What, How, and When Should Object Detectors Update in Continually Changing Test Domains?

Supplementary Material

A. Additional Details for Baselines

We provide additional implementation details for each baseline model. Our framework incorporates all baseline models using the official code except *Mean-Teacher*. The results of the experiments are reported based on the optimal hyperparameters that yield the best results in our scenario.

ActMAD [31] As ActMAD exclusively conducts experiments on the KITTI dataset, where all images have a constant height and width (e.g., 370 x 1224), ensuring consistent feature map sizes for all samples. ActMAD can easily align them along the spatial axis. However, in the general setting of object detection tasks, such as the COCO benchmark set, where image sizes and width-to-height ratios vary, aligning feature maps along the spatial axis becomes challenging due to different sizes. To adapt ActMAD to our COCO $\rightarrow$ COCO-C scenario, we perform center cropping on the feature maps to match the size of training domain feature maps and the current test sample feature maps. We employ a learning rate of 1e-5 for COCO and 1e-4 for SHIFT, respectively.

Mean-Teacher As the official code of TeST [38] is not available, we implement the EMA-updated Teacher and Student models following TeST [38], to conduct experiments in our scenarios. TeST involves three forward steps for a batch: forwarding weakly augmented samples through the student network, strong augmented samples through the teacher network, and original samples through the teacher network for outputs. However, for a fair comparison, we perform two forward steps, forwarding the original sample through the teacher network and strong augmented samples through the student network, to make predictions before adaptation for every samples. We utilize a learning rate of 1e-5 and set the EMA update rate for the teacher network to 0.999.

NORM [37] We set the hyperparameter N that controls the trade-off between training statistics and estimated target statistics as 128.

DUA [30] We set the momentum decay as 0.94, minimum momentum constant as 1e-4, and the initial momentum decay as 1e-3.

B. Effects of Bottleneck Reduction Ratio in Adaptors

Table 5 shows the results for COCO $\rightarrow$ COCO-C, SHIFT-Discrete, and SHIFT-Continuous based on the dimension reduction ratio (r) discussed in Section 3.2, representing the ratio of bottleneck size compared to the input size in the adaptor. The adaptation performance remains consistent across different r values. However, in the case of $r = 1$ in SHIFT experiments, mAP decreases, potentially due to catastrophic forgetting resulting from a large number of adaptable parameters. Since increasing the value of r significantly reduces the number of learnable parameters and memory usage, we set r to 32 in all other experiments.

Table 5. Comparison of adaptation performance (mAP), the number of trainable parameters (# Params), and memory usage (Cache) according to r of Sec. 3.2, the bottleneck reduction ratio in the adaptor. We set r as 32 for all our experiments in the main paper. SD / SC denotes SHIFT-Discrete / Continuous, respectively.

		mAP			# Params		Cache	
Backbone	r	COCO	SD	SC	Num	Ratio	Avg.	Max
Swin-T	1	22.6	40.0	21.3	4.33M	15.7%	0.75	7.51
	2	22.6	40.3	23.2	2.17M	7.85%	0.73	7.27
	4	22.6	40.4	23.2	1.09M	3.95%	0.70	7.06
	8	22.6	40.4	23.2	0.55M	2.00%	0.69	7.00
	16	22.6	40.4	23.2	0.28M	1.02%	0.67	6.98
	32	22.6	40.4	23.2	0.15M	0.54%	0.65	6.96
	64	22.6	40.4	23.2	0.08M	0.29%	0.65	6.95
ResNet50	1	22.5	38.7	20.8	6.31M	26.7%	1.55	5.89
	2	22.4	38.7	20.9	3.16M	13.4%	1.51	5.64
	4	22.3	38.6	21.3	1.59M	6.71%	1.49	5.52
	8	22.3	38.6	21.4	0.80M	3.39%	1.48	5.46
	16	22.2	38.6	21.4	0.41M	1.73%	1.48	5.43
	32	22.2	38.7	21.4	0.21M	0.89%	1.48	5.41
	64	22.1	38.7	21.3	0.11M	0.48%	1.48	5.40

