I. Related Work

Reflection removal methods generally can be divided into two categories, *i.e*., single-image reflection removal (SIRR) and multi-image reflection removal (MIRR). We focus on learning-based SIRR in this paper, some MIRR [1]–[15] and traditional SIRR [16]–[21] methods are listed for reference. In the following, we first review the learning-based SIRR methods and relevant datasets, then briefly introduce the domain generalization and network interpolation methods.

A. Learning-based SIRR Methods

For removing reflection accurately and producing clear results, Fan *et al*. [22] designed a gradient loss to constrain the transmission edge maps. Zhang *et al*. [23] further proposed an exclusion loss to minimize the gradient correlation of reflection / transmission layers. To alleviate the misalignment problem, Wei *et al*. [24] leveraged the highest-level VGG [25] features which are less sensitive to misalignment. Reflection distribution is also considered to facilitate SIRR. Dong *et al*. [26] predicted a reflection confidence map which is integrated into a composition loss. Li *et al*. [27] further defined an explicit mask loss by leveraging reflection strengths. In this paper, we enhance the SIRR task from a domain generalization perspective, which is orthogonal to and can cooperate well with the above methods. It is worth mentioning that Chang *et al*. [28] deployed loss functions over several synthesized samples with the same transmission layer but different reflections, which is also aimed to enhance the generalization ability. However, they are still limited by the static model and the compromise of learning objectives.

Various network structures have also been explored for SIRR. CEILNet presented by Fan *et al*. [22] is a two-stage architecture, where the edge maps are predicted before the estimation of the transmission layer. DMGN [29], Dong *et al*. [26], and RAGNet [27] followed this two-stage setting and estimated the reflection layer in the prior stage. Chen *et al*. [30] also adopted two-stage architecture but recovered the missing information due to saturated color at first. Zhang *et al*. [23] stacked the features extracted by multiple layers of a pre-trained VGG-19 [25] model at the beginning of SIRR models, which is retained in a series of subsequent works [24], [29]. Yang *et al*. [31] proposed to predict the reflection layer and the transmission layer iteratively. IBCLN [32] further introduced recurrent neural networks, which is also exploited in [26], [29]. Prasad *et al*. [33] proposed a multi-scale structure which starts from a low-resolution input and progressively grows into the desired resolution. It is worth noting that the backbones in this paper can cover most of these architectures.

B. Single-Image Reflection Removal Datasets

In the early studies on learning-based SIRR, due to the lack of real-world training images, synthesis methods were introduced to train the SIRR models [22], [31]. Subsequent works also tried to synthesize more realistic reflection-contaminated images by considering camera pose [23], generative adversarial networks [34], rendering software [35], and physical models [36].

Meanwhile, several real-world datasets have been collected. Zhang *et al*. [23], Li *et al*. [32], and Wei *et al*. [24] captured three real-world datasets with 109, 220, and 450 image pairs, respectively. For testing, a real-world benchmark dataset termed as SIR^2 has been collected [37], which is composed of three subsets, *i.e*., *Wild*, *Solid*, and *Postcard*. Interestingly, Wan *et al*. [38] collected 3,250 reflection layers by putting a black piece of paper behind the glass, which provides another way for synthesis. Lei *et al*. [39] collected a perfectly aligned testing dataset by capturing raw images and obtained the transmission layer by subtracting the reflection layer from the reflection-contaminated image.

In summary, multiple datasets have been built with diverse reflection types and scenes, and the non-negligible domain gaps make it a critical problem to generalize and adapt to various domains for enhancing SIRR.

C. Domain Generalization

Domain generalization aims at learning a model for generalizing to out-of-distribution samples. It has attracted upsurging attention since it was introduced by Blanchard *et al*. [40]. Recent domain generalization methods improve generalization from one or more aspects of data manipulation, representation learning, and learning strategies. The category most relevant to this paper in domain generalization is ensemble learning [41]–[45]. These methods generally train domain-specific branches or networks, and fuse the features [43], [44] or predictions [42], [45] via a predicted weight during inference. Ding *et al*. [41] learned a domain-invariant model by distilling domain-specific models under structured low-rank supervision. Most domain generalization methods focus on high-level vision tasks. In this paper, we explore the SIRR task from a domain generalization perspective, hoping to encourage the application of domain generalization methods in low-level vision tasks. Please refer to [46], [47] for a comprehensive review of domain generalization methods.

D. Network Interpolation

Checkpoint ensemble [48] is a commonly used network interpolation strategy in recent years, which interpolates the parameters from different iterations for more stable training and enhanced performance. Chen *et al*. [49] proposed to learn the residual of parameters in a pre-trained model with a different yet related learning objective, and performed model interpolation in the objective-space. Wang *et al*. [50] analyzed the interpolation between two separately trained networks, and achieved continuous imagery effect transition. However, the interpolation weights of these methods should be manually tuned. In this paper, we leverage the power of domain generalization, and predict the interpolation weights for generalizing to unseen domains.

II. More Details of the Proposed Method

In this section, we first have an analysis on existing reflection removal datasets, which motivates us to solve the SIRR problem from a domain generalization perspective. Then, given domain

Table I: Comparison between the collection configuration of commonly used reflection removal datasets.

Dataset	Acquisition Method	Dataset Partition	Camera Model	Apertures Range	Camera/Glass Angle	Acquisition Environment	Piece of Glass (Thickness)	# of Samples $\mathbf{T}$ (and $\mathbf{I}$)	Alignment of $\mathbf{I}$ and $\mathbf{T}$
Syn_{CEIL}	synthetic	train	-	-	front	indoor + outdoor	-	7,643 (∞)	perfect
Syn_{zhang}	synthetic	train	-	-	front + oblique	indoor + outdoor	-	13,700 (∞)	perfect
Unaligned	real-world	train	a DSLR	-	front + oblique	indoor + outdoor	1 (-)	250	$\sim$20 pixels shift
Real89/20	real-world	train/test	Canon EOS 600D	f/1.6 - f/2	front + oblique	indoor + outdoor	1 (-)	89/20	aligned
Nature200/20	real-world	train/test	Canon EOS 750D	f/4 - f/16	front + oblique	indoor + outdoor	2 (3/8 mm)	200/20	aligned
Wild	real-world	test	Nikon D5300	unconstrained	unconstrained	indoor + outdoor	3 (3/5/10 mm)	55	aligned
SIR^2 *Postcard*				f/11 - f/32	controlled	indoor		199	
Solid								200	

experts of various reflection types, a reflection type-aware weighting (RTAW) module is deployed for predicting the expert-wise weights, and we propose an in-domain expert (IDE) loss for training RTAW. Two adaptive network combination (AdaNEC) methods are proposed for adaptive inference and generalization to unknown target domains.

