Unsupervised Domain Adaptive Detection with Network Stability Analysis
– Supplementary Material –

In the supplementary material, we include the detailed procedure to generate different disturbances (Sec. A), the illustration of instance graph in InsD (Sec. B), adaptation results from $C \rightarrow F$ with all 8 categories, ablation study on different disturbance weights of $NSA-UDA$ loss and qualitative results and comparison of NSA-UDA and its baselines (Sec. C).

A. Procedure for Disturbance Generation

In this work, we regard the discrepancy in distribution between two domains as data disturbance and consider the three types of disturbances, i.e, Heavy and Light Image-level Disturbances (HID and LID) and Instance-level Disturbance (InsD). HID focuses on the large range of object changes in view and scale with random texture and color variations. LID mainly represents object variations in small scale and translation, and InsID represents variations of objects of the same class in style, scale and view. As shown Alg. 1, we utilize common data augmentations to simulate the above disturbances. In Fig. 2, we show variations of objects under HID, LID and InsD with several examples.

Algorithm 1 Generation Processes about Three Types of Disturbances(i.e, HID, LID and InsID)

```
Input: Original input image $x$, $S_{\text{HID}}$, $V_{\text{HID}}$, $S_{\text{LID}}$, $D_{\text{LID}}$.
/* HID Generation */
    (1). The texture and color of object are changed by transformation operations, i.e., AdjustGamma, AdjustSaturation, AdjustContrast, AdjustHue, AdjustSharpness, AdjustBrightness, Identity, Equalize and Solarize.
    (2). By randomly sampling the change value $m_{\text{HID}}$ from $[1, S_{\text{HID}}]$, we further zoom in or out of the input image with it. Meanwhile, $m_{\text{HID}}$ is recorded in $T_{\text{HID}}$, which is used to save related parameters during simulating HID.
    (3). The resized image is cropped at center position with the same size of the original input image by using CenterCrop operation. Similarly, coordinate information of the cropped roi is also recorded in $T_{\text{HID}}$.
    (4). The input image is flipped horizontally at random by using RandomHorizontallyFlip operation. In addition, the flipping state is also recorded in $T_{\text{HID}}$.
/* LID Generation */
    (1). The texture and color of object are changed by using some transformation operations, i.e, AdjustGamma, AdjustSaturation, AdjustContrast, AdjustHue, AdjustSharpness and AdjustBrightness.
    (2). By randomly sampling the change value $m_{\text{LID}}$ from $[1, S_{\text{LID}}]$, we further zoom in or out of the input image with it. Meanwhile, $m_{\text{LID}}$ is recorded in $T_{\text{LID}}$, which is used to save related parameters during simulating LID.
    (3). By randomly sampling the deviation distance and computing the deviation rate $b_{\text{LID}}$ for different feature layers, which is required to be in $[0, D_{\text{LID}}]$, the input image is further shifted, and $b_{\text{LID}}$ is recorded in $T_{\text{LID}}$.
    (4). The resized image is cropped at center position with the same size of the original input image by using CenterCrop operation, and the coordinate information of the cropped RoI is also recorded in $T_{\text{LID}}$.
/* InsD Generation */
    (1). The texture and color of object are changed by using some transformation operations, i.e, AdjustGamma, AdjustSaturation, AdjustContrast, AdjustHue, AdjustSharpness and AdjustBrightness.
Output: Perturbed images, i.e., $x_{\text{HID}}$, $x_{\text{LID}}$ and $x_{\text{InsD}}$, transformation parameters, i.e, $T_{\text{HID}}$ and $T_{\text{LID}}$.
```

