Improving Object Detection to Fisheye Cameras with Open-Vocabulary Pseudo-Label Approach

Long Hoang Pham* Quoc Pham-Nam Ho* Duong Nguyen-Ngoc Tran Tai Huu-Phuong Tran
Huy-Hung Nguyen Duong Khac Vu Chi Dai Tran Ngoc Doan-Minh Huynh
Hyung-Min Jeon Hyung-Joon Jeon Jae Wook Jeon†
Department of Electrical and Computer Engineering
Sungkyunkwan University
{phlong, hpnquoc, duongtran, taithp, huyhung91, vukhacduong,
tdc2000, ngochdm, hmjeon, joonjeon, jwjeon}@skku.edu *

Abstract

Fish-eye cameras have long been employed in traffic surveillance systems to allow for wider observation of the roads. Despite their widespread use, limited computer vision research is tailored explicitly to images captured by fish-eye cameras. The AI City Challenge 2024 - Track 4 introduces a novel fish-eye camera dataset for the 2D road object detection task. This paper proposes a framework designed to detect objects in fish-eye camera images. Our approach involves several key steps: first, we generate image data to bridge the representation gap between day and night images. Next, we leverage zero-shot open vocabulary detection to produce pseudo-labels, aiding in training supervised object detection models. Additionally, we optimize the model's hyper-parameters and inference configuration for better performance. Finally, we apply various post-processing techniques to enhance detection performance. Our solution achieves a final F1 score of 0.6194 in the AI City Challenge 2024 - Track 4, ranking third among competing teams. The source code is available at GitHub Repo.

1. Introduction

Traffic surveillance using cameras is one of the essential components of smart cities. Typically, conventional surveillance cameras have a limited field-of-view (FOV), thus requiring multiple cameras to cover an intersection fully. Fisheye cameras have become an alternative solution due to their wide and omnidirectional coverage capabilities. They offer a significant advantage by efficiently reducing the required cameras to capture broader views of roads and intersections.

Despite their advantages, fish-eye cameras present unique challenges for object detection. The distortion resulting from the circular representation of the camera's field of view leads to significant object deformation, particularly near the center and image boundaries. This warping compresses and skews environmental information, making precise object localization difficult. While previous approaches have attempted to convert fish-eye images into rectangular panoramas to mitigate this distortion, they still need to fully address the variations in object scales caused by their relative distances to the camera. Nonetheless, fish-eye cameras represent a valuable source of data and an underexplored research field.

The limited availability of publicly annotated fish-eye datasets in the literature hinders the current research progress in fish-eye cameras. To address this limitation, the FishEye8K dataset [1] has been introduced as the first dataset for 2D road object detection in fish-eye cameras. This dataset is tailored for traffic surveillance applications across five object classes: bus, bike, car, pedestrian, and truck. The FishEye8K dataset was also established as a challenge track in the AI City Challenge 2024 (AIC24) [2].

In this paper, we proposed a solution to the 2D road object detection task in the FishEye8K [1] dataset. Given that the dataset lacks the calibration matrix for each camera, our approach revolves around adapting knowledge from the general object detection domain to fish-eye cameras. As illustrated in Fig. 1, our proposed method consists of three main modules as follows:

- We present an Image Generation Module (IGM) utilizing Style Transfer to generate nighttime images from their daytime counterparts synthetically. The primary

*This work was supported by the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2021-0-01364, An intelligent system for 24/7 real-time traffic surveillance on edge devices)