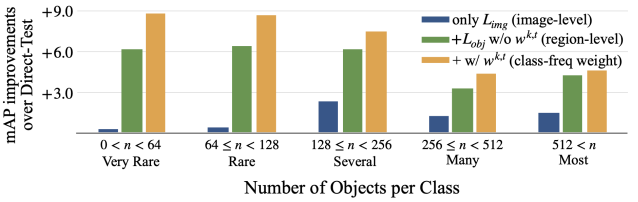


Figure 6. Adaptation performance by class object frequencies.

C. Results Based on Class Frequencies

To demonstrate the effectiveness of our alignment strategy as detailed in Eq.4, which utilizes class frequency as a weight for region-level feature alignment, we report the accuracy according to class frequencies in Fig.6. Our method boosts rare class accuracy by 6% with region-level alignment (L_{obj}, Eq.4), and an additional 9% by applying frequency weights ($w^{k,t}$, Eq.4). This improvement arises from avoiding the alignment of rare features with common class

Table 6. Comparison of mAP on COCO → COCO-C based on whether environmental indicators for test domain changes are provided. **TTA** (Test-Time Adaptation) reflects scenarios where information of test domain changes is provided to the model, allowing the adaptor to be initialized accordingly. **CTA** (Continual Test-Time Adaptation), the focus of our main paper, on the other hand, describes more challenging situations where no information about test domains is provided, preventing the model from initializing adaptors for new domains and compelling it to use a single adaptor for all domains.

		Noise			Blur				Weather				Digital					
Backbone	Method	Gau	Sht	Imp	Def	Gls	Mtn	Zm	Snw	Frs	Fog	Brt	Cnt	Els	Px	Jpg	Org.	Avg.
Swin-T [28]	TTA	13.6	15.6	14.7	14.3	14.0	15.1	9.0	23.8	27.5	37.6	36.9	27.9	27.0	23.1	23.4	42.5	22.9
	CTA	13.6	16.6	16.1	14.0	13.6	14.2	8.3	23.7	27.2	37.4	36.4	27.2	27.2	22.2	22.3	42.3	22.6
ResNet50 [11]	TTA	12.7	15.1	14.0	13.9	10.7	11.6	6.5	22.5	26.3	39.0	38.9	26.2	23.0	21.8	22.5	43.6	21.8
	CTA	12.7	17.8	17.5	12.4	11.5	11.3	6.6	22.8	26.9	38.6	38.5	28.0	25.1	21.2	22.2	41.8	22.2

Table 7. Comparison of mAP, the number of backward and forward passes, FPS, and memory usage between baselines and our models on the continually changing KITTI datasets (*Fog* → *Rain* → *Snow* → *Clear*). Our models improve mAP@50 by 15.1 and 11.3 for Swin-T and ResNet50 backbone, respectively, compared to *Direct-Test* while maintaining comparable FPS. All experiments are conducted with a batch size of 16.

		mAP@50					# For. Steps	# Backward Steps					FPS	Cache	
Backbone	Method	Fog	Rain	Snow	Clear	Avg.	All	Fog	Rain	Snow	Clear	All	Avg.	Avg.	Max
Swin-T	Direct-Test	46.9	69.5	28.7	89.6	58.7	936	0	0	0	0	0	24.7	0.4	5.5
	ActMAD	53.3	78.1	41.2	90.7	65.8	936	234	234	234	234	936	16.8	0.8	21.9
	Mean-Teacher	54.5	80.2	43.2	92.4	67.6	1,872	234	234	234	234	936	10.0	1.0	22.6
	Ours	56.7	82.1	64.6	91.8	73.8	936	234	234	234	234	936	17.1	0.4	11.8
	Ours-Skip	57.4	81.5	64.3	91.3	73.6	936	234	65	224	36	559	22.9	0.4	11.8
ResNet50	Direct-Test	33.4	63.5	29.8	88.6	53.8	936	0	0	0	0	0	27.7	0.8	4.3
	NORM	38.4	66.4	35.9	87.3	57.0	936	0	0	0	0	0	27.7	0.8	4.3
	DUA	34.8	67.7	30.9	89.0	55.6	936	0	0	0	0	0	27.7	0.8	4.3
	ActMAD	40.4	66.5	42.7	84.5	58.5	936	234	234	234	234	936	18.5	1.6	22.6
	Mean-Teacher	39.6	71.3	43.5	88.2	60.6	1,872	234	234	234	234	936	11.1	1.8	31.1
	Ours	45.6	71.4	52.5	88.3	64.5	936	234	234	234	234	936	18.8	0.8	9.4
	Ours-Skip	45.8	71.3	50.9	88.4	64.1	936	234	111	98	45	488	24.5	0.8	9.4