A. Domain Gaps between SIRR Datasets

Preliminary. As revealed in [51] and [36], the two parts of light from the transmission layer and the reflection layer are linearly superimposed to form the reflection-contaminated image. Therefore, the acquisition of reflection-contaminated images can be formally formulated by,

$$\mathbf{I} = \mathcal{I}(\mathbf{\Omega}\mathbf{T}^{raw} + \mathbf{\Phi}\mathbf{R}^{raw}), \tag{1}$$

where $\mathbf{T}^{raw}$ and $\mathbf{R}^{raw}$ are the raw signals of transmission and reflection layers. $\mathbf{\Omega}$ and $\mathbf{\Phi}$ represent the refractive and reflective amplitude coefficient maps determined by the glass characteristics (*e.g.*, material, thickness, color) and photographing parameters (*e.g.*, aperture, focal length, relative positions against the glass surface). $\mathcal{I}$ denotes the camera image signal processing (ISP) pipeline, which is also distinct in different cameras. We note that typical ISP pipelines are non-linear to obtain sRGB images consistent with human sense.

Synthetic Datasets. Most existing synthesis methods synthesize training images by approximating Eqn. (1). For example, Fan *et al*. [22] assumed the reflection layers are out-of-focus and darker than the transmission layers, thus they took random blurriness and brightness attenuation into consideration when building the Syn_{CEIL} dataset. Zhang *et al*. [23] further considered gamma correction, and simulated the camera posture via a fixed Gaussian mask in Syn_{Zhang}. However, these methods consider a limited (and different) range of parameters, and more importantly, they operate on non-linear sRGB images. Therefore, there exist obvious domain gaps between synthetic datasets, as well as between synthetic and real-world datasets.

Real-world Datasets. In order to compensate for the domain shift of synthetic datasets, several real-world datasets are also collected [23], [24], [32], [37]. The configurations to collect these datasets are summarized in Tab. I. As one can see, the key factors relevant to the reflection type like camera model, aperture, glass, and alignment are diverse for these datasets, leading to the domain gaps between existing real-world datasets. It is worth noting that images from the same dataset are similar in acquisition circumstance (*e.g.*, environment and equipment), thus we can regard them as a domain and introduce domain-level AdaNEC for better reflection removal (see Sec. II-E).

B. Overall Solution

With a unified formulation (*i.e.*, Eqn. (1)), it seems reasonable for existing SIRR methods to train SIRR models with a mixture of various datasets [24], [27], [32]. However, as shown in Sec. II-A, there exist clear domain gaps between these datasets. These models have to compromise between different training source domains, where the trained models are often not within the scope of target testing sets. Therefore, the analysis in Sec. II-A motivates us to solve the SIRR problems in a domain generalization scheme.

In particular, suppose that there are N training datasets, where the i-th dataset is denoted by $\mathcal{S}_i$. We train an SIRR model as the expert for each of the dataset. In this case, we can obtain N domain experts $\{G_i\}_{i=1}^{N}$, where the expert G_i (with parameter θ_i) is specialized on the relevant reflection types in $\mathcal{S}_i$. Then, for a reflection-contaminated image $\mathbf{I}$, we can combine these domain experts for better reflection removal according to certain weights. A reflection type-aware weighting (RTAW) module is introduced to evaluate the expertise level and predict the expert-wise weights (*e.g.*, w_i) for each domain expert G_i on $\mathbf{I}$. Two representative adaptive network combination (AdaNEC) methods are incorporated to combine these experts for better handling the reflection-contaminated images from different domains. In the following, we introduce the details of the designing and training of RTAW, as well as the proposed AdaNEC methods.

C. Reflection Type-aware Weighting Module

As shown in Fig. 1, the proposed reflection type-aware weighting (RTAW) module is comprised of two parts, *i.e.*, the feature extraction networks and a cross-domain attention module. For each domain expert G_i, we obtain the domain-specific feature vector $\mathbf{k}_i$ from the reflection-contaminated image $\mathbf{I}$ via a corresponding extractor $\mathcal{F}_i$, *i.e.*, $\mathbf{k}_i = \mathcal{F}_i(\mathbf{I})$. An extra feature vector $\mathbf{q}$ is extracted by another extractor $\mathcal{F}$ for matching with the domain-specific ones. Inspired by the self-attention layer in Transformer [52], we present a cross-domain attention module (CDAM) to evaluate the expertise level of each domain expert by matching their feature vector $\mathbf{k}_i$ with $\mathbf{q}$, *i.e.*,

$$v_i = \left(\mathbf{W}_k^\top \mathbf{k}_i\right) \odot \left(\mathbf{W}_q^\top \mathbf{q}\right), \tag{2}$$

where $\mathbf{W}_k$ and $\mathbf{W}_q$ are parameters of fully-connected layers for $\mathbf{k}$ and $\mathbf{q}$ respectively, and $\mathbf{W}_k$ is shared by all of $\mathbf{k}_1 \sim \mathbf{k}_N$, $\odot$ denotes inner-product. Then for a testing image, all domain experts are exploited, while the RTAW weights can be calculated by using the expertise level, *i.e.*,

$$\mathbf{w} = \sigma(\mathbf{v}), \tag{3}$$

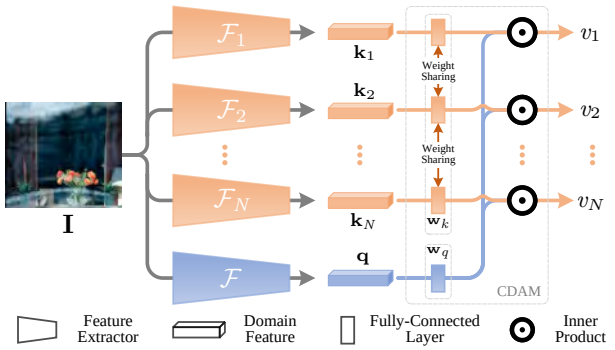


Figure 1: Structure of reflection type-aware weighting (RTAW) module. $\mathbf{I}$ denotes a reflection-contaminated image (the shown image is collected by Fan *et al*. [22]). $\mathbf{k}_i$ and $\mathbf{q}$ are vectors extracted by $\mathcal{F}_i$ and $\mathcal{F}$. v_i means expertise level of the i-th expert G_i on image $\mathbf{I}$, which can be used to calculate the expert-wise RTAW weights (*e.g.*, w_i).

where σ is the softmax operation and $\mathbf{v} = \{v_i\}_{i=1}^N$. In fact, the domain classifier used in Sec. II-A can take place of the RTAW module. However, it shows a worse training stability and performance, which is further shown in Sec. VII-A.

