Then, the transformer encoder is used as a refinement block to generate the refined feature maps. Specifically, this work adopts one transformer encoder with eight heads to refine multi-level features. Finally, these refined feature maps are interpolated and added to multi-level feature maps, $\boldsymbol{F_3}, \boldsymbol{F_4}, \boldsymbol{F_5}$, to output the enhanced feature maps, $\boldsymbol{E_3}, \boldsymbol{E_4}, \boldsymbol{E_5}$. The process can be formulated as follows:

$$\begin{aligned}
\boldsymbol{F_i^{'}} &= \mathrm{PWConv}(\boldsymbol{F_i}), \boldsymbol{i} \in (3,4,5),\\
\boldsymbol{F_3^d} &= \mathrm{Downsample}(\boldsymbol{F_3^{'}}),\\
\boldsymbol{F_5^u} &= \mathrm{Upsample}(\boldsymbol{F_5^{'}}),\\
\boldsymbol{F^c} &= \mathrm{Concat}(\boldsymbol{F_3^d}, \boldsymbol{F_4^{'}}, \boldsymbol{F_5^u}),\\
\boldsymbol{F^t} &= \mathrm{Transformer}(\boldsymbol{F^c} + \mathrm{PE}),\\
\boldsymbol{E}_3 &= \boldsymbol{F_3^{'}} + \mathrm{Upsample}(\boldsymbol{F^t}),\\
\boldsymbol{E}_4 &= \boldsymbol{F_4^{'}} + \boldsymbol{F^t},\\
\boldsymbol{E}_5 &= \boldsymbol{F_5^{'}} + \mathrm{Downsample}(\boldsymbol{F^t}).
\end{aligned} \tag{1}$$

where Downsample and Upsample represent the interpolation functions of downsample and upsample, respectively. The enhanced features $\boldsymbol{E_3}, \boldsymbol{E_4}, \boldsymbol{E_5}$ are fed into the detection head for object classification and boundingbox regression.

Transformer encoder. The input features are first flattened into a set of $\boldsymbol{N}$ patch tokens. These patch tokens are then combined with PE. The incorporation of PE is crucial for enabling the transformer encoder to adeptly capture both the content and spatial context within the input data. The split patch tokens can be formally represented as follows:

$$\boldsymbol{X}^0 = [\boldsymbol{x}_1^0; ...; \boldsymbol{x}_N^0] + \mathrm{PE}. \tag{2}$$

All patch tokens pass through the transformer encoder to perform representation learning. As shown in Fig. 4, a transformer encoder is composed of two fundamental modules: a multi-head self-attention (MHSA), dedicated to modeling the relationships among input tokens, and a multilayer perceptron (MLP), designed to facilitate the learning of more wider representations. The formulation of the transformer encoder can be expressed as:

$$\begin{aligned}
\boldsymbol{Y}^l &= \boldsymbol{X}^{l-1} + \mathrm{MHSA}(\mathrm{Norm}(\boldsymbol{X}^{l-1})),\\
\boldsymbol{X}^l &= \boldsymbol{Y}^l + \mathrm{MLP}(\mathrm{Norm}(\boldsymbol{Y}^l)).
\end{aligned} \tag{3}$$

where $\boldsymbol{X}^l \in \mathbb{R}^{N \times D}$ represents the output of the l-th transformer encoder, serving as both the input to the $(l+1)$-th transformer encoder, and $\boldsymbol{Y}^l \in \mathbb{R}^{N \times D}$ denotes the input to the MLP. The Norm signifies the layer normalization. The MLP consists of two fully connected (FC) layers, incorporating a non-linear activation function. In MHSA, it establishes long-range dependencies through the following computational process:

$$\mathrm{Attention} = \mathrm{SoftMax}(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{\boldsymbol{D_h}}})\boldsymbol{V}. \tag{4}$$

where $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ denote query, key, and value tensors, respectively. $\boldsymbol{D_h}$ represents the head dimension.