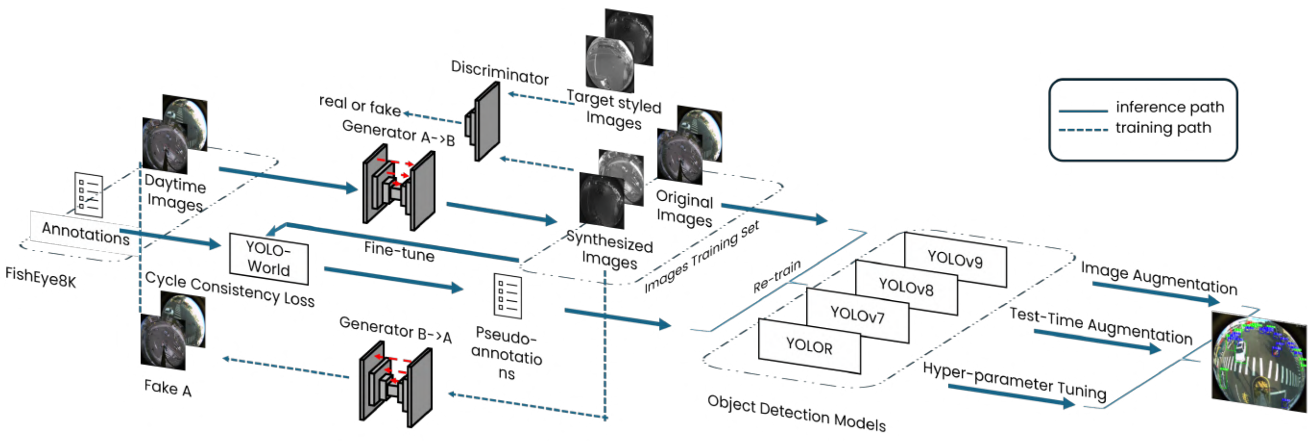

Figure 1. **The framework of our solution**.

objective is to augment the number of nighttime images, which are limited in the original FishEye8K dataset [1] training set.

- We propose an Open-Vocabulary Pseudo-Labels (OVPL) strategy that integrates knowledge distillation with semi-pseudo-labels. We employ a zero-shot open-vocabulary detection model as the teacher model to generate pseudo-labels for previously unlabeled images. These pseudo-labels are subsequently employed to train the object detection models using the standard supervised learning strategy.
- We implement an Object Detection Module (ODM) to identify 2D road objects. All object detection models are trained with guidance from the proposed PLG module. Additionally, the module integrates several popular techniques, including augmentation (both at training and test-time) and ensemble methods, to generate the final results.

To evaluate the proposed method, we submit the results to the public leaderboard of AIC24 - Track 4. Experimental results demonstrate competitive performance compared to other methods, achieving a top-3 ranking with an F1 score of 0.6194, with only a marginal difference of 0.002 from the second place.

2. Related Work

2.1. Traffic Surveillance Dataset

Traffic surveillance and monitoring stand as one of the most well-established applications in computer vision. With recent advancements in deep learning-based methods, there is a growing demand for high-quality traffic surveillance datasets. However, most existing datasets reside in private domains of government and industry, limiting access for researchers in the academic community. To overcome this obstacle, several datasets have been publicly released in recent years, including UA-DETRAC [3], MIO-TCD [4], AAU RainSnow [5], CityFlow [6], and FishEye8K [1]. These datasets provide bounding box labels for object detection tasks, and some also include unique IDs for object tracking-related tasks [3, 6]. While most datasets are captured using conventional RGB cameras, FishEye8K [1] stands out as the first dataset to utilize fish-eye lenses, offering omnidirectional and wide coverage of road intersections. Some examples of the FishEye8K [1] dataset are visualized in Fig. 2. This trend underscores traffic surveillance's significance in academic literature and industry.

2.2. Style Transfer

Style transfer or image translation from one domain to another is integral to numerous computer vision tasks. This problem is a common challenge in various domains, including medical image analysis, autonomous driving, and virtual reality. Generative adversarial networks (GANs) based methods, which emerged as a highly promising approach for image stylization, comprise two distinct components: generative and discriminative models. While the latter component is used to distinguish between the real and the generated images, the former is utilized to capture the probability distribution of data to generate images. Pix2Pix [7] is one of several ideas that uses a conditional GAN that learns to map input images from one domain to corresponding output images in another domain. Many subsequent works have employed a similar approach to that of Pix2Pix [7], utilizing the technique to tackle various tasks, including the generation of high-resolution images from semantic label maps [8] or the creation of photographic images from sketches [9, 10]. However, these approaches require paired training examples, limiting the possibility of generating more data.

Conversely, certain methods do not necessitate having corresponding data pairs. CycleGAN [11] extends the

GAN-based method to unsupervised, which results in no pair data required, with the idea of cyclic consistency. Or shared latent space methods like UNIT [12] and MUNIT [13] for multiple-domain translation.

2.3. Object Detection

Object detection remains one of the fundamental tasks in traffic surveillance systems and has been extensively studied. In earlier days, on-road object detection primarily relied on extracting moving foregrounds through background subtraction methods [14], followed by classification using machine learning techniques such as SVM or Decision Tree [15]. However, this approach requires manual feature engineering tailored to each context, limiting generalization capabilities. With the advent of deep learning, object detection can now be accomplished end-to-end, leading to more efficient and accurate detection capabilities.