D. Training Scheme and Learning Objective

Apart from the reflection removal losses, an in-domain expert (IDE) loss is introduced for training the proposed RTAW module, and we introduce the two parts of the learning objective below. Note that parameters of the domain experts are fixed when training the RTAW module, and the details for training these domain experts can be found in Secs. III to V.

Training Scheme and Reflection Removal Losses. When training RTAW, apart from the datasets utilized to train the domain experts, there is no extra dataset to serve as the target domain, thus we follow the leave-one-domain-out strategy [53]. Specifically, given a training sample from the i-th training set $\mathcal{S}_i$, we denote the reflection-contaminated image as $\mathbf{I}_i$. $\mathbf{T}$ and $\mathbf{R}$ are the ground-truth transmission and reflection layers, respectively. The i-th source domain will serve as a pseudo target domain during this iteration. Only domain experts except for G_i will be employed, *i.e.*,

$$\mathbf{w}_i^{\complement} = \sigma(\mathbf{v}_i^{\complement}), \tag{4}$$

$$\hat{\mathbf{T}}_i^{\complement} = \sum\nolimits_{j\neq i} w_j \cdot G_j(\mathbf{I}_i), \tag{5}$$

where $\complement$ denotes the complement set and $\mathbf{v}_i^{\complement} = \{v_j\}_{j\neq i}$, $\mathbf{w}_i^{\complement} = \{w_j\}_{j\neq i}$. Note that the reflection layer $\hat{\mathbf{R}}_i^{\complement}$ is generated in a similar way if the backbone model requires it for loss calculation. In the following, we briefly introduce the reflection removal losses for training.

Fidelity Loss. Generally ℓ_1 or ℓ_2 is employed as the fidelity loss on the transmission layer or reflection layer,

$$\mathcal{L}_{fid} = \|\hat{\mathbf{Y}}_i^{\complement} - \mathbf{Y}\|_p, \mathbf{Y} \in \{\mathbf{T}, \mathbf{R}, \nabla\mathbf{T}\}, \tag{6}$$

where ∇ means gradient calculation operation, $p \in \{1, 2\}$, and $\mathbf{Y}$ denotes the corresponding ground-truth of $\hat{\mathbf{Y}}_i^{\complement}$.

Reconstruction Loss. For synthetic pairs, the synthesis methods can be leveraged for better constraining both outputs of transmission and reflection layers. Therefore, the reconstruction loss is defined as,

$$\mathcal{L}_{rec} = \|Syn(\hat{\mathbf{T}}_i^{\complement}, \hat{\mathbf{R}}_i^{\complement}) - \mathbf{I}_i\|_p, \tag{7}$$

where Syn denotes the differentiable synthesis methods.

VGG-based Loss. The VGG [25] architecture has been widely used for loss function design, *i.e.*,

$$\mathcal{L}_{VGG} = \sum\nolimits_l \lambda_l \left\| (\phi_l(\hat{\mathbf{T}}_i^{\complement}) - \phi_l(\mathbf{T}) \right\|_p, \tag{8}$$

where ϕ_l denotes the l-th layer of the VGG model, λ_l is the corresponding weight. The VGG-based loss is generally used following perceptual loss [54] for better image quality (denoted by $\mathcal{L}_{percept}$). In ERRNet [24], Wei *et al*. proposed to use the "conv5_2" feature for alignment-invariant loss ($\mathcal{L}_{inv}$).

Adversarial Loss. Adversarial loss [55] is an effective approach to enhancing the output image quality. Given a discriminator D, the adversarial loss can be defined by

$$\mathcal{L}_{adv} = -\log D(\hat{\mathbf{T}}_i^{\complement}). \tag{9}$$

To sum up, the reflection removal loss $\mathcal{L}_{RR}$ can be defined by combining these loss functions or their variants,

$$\begin{aligned}\mathcal{L}_{RR} =& \lambda_{fid}\mathcal{L}_{fid} + \lambda_{rec}\mathcal{L}_{rec} + \lambda_{percept}\mathcal{L}_{percept} \\ &+ \lambda_{inv}\mathcal{L}_{inv} + \lambda_{adv}\mathcal{L}_{adv},\end{aligned} \tag{10}$$

where λ_{fid}, λ_{rec}, $\lambda_{percept}$, λ_{inv}, and λ_{adv} are hyper parameters following the original settings in the backbone methods. Note that they can be zero if the corresponding loss function is not applied in a backbone method.

In-domain Expert Loss. With the aforementioned reflection removal losses, $\mathbf{w}_i^{\complement}$ is optimized towards better reflection removal performance, and w_i is left unconstrained in this iteration. However, we argue that the source domain of the training sample is a vital information overlooked by existing methods, and ask *what if the domain expert G_i is also deployed?* Since G_i is trained on $\mathcal{S}_i$, it can be regarded as an in-domain expert. Therefore, we can require that the final results mostly rely on it. In other words, for $\mathbf{w}$ (see Eqn. (3)), the i-th element w_i should be the largest. To this end, we propose an in-domain expert (IDE) loss, which is defined based on the cross-entropy (CE) loss [56], [57] as

$$\mathcal{L}_{IDE} = CE(\mathbf{w}, i) = -\log w_i. \tag{11}$$

Note that G_i is not actually involved during training. We empirically find that the IDE loss not only promotes the reflection removal performance but also makes the training more stable. Further discussions are given in Sec. VII-A.

To sum up, the learning objective for training RTAW is

$$\mathcal{L} = \mathcal{L}_{RR} + \lambda\mathcal{L}_{IDE}, \tag{12}$$

where λ is set to 0.1 for all three backbone methods. Please refer to Secs. III to V for more details about the loss function configurations on different backbones.