C. Detection head

Similar to [10], the detection head used in this work is an anchor-free and decoupled structure. In contrast to anchor-based methods, anchor-free methods facilitate direct predictions without relying on predefined anchors. This characteristic contributes to significantly improved efficiency, making anchor-free methods more effective for real-time applications.

The loss function for our model comprises three main components: classification loss, regression loss, and confidence loss. Regression loss is implemented using IoU loss [36], while both confidence and classification loss are implemented using cross-entropy loss. This combination of loss components is designed to effectively guide the model training, ensuring accurate object localization, confidence estimation, and category classification.

IV. Experiments

In this section, we present the datasets and experimental configurations for fisheye object detection. To assess the efficacy of the proposed method, we conducted experiments on the WoodScape dataset [13]. Additionally, our method's excellent generalization ability is demonstrated through experimental results on the MS COCO dataset [23]. These experiments aim to validate the robustness and performance of our approach across different scenarios and domains.

A. Dataset

WoodScape dataset. WoodScape is a pioneering multi-task, multi-camera fisheye image dataset specifically tailored for autonomous driving applications. It comprises over 10,000 semantically annotated images and 100,000 annotated images for various other tasks. The dataset is intentionally designed to inspire researchers to develop and tailor computer vision models for fisheye images, catering to a spectrum of autonomous driving tasks. These tasks include but are not limited to semantic segmentation, depth estimation, 2D and 3D object detection, visual odometry, SLAM (simultaneous localization and mapping), motion segmentation, soiling detection, and end-to-end driving. The richness and diversity of annotations within WoodScape make it a valuable resource for advancing research in autonomous driving vision systems.

In the publicly released WoodScape dataset, there are 8,234 labeled fisheye images and 1,766 unlabeled fisheye images. For this study, a subset of 6,500 fisheye images from the labeled set is randomly selected as training data, while the remaining 1,734 fisheye images are reserved for testing data. The images were captured by four RGB fisheye cameras, each with a 190-degree horizontal field of view and a resolution of one million pixels. These images were acquired across various locations in the United States, Europe, and China. Each fisheye image maintains an original resolution of 1280×966. The dataset is annotated with five categories, encompassing pedestrians, vehicles, bicycles, traffic lights, and traffic signs, providing a diverse and comprehensive set of scenarios for evaluating fisheye object detection methods.

MS COCO dataset. In addition to the WoodScape dataset, this study leverages the MS COCO dataset to assess the generalization capabilities of the proposed method. MS COCO is a large-scale dataset designed to address various computer vision tasks, including object detection, instance segmentation, dense human pose estimation, keypoint detection, and image captioning. The dataset encompasses a substantial collection of 2,500,000 labeled instances across 328,000 images, covering 91 distinct classes of objects recognizable by a 4-year-old. For evaluation purposes, this work utilizes the train and val images from the MS COCO 2017 dataset, providing a diverse and challenging benchmark for assessing the proposed fisheye object detection method.

B. Evaluation metrics

In this study, the widely used mean average precision (mAP) serves as the evaluation metric for the detection models. mAP is calculated as the average precision (AP) value across all detected classes. The mAP score ranges from 0 to 1, where a higher value signifies more accurate object detection models.

The average precision (AP) is determined by computing the area under the precision-recall curve (P-R curve). The P-R curve is formed by a series of precision and recall pairs, ranging from 0 to 1. Precision represents the percentage of correct predictions made by the detection model, while recall indicates the percentage of positive samples predicted by the model. The mathematical expressions for precision and recall are expressed as follows:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}}, \\ \text{Recall} &= \frac{\text{TP}}{\text{TP}+\text{FN}}. \end{aligned} \tag{5}$$

where TP stands for true positive, FP stands for false positive, and FN stands for false negative.