Currently, YOLO [16] with its successor variants YOLOR [17], YOLOv5 [18], YOLOv7 [19], YOLOv8 [20], and YOLOv9 [21] have gained immense popularity for their ease of training and fine-tuning, particularly with custom datasets. These models are single-stage object detection methods that predict object classes and bounding box coordinates in a single pass. These models leverage predefined bounding box anchors and adjust scales and aspect ratios for accurate object localization. Moreover, they are supervised learning approaches that require input data in the form of images along with corresponding bounding box labels. This requirement poses a limitation, as acquiring and annotating labels for unseen data can be costly and time-consuming. Furthermore, supervised methods are sensitive to erroneous labels, which can significantly impact the performance and reliability of the model.

Recently, the zero-shot open-vocabulary object detection models, for instance, Grounding DINO [22] and YOLO-World [23], have shown impressive performance. These models accept one or more text prompts and aim to identify the location of all the objects of interest. While zero-shot detection models demonstrate increasing accuracy in detecting objects, smaller custom models are faster, more compute-efficient, and more accurate in specific domains. However, despite these trade-offs, large zero-shot models offer a significant advantage as they can automatically label unseen data.

3. Methodology

An overview of our solution can be visualized in Fig. 1. We first provide a summary of the FishEye8K [1] dataset, followed by a thorough analysis of its contents (Section 3.1). In Section 3.2, we use the IGM to synthesize nighttime images from the corresponding day-time scenes to bridge the representation domain gap. In the next step (Section 3.3), we deploy the PLG training strategy with the state-of-the-art open-vocabulary object detection model, YOLO-World [23], to infer the unlabelled images with bounding box pseudo-labels, thereby performing active learning. Finally, we train multiple object detection models within the standard supervised learning pipeline (Section 3.4). We also apply standard inference techniques such as test-time augmentation (TTA) and ensemble to produce the final solution to the challenge.

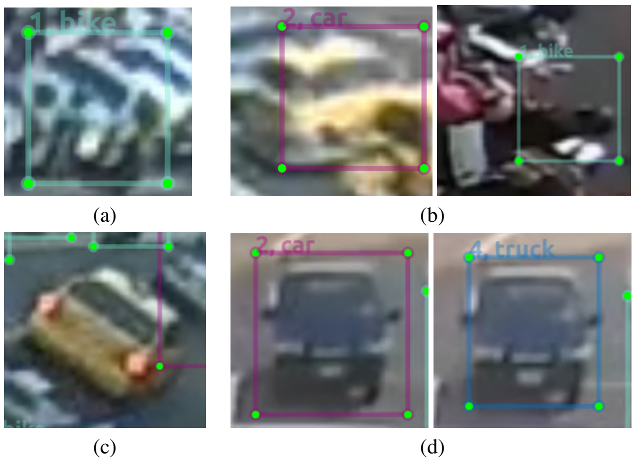

Figure 2. **Examples erroneous labels in FishEye8K [1] dataset.** a) Wrong classification. b) Inaccurate localization. c) Missing labels. d) Inconsistency labels.

Table 1. Summary of our dataset derived from FishEye8K [1].

	Item	**Distribution**
images	train	5,288
	val	2,712
	test	1,000
	background	800
	synthesis	5,841
	Total	15,641

	Class	**Distribution**	**Min (px)**	**Max (px)**
labels	bus	5,751	864	36,7302
	bike	157,544	100	39,476
	car	90,878	100	109,324
	pedestrian	22,079	230	5,104
	truck	6,362	1,435	96,775
	Total	282,614		

3.1. FishEye8K Dataset

Dataset Summary. Before proceeding, we analyze the FishEye8K dataset [1]. In summary, the dataset comprises 8,000 images extracted from 18 pre-recorded videos. These images are divided into 5,288 for training and 2,712 for validation. A total of 157,012 annotated bounding boxes are provided across 5 road object classes: ”bus”, ”bike”, ”car”, ”pedestrian”, and ”truck”. Additionally, an additional 1,000 unlabeled images are included for the testing phase. Throughout the dataset, the images' resolutions of 1245×1080 and 1920×1920 are mainly used. Most road objects are either "bike" or "car", which occupy 56.3% and 32.3% of the total bounding boxes. Alternatively, bounding boxes labeled as "pedestrian" exhibit a moderate number, 7.5% over the total boxes. At the same time, the dataset represents a modest number of objects labeled as "bus" and "truck", around 2% per class. *One final note is that the dataset does not include the camera calibration matrix for each camera. As a result, the conversion of fish-eye images into rectangular panoramas is not feasible.*

Annotation Quality. Upon conducting a thorough examination of the dataset, we have identified several flaws that could potentially impact the accuracy of the detection models. Specifically, we have uncovered four main issues: incorrect class labeling, inconsistent labeling, and inaccurate localization. Examples of these erroneous labels are visualized in Fig. 2. One common error we encountered is the bounding box of an object labeled as "bike" covering only half of the human body, which may lead to ambiguity for the model to distinguish between the classes "bike" and "pedestrian" (Fig. 2b). On the other hand, there are some inconsistencies between object classes in consecutive frames. For instance, in Fig. 2d, a "truck" instance has two different classes in two frames. Moreover, we notice many objects with missing labels, as shown in Fig. 2c.