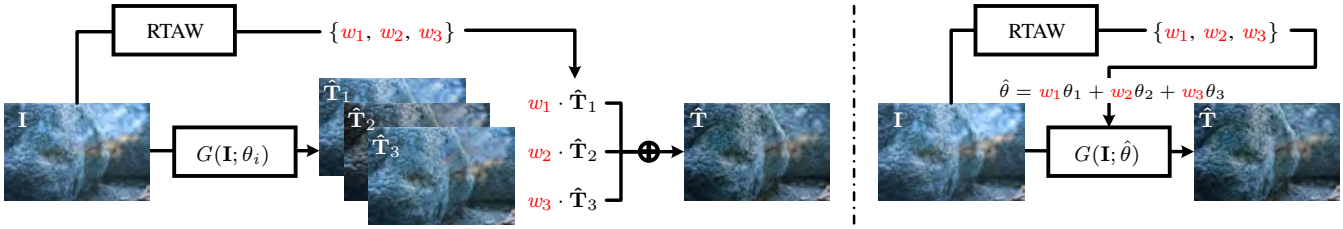


Figure 2: Illustration of the proposed two adaptive network combination (AdaNEC) methods, *i.e.*, output fusion (OF, on the left) and network interpolation (NI, on the right). We take the number of source domains $N = 3$ as an example.

Table II: Dataset configuration of the backbone methods.

Backbone	Training Datasets	Testing Datasets
ERRNet [24]	Syn_{CEIL}, *Real89*, *Unaligned*	*Real20*, SIR^2
IBCLN [32]	Syn_{Zhang}, *Real89*, *Nature200*	*Real20*, SIR^2, *Nature20*
RAGNet [27]	Syn_{CEIL}, Syn_{Zhang}, *Real89*	*Real20*, SIR^2

E. Adaptive Network Combination

As shown in Fig. 2, the RTAW weights can be employed for adaptively generalizing to target domains via adaptive network combination (AdaNEC). In this paper, we provide two representative AdaNEC methods termed output fusion (OF) and network interpolation (NI), respectively.

Output Fusion (OF). The OF manner is similar to the training procedure of RTAW, except that all domain experts are employed during inference to generate the output, *i.e.*,

$$\hat{\mathbf{T}}_{\texttt{OF}} = \sum\nolimits_{i=1}^{N} w_i \cdot G_i(\mathbf{I}), \tag{13}$$

where the results generated by the domain experts are fused as a final output. However, such a method requires multiple model inferences, which brings considerable extra computation burden and time consumption.

Network Interpolation (NI). To alleviate this problem, the NI manner interpolates the model parameters in advance, and the inference procedure is processed with the fused model parameters, which can be formulated by

$$\hat{\mathbf{T}}_{\texttt{NI}} = G(\mathbf{I}; \sum\nolimits_{i=1}^{N} w_i \cdot \theta_i), \tag{14}$$

where only a few extra computations (*i.e.*, the lightweight RTAW module and parameter interpolation) are introduced. Besides, as shown in Sec. II-A, generally the reflection-contaminated images in a dataset form a domain. In this case, we can use the average RTAW weights of a dataset (denoted by $\bar{w}_i$) for more stable prediction, where the w_i in Eqns. (13) and (14) is replaced by $\bar{w}_i$.

III. Learning Objective for ERRNet Backbone

Domain Experts. For training domain experts with ERRNet backbone [24], the loss functions are defined as follows.

Fidelity Loss

$$\begin{aligned}\mathcal{L}_{fid} =& 0.2\|\hat{\mathbf{T}} - \mathbf{T}\|_2^2 \\ &+0.4(\|\nabla_x\hat{\mathbf{T}} - \nabla_x\mathbf{T}\|_1 + \|\nabla_y\hat{\mathbf{T}} - \nabla_y\mathbf{T}\|_1).\end{aligned} \tag{15}$$

Perceptual Loss

$$\mathcal{L}_{percept} = \sum\nolimits_l \lambda_l \left\|\phi_l(\hat{\mathbf{T}}) - \phi_l(\mathbf{T})\right\|_1, \tag{16}$$

where $l \in \{conv2\_2, conv3\_2, conv4\_2, conv5\_2\}$, and the balancing weight $\lambda_l = 1$ for all these layers.

Adversarial Loss

$$\mathcal{L}_{adv} = -\log(D(\mathbf{T}, \hat{\mathbf{T}})) - \log(1 - D(\hat{\mathbf{T}}, \mathbf{T})), \tag{17}$$

which follows the relativistic adversarial loss [58].

Alignment-invariant Loss

$$\mathcal{L}_{inv} = \left\|\phi_h(\hat{\mathbf{T}}) - \phi_h(\mathbf{T})\right\|_1, \tag{18}$$

where h indicates the $conv5\_2$ layer.

Therefore, when training domain experts on Syn_{CEIL} and *Real89*, the learning objective is defined by

$$\mathcal{L}_{aligned} = \mathcal{L}_{fid} + 0.1\mathcal{L}_{percept} + 0.01\mathcal{L}_{adv}, \tag{19}$$

and the learning objective for domain expert on *Unaligned* dataset is defined by

$$\mathcal{L}_{unaligned} = 0.1\mathcal{L}_{inv} + 0.01\mathcal{L}_{adv}. \tag{20}$$

AdaNEC. The learning objective for training AdaNEC with ERRNet backbone is defined by

$$\mathcal{L} = \begin{cases} \mathcal{L}_{aligned} + 0.1\mathcal{L}_{IDE}, & \text{for } \textit{Real89}, \textit{Syn}_{CEIL} \\ \mathcal{L}_{unaligned} + 0.1\mathcal{L}_{IDE}, & \text{for } \textit{Unaligned} \end{cases}, \tag{21}$$

where $\hat{\mathbf{T}}_i^{\texttt{O}}$ takes place of $\hat{\mathbf{T}}$ in Eqn. (21).

IV. Learning Objective for IBCLN Backbone

Domain Experts. For training domain experts with IBCLN backbone [32], the loss functions are defined as follows.