The Intersection over Union (IoU) metric determines whether a prediction belongs to true positive (TP) or false positive (FP) by calculating the ratio between the overlap and the union of the detections and ground truth. The IoU is computed as shown in Eq. 6. Varying IoU thresholds result in different average precision (AP) values. For instance, AP_{50} refers to average precision with an IoU threshold of 0.5, while AP_{75} refers to average precision with an IoU threshold of 0.75.

$$\text{IoU} = \frac{\text{Detections} \cap \text{Groundtruth}}{\text{Detections} \cup \text{Groundtruth}}. \tag{6}$$

In the MS COCO dataset [23], the final mean average precision (mAP) is computed by considering the mAP across multiple IoU thresholds. These IoU thresholds range from 0.50 to 0.95, with a step size of 0.05. Furthermore, there are APs grouped by object size, including AP_S (area $<32^2$ pixels) for small objects, AP_M (32^2 pixels $<$area $<96^2$

TABLE I

PARAMETER CONFIGURATIONS FOR TRAINING.

Parameters	Configurations
Image size	640×640
GPU	RTX3090
Batch size	16
Learning rate	0.01
Optimizer	SGD
Momentum	0.9
Weight decay	0.0005
Warmup ratio	1
Warmup iterations	5
Warmup method	exponentially

TABLE II

ABLATION STUDY OF DIFFERENT DATA AUGMENTATION METHODS ON THE WOODSCAPE DATASET.

Mix-up	Mosaic	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
		39.2	**68.2**	**39.7**	**28.1**	**51.4**	**68.1**
✓		33.2	59.6	32.3	23.5	44.8	61.7
✓	✓	27.3	51.6	25.2	18.0	37.3	43.2

TABLE III

ABLATION STUDY OF THE EFFECTIVENESS OF EACH COMPONENT ON THE WOODSCAPE DATASET.

ShuffleNet V2	TFEM	Params (M) ↓	FLOPs (G) ↓	mAP ↑
		13.32	8.94	52.9
✓		8.57	5.46	45.6
✓	✓	**7.08**	**4.0**	**53.5**

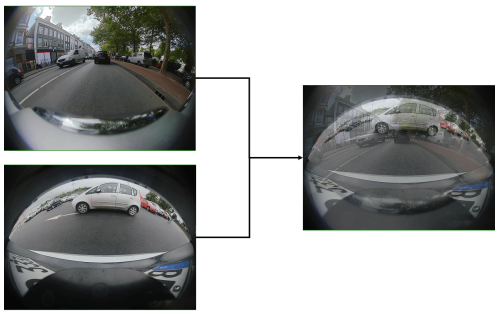

Fig. 5. An example of mix-up augmentation without ground truth-bounding boxes.

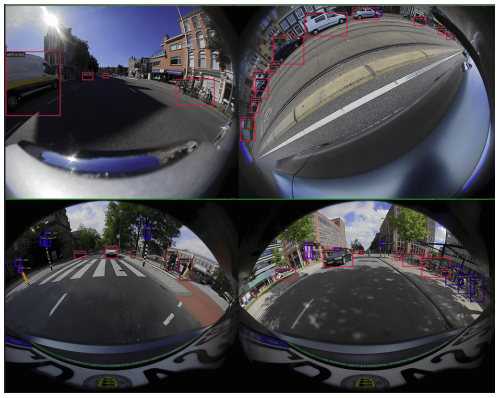

Fig. 6. An example of mosaic augmentation with ground truth-bounding boxes.

pixels) for medium objects, and AP_L (area $>96^2$ pixels) for large objects. This detailed evaluation provides insights into the model's performance across different object sizes and IoU thresholds.

C. Implementation details

In our experiments, we initialize the backbone using pre-training weights on ImageNet [37]. The input image is resized to a resolution of 640×640 before being input to the backbone. All experiments are carried out on an RTX 3090 graphics card with a memory capacity of 24 GB.

The training process remains consistent with the source code released by OmniDet [18]. The proposed model is trained with an SGD optimizer, where the batch size is set to 16 and the initial learning rate is set to 0.01. Additional parameter configurations are detailed in Tab. I. Furthermore, our algorithm is implemented using PyTorch [38].