features and intensifying alignment of rare classes features.

D. Effects of Batch Size

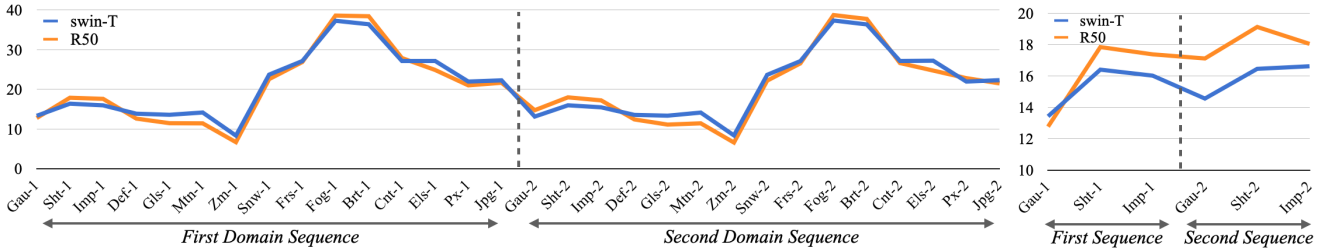
In the main paper, all experiments were conducted with a batch size of 4. However, due to the nature of test-time online adaptation, the online adaptation should remain effective even with smaller batch sizes, allowing for immediate processing of incoming data. Therefore, we conducted experiments with extremely small batch sizes, such as 1 and 2. For ResNet50, the mAP scores are 22.2, 22.1, and 21.6 at batch sizes of 4, 2, and 1, respectively. Similarly, for Swin-T, the mAP scores are 22.6, 22.5, and 22.1, demonstrating consistent performance across batch sizes.

E. Effects of Initialization on Adaptor: Comparing TTA and CTA Performance

To understand the initialization effects on adaptors, we conduct ablation study where an indicator of test domain changes is provided from envinronment directly. With this signal, adaptors can be initialized for each new test domain, akin to Test-Time Adaptation (TTA) [18, 45, 49], which is easier than Continual Test-Time Adaptation (CTA) as it adapts models to a single, i.i.d. test domain. However, for the ResNet50 backbone, reinitializing adaptors at each domain change lower the average mAP by 0.4, resulting in 21.8, as shown in Tab. 6. Despite the more challenging CTA scenarios compared to TTA, we speculate the mAP is higher because some sequential domains with similar characteristics provide a beneficial initialization for the adaptors. In Tab. 6, for the 'Shot' and 'Impulse Noise' domains, which are similar to previous ones, the performance drop from CTA to TTA is sharper at 2.7 and 3.5, respectively; otherwise, the impact is minimal. This highlights the importance of good adaptor initialization, despite their quick adaptibility. While the average mAP of CTA is lower than that of TTA for the Swin-T backbone, the mAP performance trends across domain sequence follow a similar pattern.