Fidelity Loss

$$\mathcal{L}_{fid} = \sum\nolimits_{t=1}^{N} \|\hat{\mathbf{T}}_t - \mathbf{T}\|_2 + \|\hat{\mathbf{R}}_t - \mathbf{R}\|_2 \tag{22}$$

where t denotes the time step and $N = 3$.

Reconstruction Loss

$$\mathcal{L}_{rec} = \sum\nolimits_{t=1}^{N} \left\|Syn(\hat{\mathbf{T}}_t, \hat{\mathbf{R}}_t) - \mathbf{I}\right\|_2. \tag{23}$$

Perceptual Loss

$$\mathcal{L}_{percept} = \sum\nolimits_s \omega_s \sum\nolimits_l \lambda_l \left\|\phi_l(\hat{\mathbf{T}}_{\downarrow s}) - \phi_l(\mathbf{T}_{\downarrow s})\right\|_1, \tag{24}$$

where $l \in \{conv1\_2, conv2\_2\}$, and the balancing weights $\lambda_{conv1\_2} = 1/2.6$ and $\lambda_{conv2\_2} = 1/4.8$. $s \in \{1, 2, 4\}$, which means using the output/ground-truth with the $1/s$ size of the original image, where $\omega_1 = 1$, $\omega_2 = 0.8$, and $\omega_4 = 0.6$.

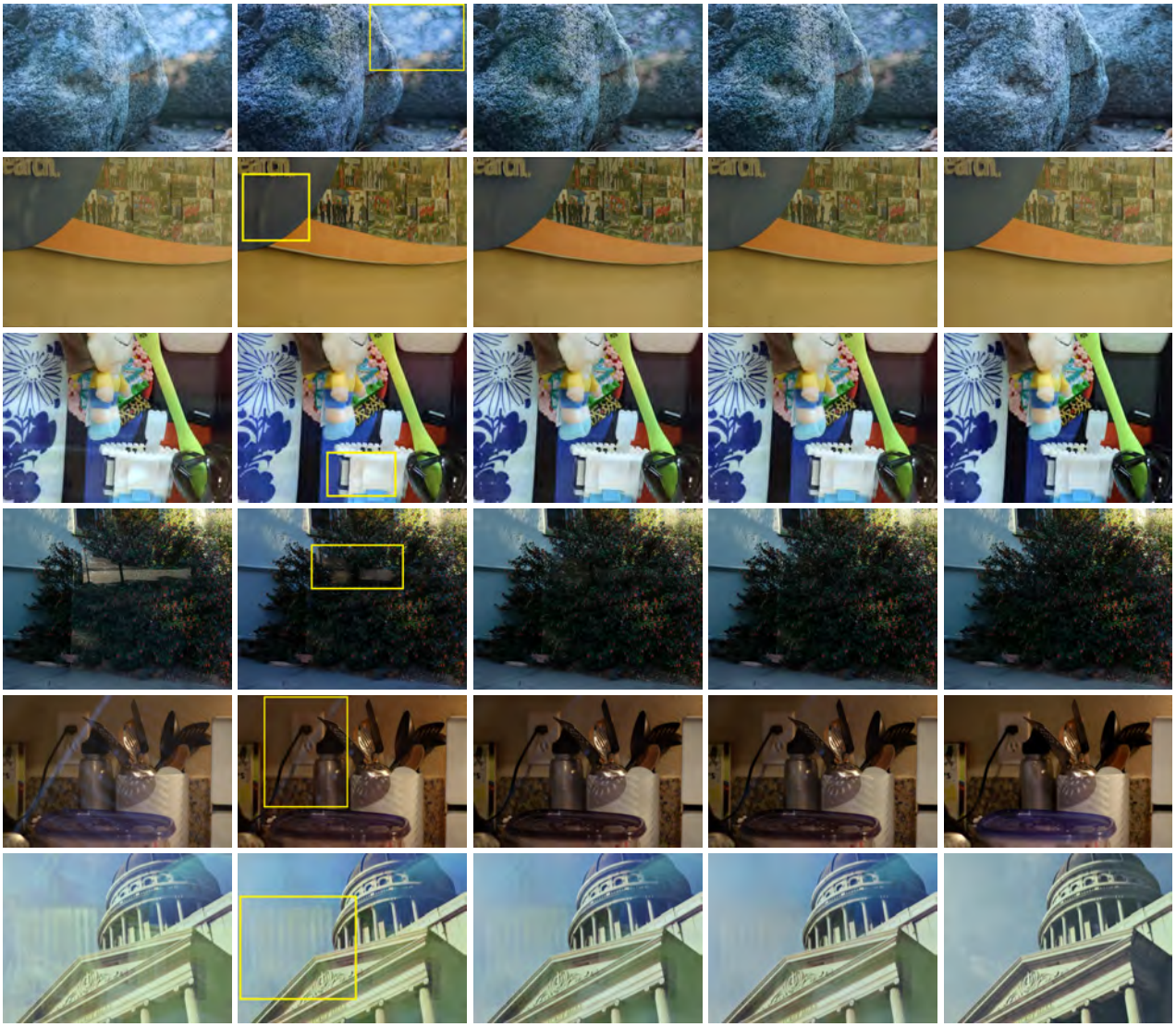


Figure 3: Visual comparison of ERRNet based AdaNEC methods.

Adversarial Loss

$$\mathcal{L}_{adv} = -\log(D(\hat{\mathbf{T}}, \mathbf{T})). \tag{25}$$

Therefore, when training domain experts on *Syn*$_{Zhang}$, *Real89* and *Nature200*, the learning objective is defined by

$$\mathcal{L}_{RR} = 2\mathcal{L}_{fid} + 2\mathcal{L}_{rec} + \mathcal{L}_{percept} + 0.01\mathcal{L}_{adv}. \tag{26}$$

AdaNEC. The learning objective for training AdaNEC with IBCLN backbone is defined by

$$\mathcal{L} = \mathcal{L}_{RR} + 0.1\mathcal{L}_{IDE}, \tag{27}$$

where $\hat{\mathbf{T}}_i^{\mathbb{C}}$ and $\hat{\mathbf{R}}_i^{\mathbb{C}}$ take place of $\hat{\mathbf{T}}$ and $\hat{\mathbf{R}}$ in Eqn. (27), respectively.

V. Learning Objective for RAGNet Backbone

Domain Experts. For training domain experts with RAGNet backbone [27], the loss functions are defined as follows.