D. Ablation study

Influence of data augmentation. In [10], the effectiveness of both mix-up and mosaic for enhancing the accuracy of object detection on standard images has been demonstrated. However, in our experiments, we observed that data augmentation methods such as mix-up and mosaic, which prove effective on standard images, do not yield positive results on fisheye images. In fact, these augmentation techniques were found to degrade the performance of object detection when applied to fisheye images. The results are shown in Tab. II. This highlights the importance of considering the specific characteristics and requirements of fisheye images when choosing and applying data augmentation strategies.

As illustrated in Fig. 5 and Fig. 6, notable incoherence and differences are evident in the merged images. This discrepancy is attributed to the inherent property of fisheye cameras, which capture images within a circular area. In light of these observations, mix-up and mosaic data augmentation methods were omitted in this work.

Effect of key components. In Tab. III, we present a summary of results for different module configurations. The baseline model chosen for this analysis is YOLOX-S [10]. By substituting the backbone with ShuffleNet V2 [31], a significant reduction in computational complexity is achieved: the parameter size shrinks from 8.94 M to 5.46 M, and the FLOPs decrease from 13.32 G to 8.57 G. However, the detection performance experiences a decline compared to the baseline model.

To enhance detection performance while maintaining computational efficiency, we introduced a TFEM for multi-level feature fusion. The results demonstrate that the combination of ShuffleNet V2 and TFEM yields improved performance and efficiency, striking a balance between model accuracy and computational cost.

TABLE IV

THE PERFORMANCE OF DIFFERENT DETECTION METHODS ON THE WOODSCAPE DATASET.

Method	Params (M) ↓	FLOPs (G) ↓	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
OmniDet [18]	25.03	38.11	35.9	**69.3**	33.3	24.7	45.7	44.2
Ours	**4.00**	**7.08**	**40.5**	68.5	**41.6**	**30.9**	**51.4**	**72.2**

TABLE V

THE PERFORMANCE OF DIFFERENT DETECTION METHODS ON THE MS COCO DATASET.

Method	Params (M) ↓	FLOPs (G) ↓	mAP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
OmniDet [18]	25.03	38.11	21.5	**41.4**	20.6	10.5	24.2	27.0
Ours	**4.00**	**7.08**	**22.6**	39.7	**22.9**	**8.6**	**25**	**32.8**

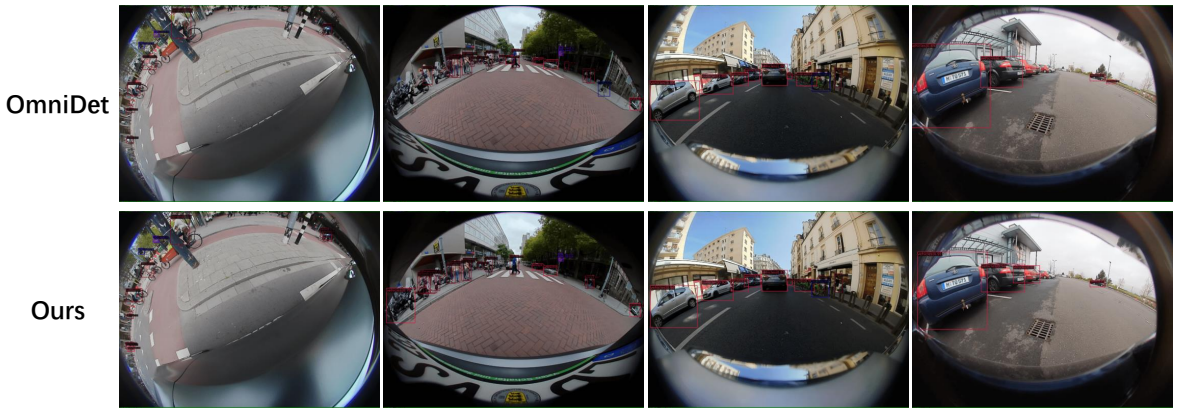

Fig. 7. Detection results on the WoodScape dataset. The first row represents the detection results produced by OmniDet, and the last row denotes the detection results produced by our method.