The dataset cannot be utilized directly due to the presence of erroneous bounding boxes. Therefore, we undertake a data-cleaning procedure to eliminate incorrect bounding boxes and annotate additional objects to enhance image consistency. Table 1 presents the comprehensive distribution of annotated instances.

3.2. Image Generation Module (IGM)

In the FishEye8K [1] dataset, most images are captured during the daytime, while some are captured at nighttime. There is a significant difference between the two scenarios. The daytime images are in full RGB color, whereas the nighttime images are black-and-white (BW). However, a major issue is the limited number of annotated nighttime images. Hence, the dataset is skewed towards daytime images, which may lead to suboptimal training results. There are two approaches to address this issue: training separate models for each scenario or balancing the data. This paper follows the second approach by synthesizing more nighttime images using style transfer.

We opted for an optimized and lightweight solution to enhance the dataset by employing style transfer, specifically utilizing CycleGAN [11] to transform the daytime images in the dataset into a comparable nighttime version. CycleGAN [11] was selected due to its ability to perform style transfer without requiring paired data and ease of training and fine-tuning. We divided the dataset into two subsets: morning and afternoon images and images from the remaining periods. Due to time constraints, we chose not to split the data into four categories: morning, afternoon, evening, and night. The CycleGAN [11] model consists of two generative networks based on the image transformation network by Johnson et al. [24] and two discriminative networks employing 70x70 PatchGAN [7]. Examples of images generated by the image generation module are depicted in Fig. 3. Adding this augmented data significantly enhanced the performance of our detection models, as evidenced by the results outlined in Table 2.

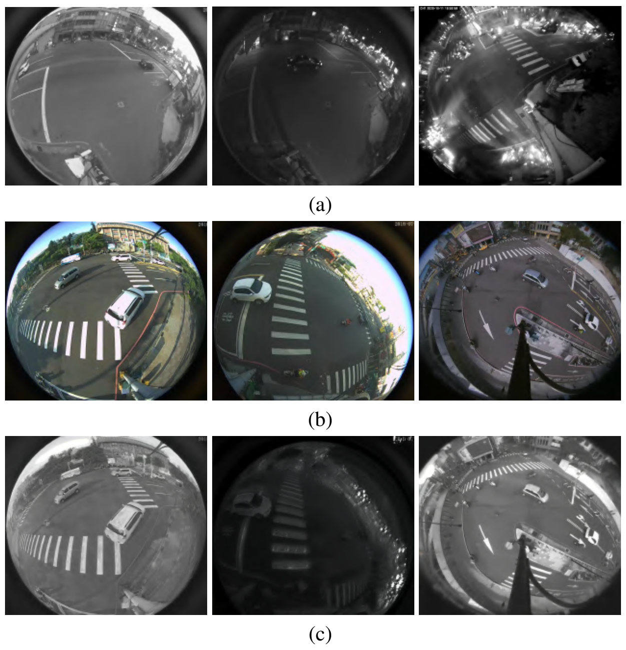

Figure 3. **Examples of nighttime data generation using style transfer**. a) Real nighttime image. b) Original daytime images. c) Synthesized nighttime images.

3.3. Open-Vocabulary Pseudo-Labels (OVPL)

Zero-shot object detection models provide a solution for localizing objects in an image using text prompts. These models are trained on extensive datasets to recognize a wide range of objects, eliminating the need for custom training. Despite their effectiveness, zero-shot models tend to be large and resource-intensive compared to fine-tuned models, making them impractical for large-scale, real-time, or edge applications. However, they can still be utilized across images for analysis or to automatically label data, serving as a robust teacher model for training smaller, fine-tuned models.

In our framework, we integrate the YOLO-World [23] model to perform pseudo-labeling on the unannotated test set of the FishEye8K [1] dataset. We chose the YOLO-

World's implementation [20] for its state-of-the-art performance and ease of training and fine-tuning on custom datasets. Specifically, we optimized the model to detect only the classes defined in the FishEye8K [1] dataset. By providing custom prompts, we direct the model's attention towards objects of interest, enhancing the detection results' relevance and accuracy. We choose a moderate confidence threshold value of 0.3 during the inference stage. This threshold selection helps mitigate false positive detections while inferring the test-set images. The detailed training procedures can be found in Section 4.3.