Fidelity Loss

$$\mathcal{L}_{fid} = \|\hat{\mathbf{T}} - \mathbf{T}\|_1 + \kappa\|\hat{\mathbf{R}} - \mathbf{R}\|_1. \tag{28}$$

Perceptual Loss

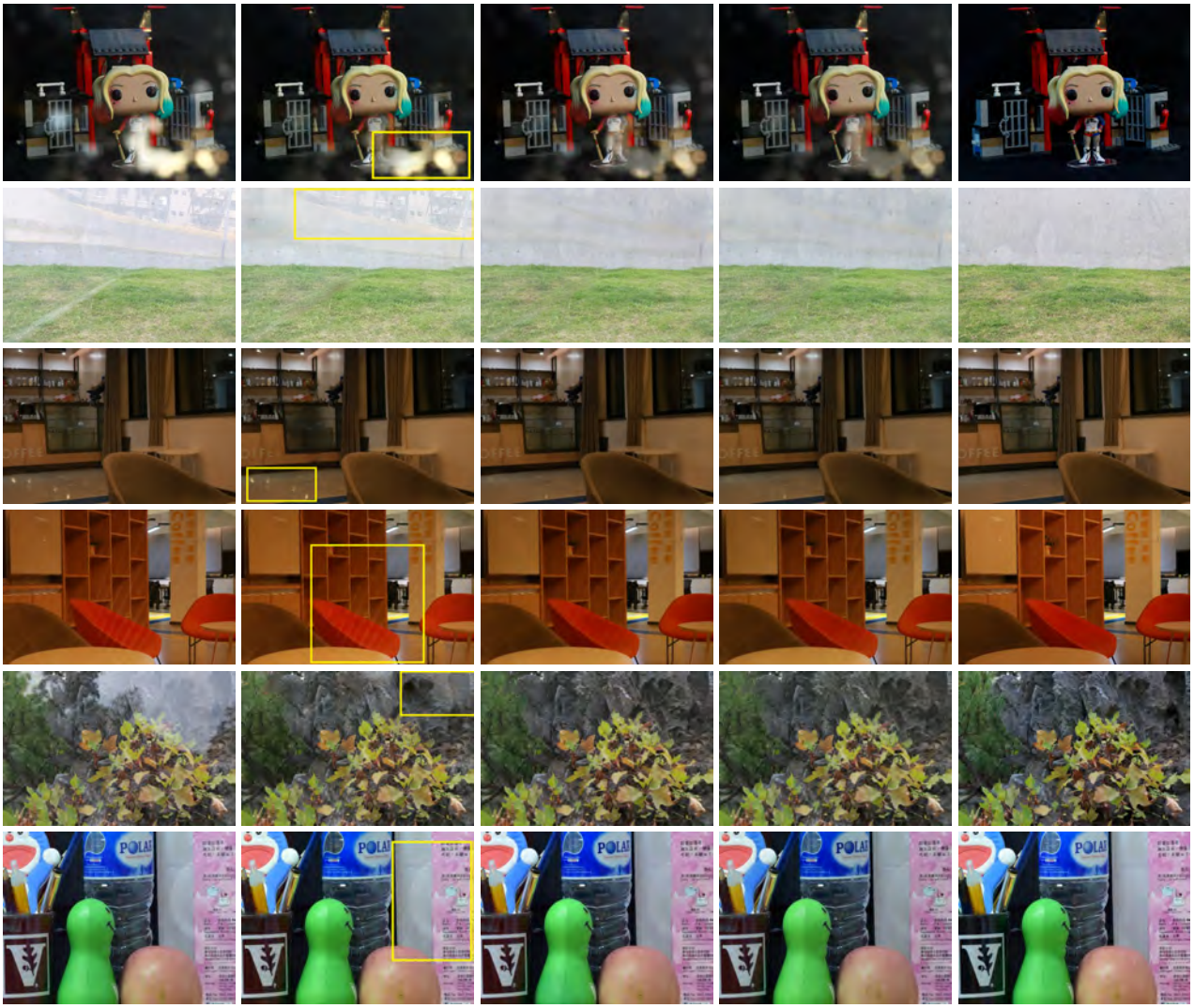
$$\begin{aligned}\mathcal{L}_{percept} = \sum\nolimits_l \lambda_l(&\left\|\phi_l(\hat{\mathbf{T}}) - \phi_l(\mathbf{T})\right\|_1 \\ &+\kappa\left\|\phi_l(\hat{\mathbf{R}}) - \phi_l(\mathbf{R})\right\|_1),\end{aligned} \tag{29}$$

where $l \in \{conv1\_2, conv2\_2, conv3\_2, conv4\_2, conv5\_2\}$, and the balancing weights λ_{conv1_2}, λ_{conv2_2}, λ_{conv3_2}, λ_{conv4_2}, and λ_{conv5_2} are set as 1/2.6, 1/4,8, 1/3.7, 1/5.6, and 10/1.5 following Zhang *et al*. [23], respectively.

Exclusion Loss Following [23], the exclusion loss is

$$\mathcal{L}_{\text{excl}} = \sum\nolimits_s \sqrt{\|\Psi(\hat{\mathbf{T}}_{\downarrow s}, \hat{\mathbf{R}}_{\downarrow s})\|_F}. \tag{30}$$

where $\Psi(\hat{\mathbf{T}}, \hat{\mathbf{R}}) = \tanh(\lambda_T|\nabla\hat{\mathbf{T}}|) \circ \tanh(\lambda_R|\nabla\hat{\mathbf{R}}|)$. λ_T and λ_R denote normalization factors. $\nabla\hat{\mathbf{T}}$ and $\nabla\hat{\mathbf{R}}$ are gradients of $\hat{\mathbf{T}}$ and $\hat{\mathbf{R}}$. $\|\cdot\|_F$ is the Frobenius norm. $\hat{\mathbf{T}}_{\downarrow s}$ and $\hat{\mathbf{R}}_{\downarrow s}$ represent



Input IBCLN [32] IBCLN$_{OF}$ IBCLN$_{NI}$ Ground-truth

Figure 4: Visual comparison of IBCLN based AdaNEC methods.

the $s\times$ down-sampling versions of $\hat{\mathbf{T}}$ and $\hat{\mathbf{T}}$. $s \in 1, 2, 4$, $\lambda_T = \frac{1}{2}$, and $\lambda_R = \|\nabla\hat{\mathbf{T}}\|_1/\|\nabla\hat{\mathbf{R}}\|_1$.