E. Experimental results on the WoodScape dataset

We conducted a comparison between the proposed method and OmniDet [18]. Specifically, we trained OmniDet using the published source codes on our split dataset, and the hyperparameters for both our method and OmniDet were maintained at the same settings. The experimental results are summarized in Tab. IV. Notably, the proposed method outperforms OmniDet by 4.6 mAP while simultaneously reducing the parameter size from 25.03 M to 4 M and decreasing the FLOPs from 38.11 G to 7.08 G. This outcome signifies that our method achieves superior detection accuracy with fewer computational costs, resulting in faster inference times. For a visual representation of the detection results, refer to Fig. 7. These visualizations further underscore the improved performance of the proposed method compared to OmniDet.

F. Generalization ability on the MS COCO dataset

Additionally, we trained both our method and OmniDet on the MS COCO dataset to evaluate their generalization abilities on standard, non-distorted images. As indicated in Tab. V, the proposed method achieves better detection performance than OmniDet by 1.1 mAP. These results highlight the consistent performance improvements offered by our method across both fisheye and standard images, emphasizing its versatility and effectiveness in diverse scenarios.

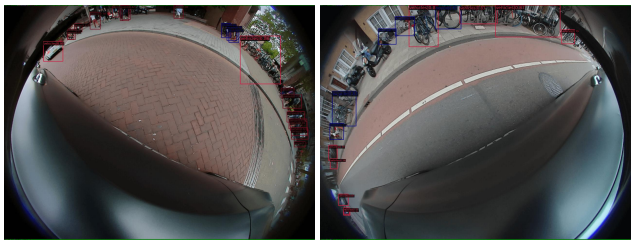

Fig. 8. Failure detection cases. The proposed model exhibits misclassification, often identifying bicycles as vehicles.

G. Failure case analysis

It's worth noting that, similar to many object detection methods, our approach faces challenges in crowded scenes. As illustrated in the Fig. 8, in a densely populated bicycle scene, the proposed method incorrectly identifies the bicycle as a vehicle. Crowded scenarios, where objects are closely packed or occluded, can pose difficulties for accurate detection. Addressing such challenges often requires specialized techniques, and future improvements to our method could involve refining its performance in crowded scenes for enhanced accuracy and robustness.

V. Conclusion

In this study, we present a lightweight fisheye object detection network tailored for autonomous driving. Our approach leverages the efficient backbone of ShuffleNet V2 to reduce computational costs. To facilitate the flow of information across multi-level features, we introduce a TFEM for effective multi-level feature fusion. Notably, we observe that data augmentation methods like mix-up and mosaic, effective on standard images, do not yield positive results on fisheye images. Experimental results on the WoodScape and MS COCO datasets showcase that our proposed method achieves excellent detection performance and generalization capabilities. Importantly, it accomplishes this with fewer parameters and FLOPs, emphasizing its efficiency and applicability in fisheye-based object detection for autonomous driving scenarios.

Acknowledgement

This work is supported by the MANNHEIM-CeCaS (Central Car Server–Supercomputing for Automotive, No. 16ME0820), in part by National Natural Science Foundation of China (No. 62372329), in part by Shanghai Scientific Innovation Foundation (No.23DZ1203400), in part by Tongji-Qomolo Autonomous Driving Commercial Vehicle Joint Lab Project, and in part by Xiaomi Young Talents Program.

References

[1] D. Feng, C. Haase-Schütz, L. Rosenbaum, H. Hertlein, C. Glaeser, F. Timm, W. Wiesbeck, and K. Dietmayer, "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1341–1360, 2020.

[2] H. Cao, G. Chen, H. Zhao, D. Jiang, X. Zhang, Q. Tian, and A. Knoll, "Sdpt: Semantic-aware dimension-pooling transformer for image segmentation," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2024.

[3] J. Horgan, C. Hughes, J. McDonald, and S. Yogamani, "Vision-based driver assistance systems: Survey, taxonomy and advances," in *ITSC*, 2015.

[4] H. Cao, G. Chen, J. Xia, G. Zhuang, and A. Knoll, "Fusion-based feature attention gate component for vehicle detection based on event camera," *IEEE Sensors Journal*, vol. 21, no. 21, pp. 24 540–24 548, 2021.