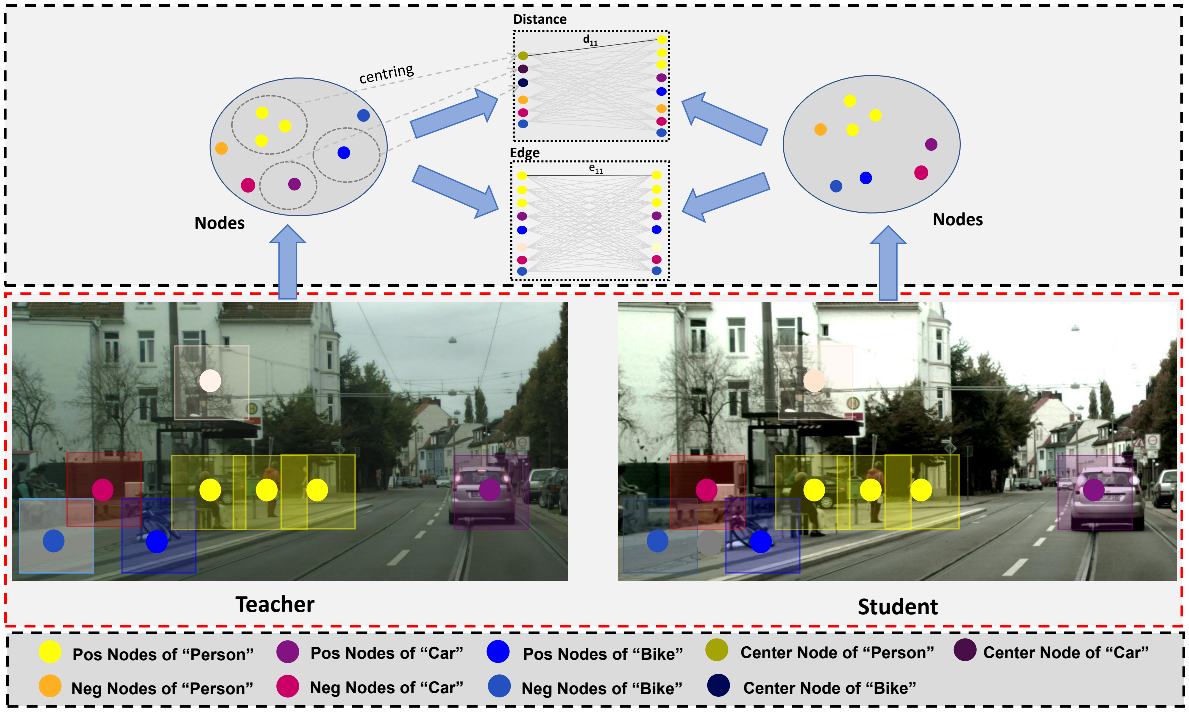


Figure 1. Visualization of the process of building instance graph $G(V, E, D)$.

B. Instance Graph

In this work, we build the instance graph from each of internal pixel-level and/or instance-level feature layers. Then, we further analyze the relationship among nodes, centers of all classes and similar negative nodes collected from background areas on feature space by using the contrastive loss function. For instance-level feature layers (i.e, RCNN of Faster-RCNN detector), the undirected graph can be built directly based on proposals. For pixel-level feature layers, we first obtain the proposals of instance-level features by using the slide window method, and choose nodes of objects with the restraint condition of object areas and proposal nodes of negative samples from background areas with $W_t = 1$. By the way, the areas of objects of nodes are required to be in $(\alpha_1 s \times \alpha_1 s, \alpha_2 s \times \alpha_2 s)$, where α_1 and α_2 are constant coefficients (α_1 and α_2 are empirically set to 1 and 7), and s is the stride of feature layers.

C. Adaptation Results

As shown in Tab. 1, we report the detection accuracy for each category from $C \rightarrow F$. It can be observed that our NSA-UDA obtain the best mAP score of 52.7%. Meanwhile, our NSA-UDA achieves the best performance on 3 of 8 categories from $C \rightarrow F$.

D. Qualitative results of NSA-UDA

Fig. 3, 4 and 5 provide some qualitative results of our NSA-UDA with three training stages in different adaptations. It verifies that our NSA-UDA method can achieve state-of-the-art performance on training data of not only single source domain but also source and target domains in various complex scenarios.

References

[1] Qi Cai, Yingwei Pan, Chong-Wah Ngo, Xinmei Tian, Lingyu Duan, and Ting Yao. Exploring object relation in mean teacher for cross-domain detection. In *CVPR*, 2019.

[2] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *CVPR*, 2021.