F. Analyzing Adaptor Memorization for Previously Encountered Domains

To evaluate the memorization capability of adaptors for previously encountered domains, we conduct experiments on COCO-C in two scenarios: (1) repeating a long sequence that includes all 15 various corrupted test domains (Fig. 7a), and (2) repeating a short sequence consisting of only the 'Gaussian → Shot → Implulse' domains, which share similar characteristics (Fig. 7b). In scenario (1), both the ResNet50 and swin-T backbone show negligible performance difference between the first and second sequences



(a) Long sequence repetition including all 15 corrupted test domains

(b) Short sequence repetition

Figure 7. Performance comparison of adaptors in memorization capability for previously encountered domains.

for, suggesting that adaptors may not retain knowledge of domains encountered earlier in a long sequence, leading to no improvement upon revisiting domains. In contrast, in scenario (2), there is a notable average performance increase of 2.1 for ResNet50 and 0.7 for swin-T in the second sequence, indicating that adaptors can remember and leverage knowledge from previously seen, similar domains, resulting in a performance boost. This reveals that while adaptors possess a rapid adaptation capability, they also have a tendency to forget domain knowledge over long periods, underscoring the significance of similarity between immediately consecutive domains.

G. Further Dataset: Results on KITTI

We conduct additional experiments on the KITTI [8] dataset, the commonly used object detection dataset consisting of driving scenes with 8 classes (car, van, truck, person, person sitting, cyclist, tram, misc). To simulate the continually changing domains, we use the following scenario (*Fog* $\rightarrow$ *Rain* $\rightarrow$ *Snow* $\rightarrow$ *Clear*) as done in [31]. We use the physics-based rendered dataset [10] for *fog* and *rain* and simulate *snow* using the corruption library from [12]. We use the same split of [31], which divides the 7,441 training samples into 3,740 training and 3,741 test samples. We train the Faster-RCNN using 3,741 training samples representing the *Clear* attribute with Swin-Transformer and ResNet50 backbones, and evaluate it sequentially on *Fog*, *Rain*, *Snow*, and *Clear* test samples.

We conduct all experiments with a batch size of 16 on 1 RTX A6000 GPU. Table 7 shows the mAP@50, the number of forward and backward steps, FPS, and memory usage (Cache). *Ours* improves the mAP@50 by 15.1 and 10.7 for Swin-T and ResNet50 backbones, respectively, compared to *Direct-Test*. Compared to *ActMAD* and *Mean-Teacher*, our model not only improves the adaptation performance but also reduces memory usage, as we update only an extremely small number of parameters of the adaptor. Furthermore, using our skipping criteria of Sec. 3.4 with $\tau = 1.1$ and $\beta = 1.05$, we can improve FPS by more than 5.8 without sacrificing mAP@50, resulting in much faster inference speed compared to other TTA baselines.

H. Further Object Detector: Results on FCOS

To validate our methods across different object detectors, we conduct experiments with FCOS, a representative anchor-free one-stage detector, using the ResNet50 backbone on COCO $\rightarrow$ COCO-C. Unlike RPN-based detectors, FCOS directly predicts foreground classes directly at each feature map position, excluding the background class. For FCOS, we modify the object-level feature extraction method described in Eq. 3 of the main paper by filtering out features with a maximum class probability above 0.3, as follows:

$$F_{te}^{k,t} = \left\{ f_{te}^{t} \middle| \underset{c}{\text{argmax}}(p_{fg}) = k, \underset{c}{\max}(p_{fg}) > 0.3 \right\}, \quad \text{where } h_{cls}(f_{te}^{t}) = [p_{fg}] = [p_0, ..., p_{C-1}]. \tag{6}$$