Mask Loss

$$\mathcal{L}_{mask} = \sum_{i=1}^{4}(\left\|\mathbf{M}_{diff}^{i}[\mathbf{R} > \varphi]\right\|_1 + \left\|\mathbf{M}^{i}[\mathbf{R} < \xi] - 1\right\|_1), \tag{31}$$

where $\mathbf{M}_{diff}^{i}$ and $\mathbf{M}^{i}$ are two masks from the i-th layer, which indicate the contribution of encoder features (*i.e.*, features from the reflection-contaminated input image) and decoder features (*i.e.*, features from the in-paining path) φ and ξ are thresholds set as 0.3 and 0.01, respectively. Please refer to [27] for more details.

Adversarial Loss

$$\mathcal{L}_{adv} = -\log(D(\hat{\mathbf{T}}, \mathbf{I})). \tag{32}$$

Therefore, when training domain experts on *Syn*$_{CEIL}$, *Syn*$_{Zhang}$, and *Real89*, the learning objective is defined by

$$\mathcal{L}_{RR} = \mathcal{L}_{fid} + \mathcal{L}_{percept} + \mathcal{L}_{mask} + 0.2\mathcal{L}_{excl} + 0.01\mathcal{L}_{adv}. \tag{33}$$

Particularly note that κ in Eqns. (28) and (29) is set as 1 on *Syn*$_{CEIL}$ and *Syn*$_{Zhang}$, while is set as 0 on *real89*.

AdaNEC. The learning objective for training AdaNEC with RAGNet backbone is defined by

$$\mathcal{L} = \mathcal{L}_{RR} + 0.1\mathcal{L}_{IDE}, \tag{34}$$

where $\hat{\mathbf{T}}_i^{\complement}$ and $\hat{\mathbf{R}}_i^{\complement}$ take place of $\hat{\mathbf{T}}$ and $\hat{\mathbf{R}}$ in Eqn. (34).

VI. Experiments

A. Implementation Details

Datasets. Since the core idea of this work is to make better use of the training sets and enhance SIRR from a domain generalization perspective, we follow the training configuration of the backbone methods and have no requirements on extra training datasets. In the following, the relevant training and testing datasets are briefly introduced.

***Syn*$_{CEIL}$** - The synthetic dataset proposed in CEILNet [22], where 7,643 image pairs are selected from the PASCAL VOC[1]

[1]http://host.robots.ox.ac.uk/pascal/VOC/

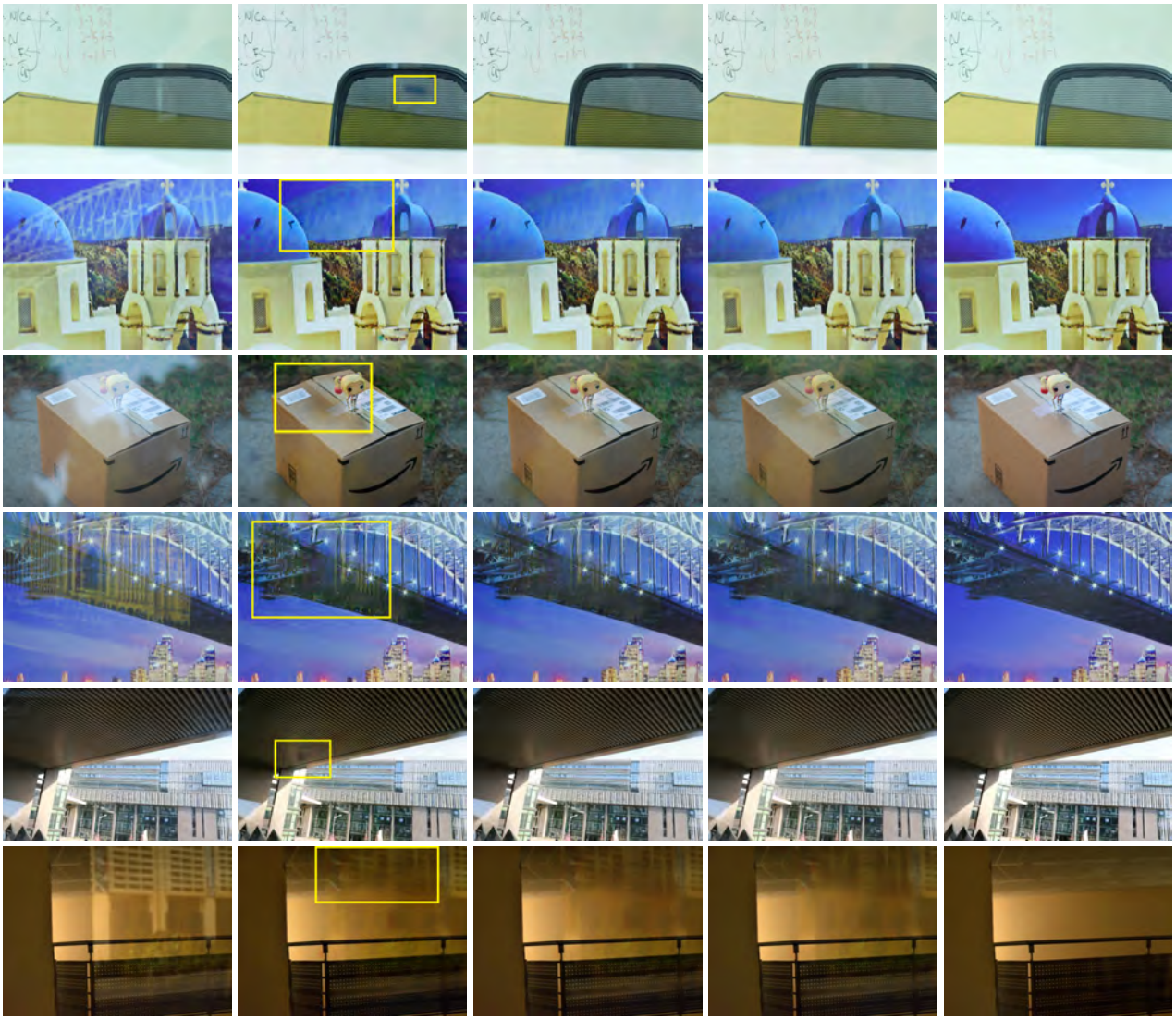


Figure 5: Visual comparison of RAGNet based AdaNEC methods.

dataset [59], and each pair serves as transmission and reflection layer, respectively. Blurriness and luminance of the reflection layer are considered for synthesis.