[3] Mengzhe He, Yali Wang, Jiaxi Wu, Yiru Wang, Hanqing Li, Bo Li, Weihao Gan, Wei Wu, and Yu Qiao. Cross domain object detection by target-perceived dual branch distillation. In *CVPR*, 2022.

Table 1. Experiments from **C**→**F** using average precision (AP, in %). The best two results are highlighted in **red** and **blue** fonts, respectively, for all state-of-the-art comparison tables.

Method	Backbone	person	rider	car	truck	bus	train	mbike	bicycle	mAP
Baseline	VGG-16	17.8	23.6	27.1	11.9	23.8	9.1	14.4	22.8	18.8
MOTR [1] [CVPR'2019]	ResNet-50	30.6	41.4	44.0	21.9	38.6	40.6	28.3	35.6	35.1
GPA [8] [CVPR'2020]	ResNet-50	32.9	46.7	54.1	24.7	45.7	41.1	32.4	38.7	39.5
CFFA [10] [CVPR'2020]	VGG-16	43.2	37.4	52.1	34.7	34.0	46.9	29.9	30.8	38.6
DSS [7] [CVPR'2020]	ResNet-50	42.9	51.2	53.6	33.6	49.2	18.9	36.2	41.8	40.9
D-adapt [4] [ICLR'2022]	VGG-16	44.9	54.2	61.7	25.6	36.3	24.7	37.3	46.1	41.3
UMT [2] [CVPR'2021]	VGG-16	56.5	37.3	48.6	30.4	33.0	46.7	46.8	34.1	41.7
MeGA-CDA [6] [CVPR'2021]	VGG-16	37.7	49.0	52.4	25.4	49.2	46.9	34.5	39.0	41.8
TIA [9] [CVPR'2022]	VGG-16	52.1	38.1	49.7	37.7	34.8	46.3	48.6	31.1	42.3
SDA [11] [arXiv'2021]	VGG-16	38.3	47.2	58.8	34.9	57.7	48.3	35.7	42.0	45.2
TDD [3] [CVPR'2022]	VGG-16	39.6	47.5	55.7	33.8	47.6	42.1	37.0	41.4	43.1
SIGMA [5] [CVPR'2022]	VGG-16	46.9	48.4	63.7	27.1	50.7	35.9	34.7	41.4	43.5
Baseline w. Data Aug. (Ours)	VGG-16	39.0	45.8	47.5	21.4	33.9	10.0	34.3	41.8	34.2
NSA-UDA (Ours)	VGG-16	50.2	60.1	67.7	37.4	57.4	46.9	47.3	54.3	52.7
Oracle (S1)	VGG-16	48.1	53.6	69.0	32.7	54.0	24.9	41.3	49.6	46.7
Oracle (S2)	VGG-16	50.1	57.9	70.8	39.6	59.2	48.5	44.4	53.2	53.0