In the following phase, we leverage the pseudo-labeled data acquired from YOLO-World [23], making slight manual corrections before dividing it into training and validation subsets. The incorporation of pseudo-labels has significantly enhanced the performance of our object detection models, as demonstrated in Table 2.

3.4. Object Detection Module (ODM)

Model. Our selection process entailed an extensive review of the literature to identify the top-performing models similar to [1]. Due to limitations in time and resources, we focused solely on single-stage object detection models. This included assessing models such as YOLOR [17], YOLOv7 [19], YOLOv8 [20], and YOLOv9 [21]. We experimented with the largest variant available for each model to explore its potential for our task.

Data Augmentation. Data augmentation involves applying various transformations to the training images. These transformations help increase the diversity of the training data, allowing the model to learn from a wider range of scenarios and improve its generalization ability. Our framework applies the following transformations during training: rotation, scaling, horizontal flipping, random cropping, mixup, and mosaic.

Test-Time Augmentation (TTA) [25] is a technique used during inference where multiple augmentations of a single test image are generated, and the model makes predictions on each augmented version. The final prediction for each object is typically obtained by aggregating the predictions from all augmented image versions, such as taking the average or the maximum confidence score. This technique enables the model to detect objects from various perspectives, which increases its ability to identify objects that may not be visible from certain angles. The TTA combines scaling (×1, ×0.83, ×0.67) and horizontal flipping for all YOLO models.

Ensemble. Model ensembling is a strategy used to boost overall performance by merging predictions from multiple individual models. The goal is to minimize variance and bias, enhancing the final predictions' reliability and consistency. Mean ensemble, a widely used ensembling method, is frequently applied to YOLO models and other object detection architectures. Each YOLO model's predictions are gathered and aggregated in the mean ensemble by calculating the average of their bounding box coordinates and confidence scores for each detected object. This amalgamation process yields a more polished and inclusive collection of predictions compared to the output of any single model on its own.

4. Implementation and Experiments

4.1. Experimental Settings

Implementation Details. We train and test all deep learning models on a machine equipped with 4× NVIDIA A6000 GPUs, each with 48GB of VRAM. All methods are implemented using PyTorch.

Dataset. Our training and validation datasets combine the provided data and our generated data (as described in Sec. 3.2 and 3.3). Additionally, we obtained 800 background images without any labels from the COCO [26] dataset to reduce false positives, as suggested in [20]. We obtained around 15,641 images with 282,614 bounding boxes for training and validation. A summary of our modified dataset can be found in 1.

4.2. Metrics

We use the COCO evaluation script [26] to measure the performance of all models. The evaluation metrics include *Precision*, *Recall*, APs, and F1 score. However, our primary focus lies on the F1 score, given its significance as the main evaluation criterion for the AIC24 - Track 4 [2]. The F1 score is calculated as:

$$\begin{aligned} F_1 &= \frac{2 \times Precision \times Recall}{Precision + Recall}, \\ Precision &= \frac{TP}{TP + FP}, \; Recall = \frac{TP}{TP + FN}, \end{aligned} \tag{1}$$

where TP, FP, and FN are the true positives, false positives, and false negatives.

The choice between using AP or F1 scores can dictate different approaches to inference. When aiming to optimize AP, the primary objective is to enhance the precision-recall curve. This is often achieved by reducing the confidence threshold, leading to more true positive detections and improving recall. Consequently, this strategy typically yields a higher number of overall detections. Conversely, prioritizing the F1 score aims to strike a harmonious balance between precision and recall, thereby minimizing false positives and negatives. Simply increasing true positives may not improve the F1 score if it results in a disproportionate rise in false positives. Consequently, adjusting model thresholds or employing post-processing techniques to diminish false positives and negatives simultaneously becomes crucial for optimizing the F1 score. While AP optimization often involves lowering the confidence threshold

Table 2. Ablation study on the impact of proposed modules: Background images (BG), Image Generation Module (IGM), Open-Vocabulary Pseudo-Labels (OVPL), and Ensemble (Ens). The first row is the baseline results from YOLOR-D6. All models are trained using the input size of 1280 for 50 epochs and are inferred with TTA on the validation set of FishEye8K [1].