[5] Y. Cai, T. Zhang, H. Wang, Y. Li, Q. Liu, and X. Chen, "3d vehicle detection based on lidar and camera fusion," *Automotive Innovation*, vol. 2, pp. 276–283, 2019.

[6] M. Heimberger, J. Horgan, C. Hughes, J. McDonald, and S. Yogamani, "Computer vision in automated parking systems: Design, implementation and challenges," *Image and Vision Computing*, vol. 68, pp. 88–101, 2017.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *NeurIPS*, 2015.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *ICCV*, 2017.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

[10] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint*, 2021.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *ECCV*, 2016.

[12] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *ICCV*, 2019.

[13] S. Yogamani, C. Hughes, J. Horgan, G. Sistu, P. Varley, D. O'Dea, M. Uricár, S. Milz, M. Simon, K. Amende *et al.*, "Woodscape: A multi-task, multi-camera fisheye dataset for autonomous driving," in *ICCV*, 2019.

[14] Z. Duan, O. Tezcan, H. Nakamura, P. Ishwar, and J. Konrad, "Rapid: rotation-aware people detection in overhead fisheye images," in *CVPRW*, 2020.

[15] H. Cao, B. Peng, L. Jia, B. Li, A. Knoll, and G. Chen, "Orientation-aware people detection and counting method based on overhead fisheye camera," in *IEEE MFI*, 2022.

[16] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint*, 2018.

[17] H. Rashed, E. Mohamed, G. Sistu, V. R. Kumar, C. Eising, A. El-Sallab, and S. Yogamani, "Generalized object detection on fisheye cameras for autonomous driving: Dataset, representations and baseline," in *WACV*, 2021.

[18] V. R. Kumar, S. K. Yogamani, H. Rashed, G. Sistu, C. Witt, I. Leang, S. Milz, and P. Mäder, "Omnidet: Surround view cameras based multi-task visual perception network for autonomous driving," *IEEE Robotics Automation Letter*, vol. 6, no. 2, pp. 2830–2837, 2021.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, 2017.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint*, 2020.

[21] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on vision transformer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 87–110, 2022.

[22] H. Cao, Z. Qu, G. Chen, X. Li, L. Thiele, and A. Knoll, "Ghostvit: Expediting vision transformers via cheap operations," *IEEE Transactions on Artificial Intelligence*, 2023.

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014.

[24] R. Girshick, "Fast r-cnn," in *ICCV*, 2015.

[25] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *ICCV*, 2017.

[26] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *ICCV*, 2019.

[27] V. R. Kumar, S. A. Hiremath, M. Bach, S. Milz, C. Witt, C. Pinard, S. K. Yogamani, and P. Mäder, "Fisheyedistancenet: Self-supervised scale-aware distance estimation using monocular fisheye camera for autonomous driving," in *ICRA*, 2020.

[28] V. R. Kumar, S. K. Yogamani, M. Bach, C. Witt, S. Milz, and P. Mäder, "Unrectdepthnet: Self-supervised monocular depth estimation using a generic framework for handling common camera distortion models," in *IROS*, 2020.

[29] V. R. Kumar, M. Klingner, S. K. Yogamani, M. Bach, S. Milz, T. Fingscheidt, and P. Mäder, "Svdistnet: Self-supervised near-field distance estimation on surround view fisheye cameras," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2021.

[30] V. R. Kumar, M. Klingner, S. S. Milz, T. Fingscheidt, and P. Mäder, "Syndistnet: Self-supervised monocular fisheye camera distance estimation synergized with semantic segmentation for autonomous driving," in *WACV*, 2021.

[31] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *ECCV*, 2018.

[32] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017.

[33] S. Liu, D. Huang, and Y. Wang, "Learning spatial fusion for single-shot object detection," *arXiv preprint*, 2019.

[34] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *CVPR*, 2018.

[35] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra r-cnn: Towards balanced learning for object detection," in *CVPR*, 2019.

[36] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *ACM MM*, 2016.

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[38] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, 2019.