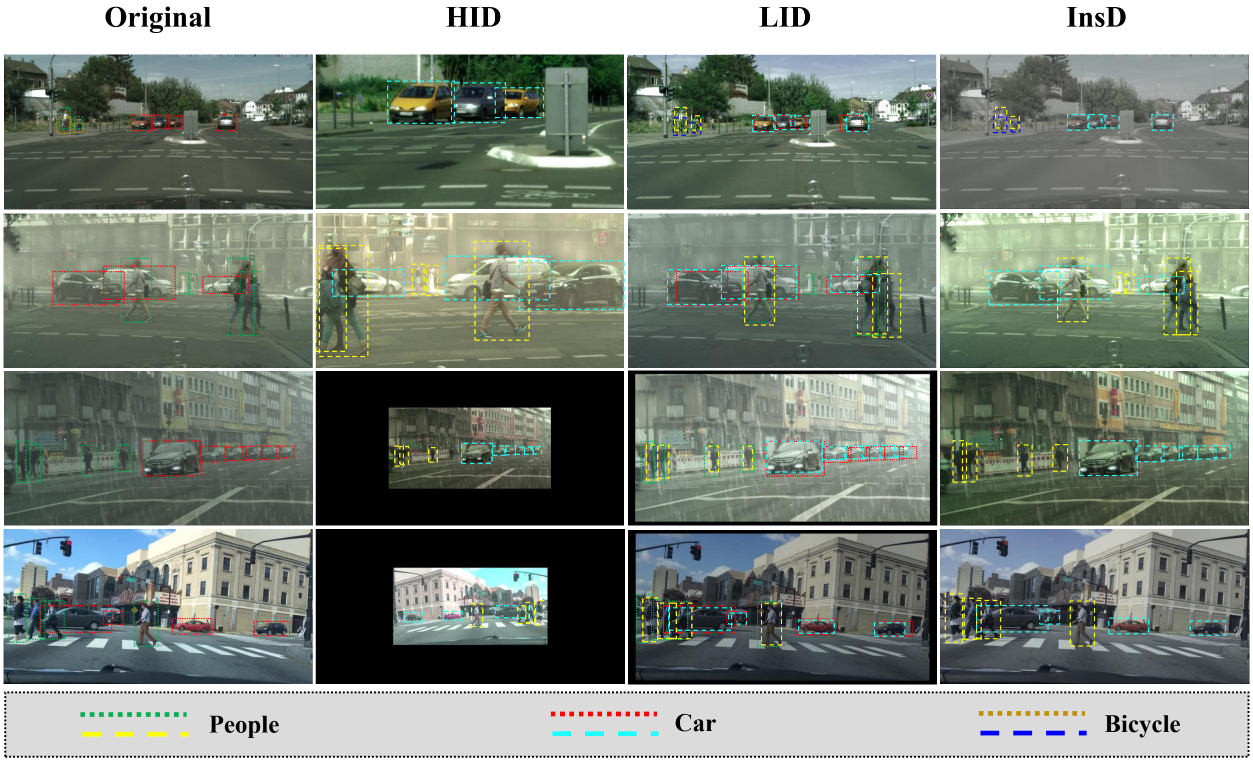


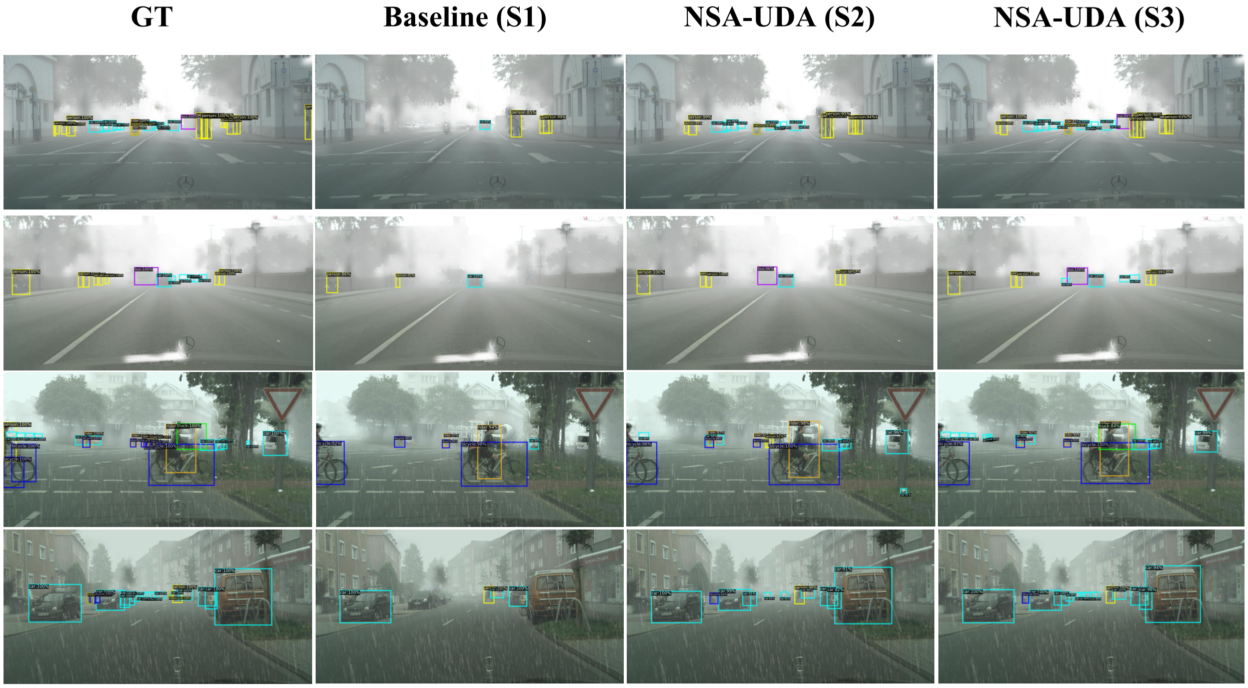
Figure 2. Visualization of several examples of HID, LID and InsD. The dashed rectangles in the images represent the objects under disturbances