BG	IGM	OVPL	AP	F1
	Baseline		28.19	37.68
✓			30.00 (+1.81)	39.89 (+2.21)
	✓		29.66 (+1.47)	40.79 (+3.11)
		✓	29.27 (+1.08)	38.50 (+0.82)
✓	✓		30.49 (+2.30)	41.18 (+3.50)
✓		✓	30.13 (+1.94)	40.93 (+3.25)
	✓	✓	31.94 (+3.75)	41.17 (+3.49)
✓	✓	✓	**32.16 (+3.97)**	**43.30 (+5.62)**

Table 3. Ablation study on the impact of training input sizes and model ensemble. All models are trained using the suggested modules in Table 2.

1280	1536	1920	AP	F1
✓			32.16	43.30
	✓		33.44 (+1.28)	44.10 (+0.80)
		✓	34.29 (+2.13)	44.85 (+1.55)
✓	✓		32.37 (+0.21)	44.08 (+0.78)
✓		✓	32.50 (+0.34)	43.98 (+0.68)
	✓	✓	33.81 (+1.65)	45.23 (+1.93)
✓	✓	✓	**35.33 (+3.17)**	**45.53 (+2.23)**

to enhance recall and maximize the precision-recall trade-off, F1 score optimization necessitates a more nuanced approach that addresses false positives and false negatives for a well-rounded performance.

4.3. Training Procedure

Style Transfer. By utilizing the summer-winter pre-trained weight, a pre-trained weight for image translating from summer to winter, we fine-tune it on our custom dataset, with 1080x1080 of the image size, for 100 epochs using Adam optimizer. The learning rate is set at 0.0002 and decayed by a lambda scheduler.

YOLO-World. In this phase, we conducted fine-tuning of the YOLO-World model [20], initializing it with COCO pre-trained weights on the FishEye8k dataset [1] for 100 epochs. We set the learning rate to 0.0002 and maintained the default configuration of the YOLO-World X model. Additionally, we applied augmentations such as rotation and flipping during the training process to enhance the model's robustness and generalization capabilities.

Object Detection. we utilize a COCO pre-trained weight for each YOLO model and fine-tune it on our generated dataset. Initially, we perform ablation study by training YOLOR-D6 [17], YOLOv7-E6E [19], YOLOv8x [20], and YOLOv9-E [21] for 50 epochs using the default training configuration. Upon identifying the best-performing model through the ablation study (Sec. 4.4, which is YOLOR-D6 in our case, we proceed with the actual training procedure.

For the final training procedure of YOLOR-D6, we fine-tune it on three different input sizes: 1280, 1536, and 1920 for 300 epochs using the SGD optimizer. The learning rate is set to 0.01 and decayed by a cosine scheduler at 0.0005. Additionally, during the later training phase, we incorporate the validation set of the FishEye8K dataset [1] into our training set.

4.4. Evaluation

Ablation Study. We conduct an ablation study to assess the impact of each proposed module. The baseline results are evaluated using a COCO pre-trained YOLOR-D6 [17] fine-tuned on FishEye8K [1] training and validation sets. Subsequently, we systematically incorporate all the proposed modules and report the performance results in Table 2. All models are trained with the input size of 1280×1280 for 50 epochs. We can observe that incorporating the BG, IGM, and OVPL modules significantly improves both the AP and F1 scores. However, the most substantial enhancement is observed when these modules are utilized together. We investigated the relationship between input size and detection accuracy in the second ablation study. We systematically increased the training and inference input size of the YOLOR-D6 model. Interestingly, we observed improvements in both AP and F1 scores when using a resolution of 1920. Finally, we tried an ensemble of all three models and observed a considerable boost in both AP and F1 scores. The ablation results are in Table 3.

Hyper-parameter Tuning. We conduct hyper-parameter tuning for both the training and inference pipelines. To determine the optimal values, we submit several results to the AIC24 - Track 4 evaluation system and obtain the metrics. Our submission results are summarized in Table 4. Initially, we focus on maximizing the AP score, which involves progressively lowering the confidence threshold for object detection. This strategy aims to increase the number of true positives captured while accepting the trade-off of potentially increasing false positives. The intention is to explore the detection space comprehensively, thereby enhancing recall and overall AP score. However, balancing these adjustments and the risk of introducing excessive false positives is essential, which could adversely affect precision and overall performance.

As the challenge progresses, the evaluation metric shifts to the F1 score. Consequently, our subsequent submissions prioritize achieving a higher F1 score. To accomplish this, we gradually increase the confidence threshold to reduce false positives and potentially enhance precision. Moreover,

Table 4. Hyper-parameter tuning using the testing set from AIC24 - Track 4. The best results are shown in red, and the second-best are shown in blue.