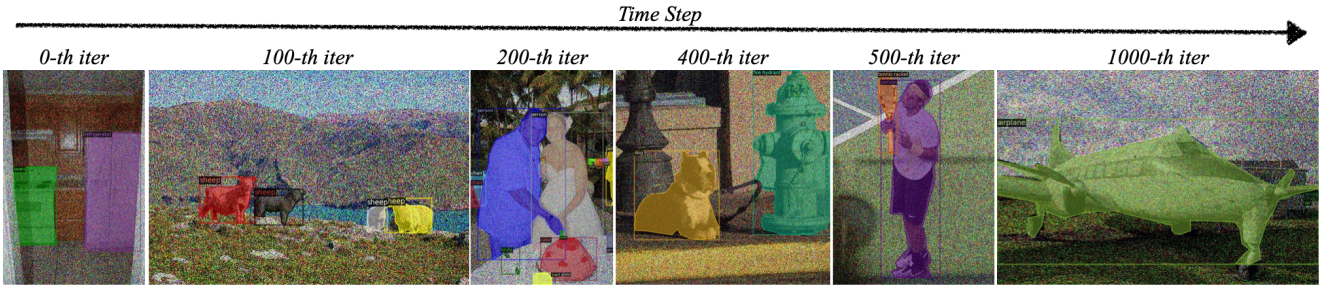
The rest of the method is the same as described in the main paper. When the average mAP for *Direct-Test* of FCOS is 11.8, the adaptation performance increases to 13.5 (+1.7) with *NORM*, decreases to 10.9 (-0.7) with *DUA*. Our method achieves 16.2 (+ 4.4), and with Ours-Skip, it reaches 15.7 (+3.9) requiring only 14.6% backward steps. This underscores the efficiency and efficacy of our online adaptation method across different object detector types.

I. Qualitative Results

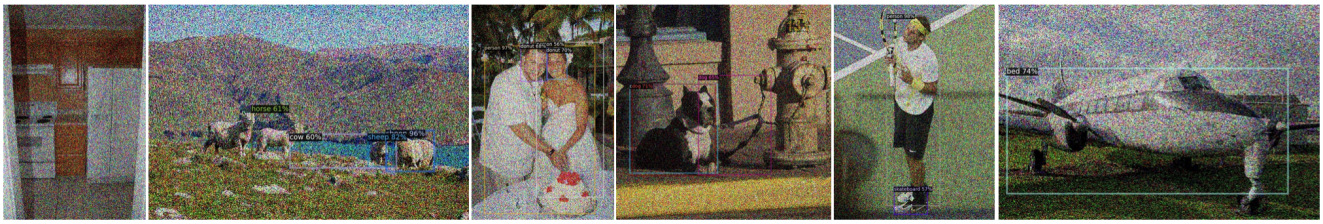
Figs. 8, 9, and 10 show the qualitative results of *Ours* and *Direct-Test* which predict the samples without adaptation for COCO $\rightarrow$ COCO-C and SHIFT, respectively.

I.1. COCO $\rightarrow$ COCO-C

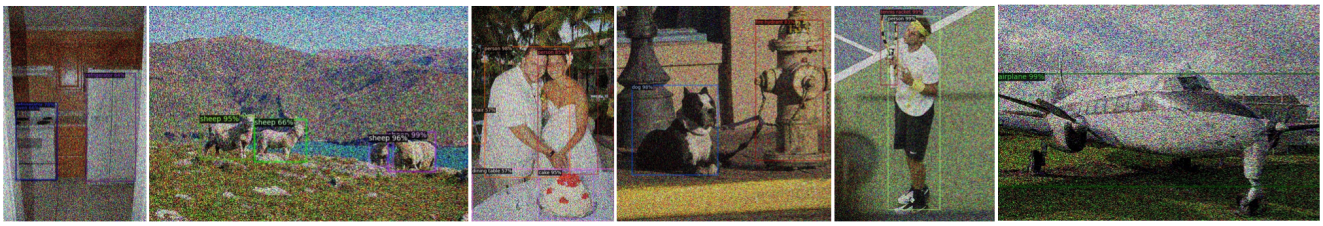
Figs. 8 and 9 compare the prediction results for COCO images corrupted. When the model encounters test images with various corruptions sequentially (*Gaussian-Noise* $\rightarrow$ *Shot-Noise* $\rightarrow$ *Impulse-Noise* $\rightarrow$ *Defocus-Blur* $\rightarrow$ *Glass-Blur* $\rightarrow$ *Motion-Blur* $\rightarrow$ *Zoom-Blur* $\rightarrow$ *Snow* $\rightarrow$ *Frost* $\rightarrow$ *Fog* $\rightarrow$ *Brightness* $\rightarrow$ *Contrast* $\rightarrow$ *Elastic-Transform* $\rightarrow$ *Pixelate* $\rightarrow$ *JPEG-Compression* $\rightarrow$ *Original*), Fig. 8 and



(a) GT bounding boxes.