[4] Junguang Jiang, Baixu Chen, Jianmin Wang, and Mingsheng Long. Decoupled adaptation for cross-domain object detection. In *ICLR*, 2022.

[5] Wuyang Li, Xinyu Liu, and Yixuan Yuan. Sigma: Semantic-complete graph matching for domain adaptive object detection. In *CVPR*, 2022.

[6] Vibashan VS, Vikram Gupta, Poojan Oza, Vishwanath A. Sindagi, and Vishal M. Patel. Mega-cda: Memory guided attention for category-aware unsupervised domain adaptive object detection. In *CVPR*, 2021.

[7] Yu Wang, Rui Zhang, Shuo Zhang, Miao Li, Yangyang Xia, Xishan Zhang, and Shaoli Liu. Domain-specific suppression for adaptive object detection. In *CVPR*, 2021.

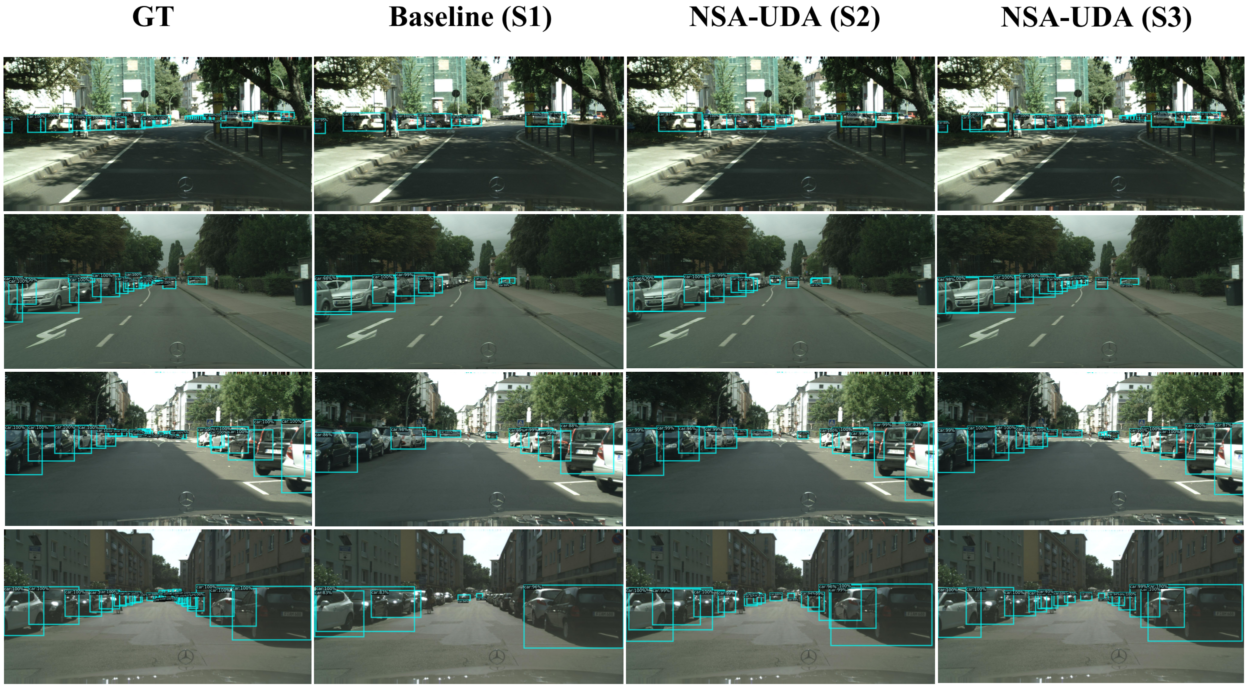

GT

Baseline (S1)

NSA-UDA (S2)

NSA-UDA (S3)

Figure 3. Qualitative results of our NSA-UDA with three training stages from Weather adaptation. The first two lines and the last two lines are respectively from C→F and C→R.