Method	Train Size	Infer Size	Confidence	IoU	AP	AP@50	AP@S	AP@M	AP@L	F1
YOLOR-P6	1280	1280	0.0001	0.5	0.5801	0.8594	0.4125	0.7121	0.5581	0.1631
YOLOR-D6	1280	1280	0.00001	0.5	0.5867	0.8647	0.4350	0.7133	0.5704	0.1094
	1280	1280	0.0001	0.5	0.5861	0.8630	0.4341	0.7127	0.5697	0.3377
	1280	1280	0.001	0.5	0.5845	0.8602	0.4326	0.7109	0.5607	0.4773
	1280	1280	0.01	0.5	0.5844	0.8601	0.4330	0.7110	0.5605	0.4774
	1280	1280	0.1	0.5	0.5812	0.8522	0.4287	0.7086	0.5610	0.5486
	1280	1280	0.2	0.5	0.5789	0.8466	0.4259	0.7079	0.5610	0.5652
	1280	1280	0.3	0.5	0.5752	0.8395	0.4207	0.7072	0.5610	0.5822
	1280	1280	0.4	0.5	0.5714	0.8318	0.4163	0.7071	0.5605	0.5903
	1280	1280	0.5	0.5	0.5653	0.8189	0.4085	0.7065	0.5599	0.5976
	1280	1280	0.6	0.5	0.5519	0.7912	0.3900	0.7029	0.5599	0.5995
	1920	1920	0.6	0.5	0.5547	0.7765	0.3522	0.7041	0.5748	0.6134
	1920, 1536, 1280	1920	0.6	0.5	0.5690	0.8022	0.3943	0.7097	0.5716	0.6151
	1920, 1536, 1280	2560	0.6	0.5	0.5680	0.7933	0.3941	0.7134	0.5579	0.6194
	1920, 1536, 1280	2560	0.65	0.5	0.5462	0.7551	0.3776	0.6976	0.5489	0.6101
	1920, 1536, 1280	3200	0.6	0.5	0.5386	0.7572	0.3989	0.6739	0.5252	0.6061
	1920, 1536, 1280	varying	0.6	0.5	0.5684	0.8148	0.4274	0.7021	0.5606	0.6051
YOLOv7-E6E	1280	1280	0.0001	0.5	0.5679	0.8497	0.4016	0.6986	0.5953	0.1467
	1920, 1536, 1280	1280	0.65	0.5	0.5597	0.8045	0.3860	0.6934	0.5724	0.5933
	1920, 1536, 1280	2560	0.5	0.5	0.5773	0.8333	0.4271	0.7080	0.5640	0.5938
	1920, 1536, 1280	2560	0.65	0.5	0.5578	0.8001	0.3981	0.6921	0.5638	0.5942
	1920, 1536, 1280	varying	0.65	0.5	0.5801	0.8243	0.4103	0.7133	0.5714	0.6082
YOLOv8x	1280	1280	0.0001	0.5	0.5793	0.8553	0.4367	0.6927	0.5700	0.1688
YOLOv9-E	1280	1280	0.0001	0.5	0.5817	0.8585	0.4293	0.7072	0.5764	0.1586
	1280	1280	0.6	0.5	0.5702	0.8336	0.4181	0.7020	0.5540	0.5771

Table 5. The public leaderboard of AIC24 - Track 4.

Rank	Team ID	Team Name	F1
1	9	VNPT AI	0.6406
2	40	NetsPresso	0.6196
3	5	**SKKU-AutoLab (ours)**	**0.6194**
4	63	UIT-AICLUB	0.6077
5	15	SKKU-NDSU	0.5965
6	33	MCPRL	0.5883
7	156	zzl	0.5828
8	52	DeepDrivePL	0.5825
9	86	NCKU-ACVLAB	0.5637
10	13	FRDC-SH	0.5606

we observe that enhancing the input resolution can contribute to more accurate detection of smaller objects. Additionally, employing ensemble techniques can further enhance accuracy, even if only marginally. Our final solution is an ensemble of three YOLOR-D6 models, each trained on input sizes of 1280, 1536, and 1920. We set the input size to 2560 during inference and maintain a confidence threshold of 0.6.

Comparison. We submit our final solution to the evaluation system of AIC24 - Track 4. The rankings are summarized in Table 5. Our solution achieves an F1 score of 0.6194 and ranks 3rd among 52 teams. Notably, our score has a marginal difference compared to the 2nd rank, with only 0.002.