(b) Prediction results of *Direct-Test*.



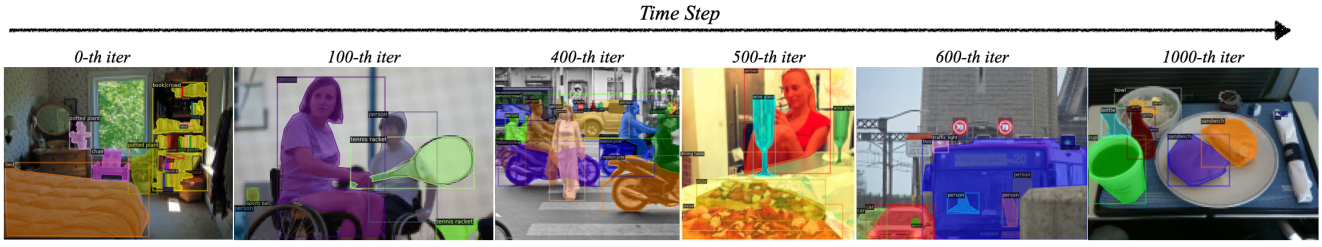
(c) Prediction results of *Ours*.

Figure 8. Results of COCO images corrupted by *Shot-Noise*. In the analysis of Sec. 4.5, we conjecture that *Ours* largely skips adaptation in *Shot-Noise* domain, despite the low mAP of *Direct-Test*, because the model has already adapted to a similar domain, *Gaussian-Noise*. In (c), at the first step before adaptation to the *Shot-Noise*, our model already predicts 'Oven' and 'Refrigerator' which *Direct-Test* fails to detect. This results in a much faster adaptation, and *Ours* successfully detects various objects, including rare ones such as 'Fire Hydrants', in the remaining images of the *Shot-Noise* domain.

9 shows the results when the test images are corrupted by *Shot-Noise* and *Pixelate*, respectively. Compared to *Direct-Test*, our model adapts to the current domain within a few steps, such as 100 iterations, and detects various objects very well in the remaining incoming images.

I.2. SHIFT-Discrete

Fig. 10 shows the qualitative results for SHIFT-Discrete. In the SHIFT-Discrete scenario, the model encounters environments sequentially, transitioning from *cloudy* $\rightarrow$ *overcast* $\rightarrow$ *foggy* $\rightarrow$ *rainy* $\rightarrow$ *dawn* $\rightarrow$ *night* $\rightarrow$ *clear*. Figure. 10 selectively shows the *foggy* $\rightarrow$ *rainy* $\rightarrow$ *dawn* $\rightarrow$ *night* sequence, where the domain gap from the original *clear* environments is relatively large. Compared to *Direct-Test*, *Ours* detects various objects such as 'cars' and 'pedestrians' regardless of distribution changes.



(a) GT bounding boxes.

(b) Prediction results of *Direct-Test*.

(c) Prediction results of *Ours*.

Figure 9. Results for COCO images corrupted by *Pixelate*. In the *Pixelate* domain, where the model has already experienced various corruptions in a long sequence, *Ours* initially incorrectly detects objects. In (c), it misidentifies a bed as a couch in the first step. However, it rapidly adapts to the *Pixelate* domain and effectively detects various objects. Notably, even in cases where *Direct-Test* correctly identifies objects but with low confidence, *Ours* detects them with much higher confidence.

(a) GT bounding boxes.

(b) Prediction results of *Direct-Test*.

(c) Prediction results of *Ours*.

Figure 10. Results for SHIFT-Discrete with continually changing attributes, *foggy* → *rainy* → *dawn* → *night*.