GT

Baseline (S1)

NSA-UDA (S2)

NSA-UDA (S3)

Figure 4. Qualitative results of our NSA-UDA with three training stages from Cross-Camera adaptation and Synthetic-to-Real adaptation. The first two lines and the last two lines are respectively from K→C and M→C.

[8] Minghao Xu, Hang Wang, Bingbing Ni, Qi Tian, and Wenjun Zhang. Cross-domain detection via graph-induced prototype alignment.

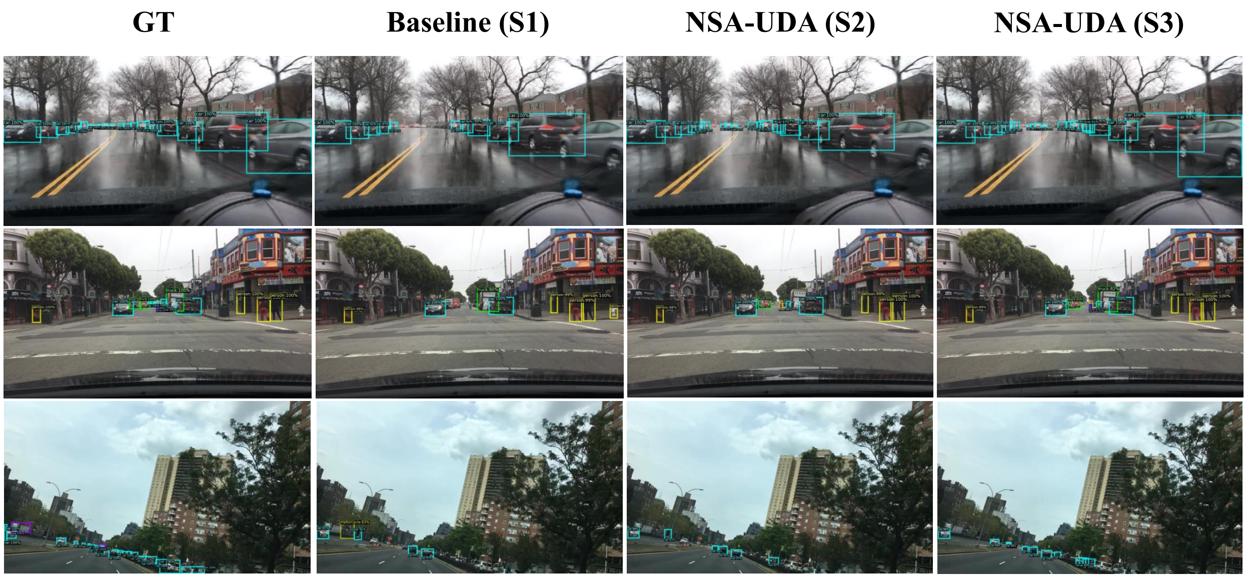


Figure 5. Qualitative results of our NSA-UDA with three training stages from Small-to-Large adaptation.

In *CVPR*, 2020.

[9] Liang Zhao and Limin Wang. Task-specific inconsistency alignment for domain adaptive object detection. In *CVPR*, 2022.

[10] Yangtao Zheng, Di Huang, Songtao Liu, and Yunhong Wang. Cross-domain object detection through coarse-to-fine feature adaptation. In *CVPR*, 2020.

[11] Qianyu Zhou, Qiqi Gu, Jiangmiao Pang, Zhengyang Feng, Guangliang Cheng, Xuequan Lu, Jianping Shi, and Lizhuang Ma. Self-adversarial disentangling for specific domain adaptation. *arXiv*, 2021.