5. Conclusion

This paper introduces an innovative approach to the 2D road object detection task tailored specifically for fish-eye cameras. Our method incorporates various techniques to adapt general object detection knowledge to the fish-eye image domain. Firstly, we deploy an image generation module to reconcile the representation differences between daytime and nighttime images. Secondly, we leverage the state-of-the-art zero-shot open-vocabulary object detection model, YOLO-World, to facilitate semi-pseudo-labeling. We trained and evaluated several single-stage object detection models with the modified dataset. The final solution of our method was evaluated on the AIC24 - Track 4 dataset, achieving the third ranking with an F1 score of 0.6194, demonstrating the efficacy of our approach.

References

[1] M. Gochoo, M.-E. Otgonbold, E. Ganbold, J.-W. Hsieh, M.-C. Chang, P.-Y. Chen, B. Dorj, H. A. Jassmi, G. Batnasan, F. Alnajjar, M. Abduljabbar, and F.-P. Lin, "Fisheye8k: A

benchmark and dataset for fisheye camera object detection," in *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, June 2023, pp. 5305–5313. 1, 2, 3, 4, 5, 6

[2] S. Wang, D. C. Anastasiu, Z. Tang, M.-C. Chang, Y. Yao, L. Zheng, M. S. Rahman, M. S. Arya, A. Sharma, P. Chakraborty, S. Prajapati, Q. Kong, N. Kobori, M. Gochoo, M.-E. Otgonbold, G. Batnasan, F. Alnajjar, P.-Y. Chen, J.-W. Hsieh, X. Wu, S. S. Pusegaonkar, Y. Wang, S. Biswas, and R. Chellappa, "The 8th AI City Challenge," in *IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2024. 1, 5

[3] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu, "Ua-detrac: A new benchmark and protocol for multi-object detection and tracking," *Computer Vision and Image Understanding*, vol. 193, p. 102907, 2020. 2

[4] Z. Luo, F. Branchaud-Charron, C. Lemaire, J. Konrad, S. Li, A. Mishra, A. Achkar, J. Eichel, and P.-M. Jodoin, "Mio-tcd: A new benchmark dataset for vehicle classification and localization," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5129–5141, 2018. 2

[5] C. H. Bahnsen and T. B. Moeslund, "Rain removal in traffic surveillance: Does it matter?" *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 2802–2819, 2019. 2

[6] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang, "Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8789–8798. 2

[7] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5967–5976. 2, 4

[8] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, "Learning to generate images of outdoor scenes from attributes and semantic layouts," *arXiv preprint arXiv:1612.00215*, 2016. 2

[9] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," in *International Conference on Compututer Vision*, 2017, pp. 6836–6845. 2

[10] C. Gao, Q. Liu, Q. Xu, L. Wang, J. Liu, and C. Zou, "Sketchycoco: Image generation from freehand scene sketches," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5173–5182. 2

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," pp. 2242–2251, 2017. 2, 4

[12] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Conference on Neural Information Processing Systems*, vol. 30, 2018. 3

[13] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *European Conference on Computer Vision*, 2018. 3

[14] H.-H. Nguyen, P.-T. Tran-Huu, and H. M. T. S. V.-U. Ha, "Improved optical flow estimation in traffic monitoring system," in *2013 Third World Congress on Information and Communication Technologies (WICT 2013)*, 2013, pp. 165–169. 3

[15] L. H. Pham, H. N. Phan, N. M. Chung, T.-A. Vu, and S. V.-U. Ha, "A robust multiclass vehicle detection and classification algorithm for traffic surveillance system," in *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, 2020, pp. 1–6. 3

[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788. 3

[17] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "You only learn one representation: Unified network for multiple tasks," *Journal of Information Science and Engineering*, 2023. 3, 5, 6

[18] G. Jocher, "YOLOv5 by Ultralytics," May 2020. [Online]. Available: https://github.com/ultralytics/yolov5 3

[19] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7464–7475. 3, 5, 6

[20] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," January 2023. [Online]. Available: https://github.com/ultralytics/ultralytics 3, 5, 6

[21] C.-Y. Wang and H.-Y. M. Liao, "Yolov9: Learning what you want to learn using programmable gradient information," 2024. 3, 5, 6

[22] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, and J. Zhu, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," *arXiv preprint arXiv:2303.05499*, 2023. 3

[23] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "Yolo-world: Real-time open-vocabulary object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2024, pp. 1–15. 3, 4, 5

[24] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*, 2016, pp. 694–711. 4

[25] D. Shanmugam, D. Blalock, G. Balakrishnan, and J. Guttag, "Better aggregation in test-time augmentation," in *International Conference on Compututer Vision*, 2021, pp. 1194–1203. 5

[26] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*, 2014, pp. 740–755. 5