Syn$_{Zhang}$ - The synthetic dataset proposed by Zhang *et al*. [23], where 13,700 indoor-outdoor image pairs are collected from Flickr[2]. The synthesis method is adapted from *Syn$_{CEIL}$* and considers the camera pose for capturing images.

Real89 & Real20, Nature200 & Nature20 - Two well aligned real-world datasets captured by Zhang *et al*. [23] and Li *et al*. [32], where *Real89* and *Nature200* are used for training while *Real20* and *Nature20* are used for testing.

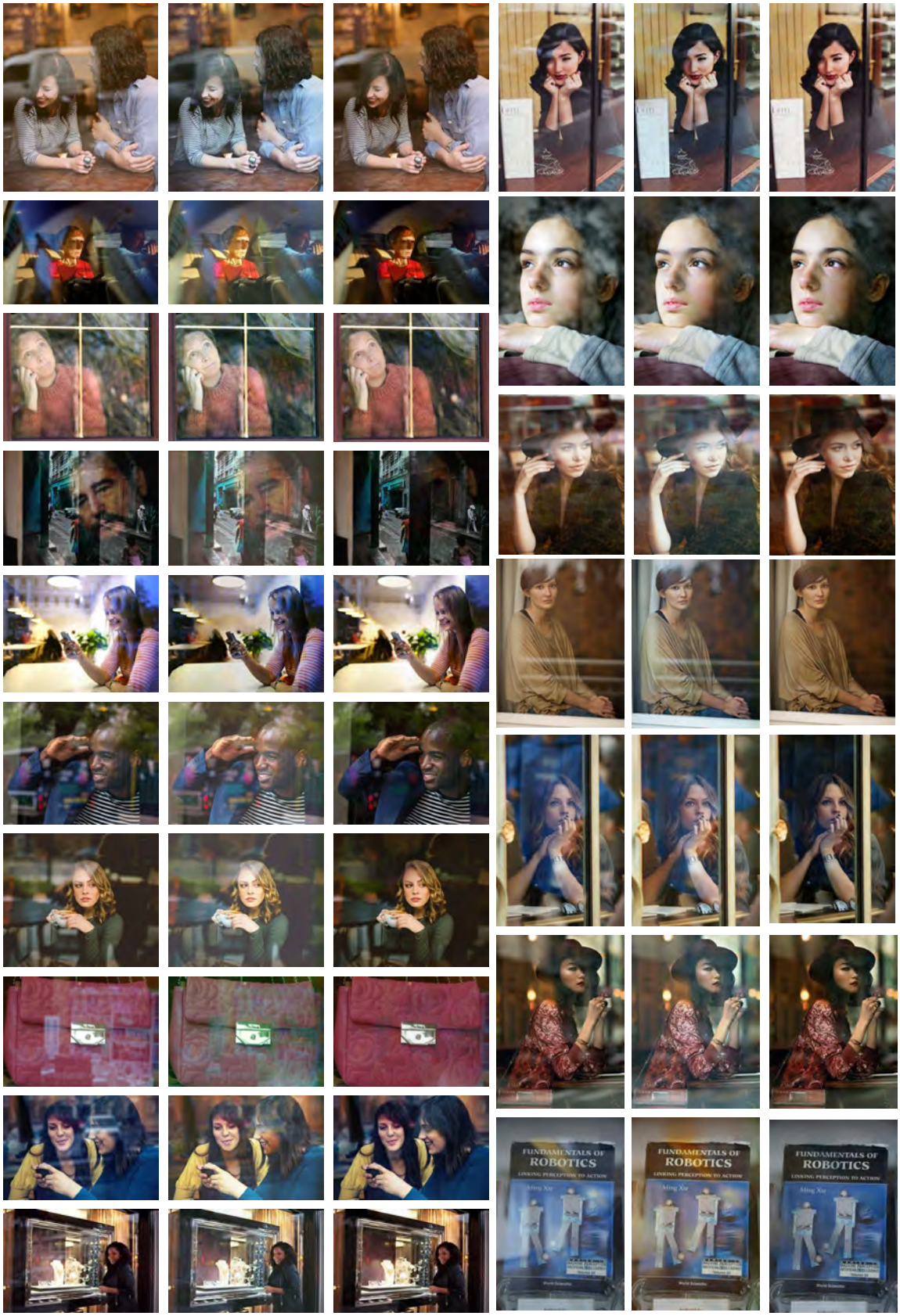
Unaligned - The training set collected by Wei *et al*. [24], where the reflection-contaminated images and transmission layers are mildly aligned. Note that there are 250 pairs captured by DSLR camera and 200 by mobile phone, we use the 250 DSLR pairs following the official code of ERRNet [24].

SIR[2] - A benchmark dataset[3] collected by Wan *et al*. [37], which is composed of three subsets, *i.e*., *Wild* (55), *Postcard* (199), and *Solid* (200), where the *Wild* subset is captured in the wild and the other two subsets are collected in a controlled laboratory environment.

The training/testing dataset configuration of the backbone methods is summarized in Tab. II.

Backbones and Implementation Details. In this paper, we use three state-of-the-art methods as backbone models, *i.e*., ERRNet[4] [24], IBCLN[5] [32], and RAGNet[6] [27]. As introduced in Sec. I-A, they are the best performed SIRR methods with source code and pre-trained model available, and most of recent structure design categories are covered by these methods. All the three methods are implemented in the PyTorch [60] framework. The experiments are conducted on a PC with an Nvidia RTX 3090 GPU.

[2]https://www.flickr.com/
[3]https://rose1.ntu.edu.sg/dataset/sir2Benchmark
[4]https://github.com/Vandermode/ERRNet
[5]https://github.com/JHL-HUST/IBCLN
[6]https://github.com/liyucs/RAGNet



Input RAGNet RAGNet$_{\mathtt{NI}}$ *w/o* $\bar{w}_i$ Input RAGNet RAGNet$_{\mathtt{NI}}$ *w/o* $\bar{w}_i$

Figure 6: More qualitative results on *Real45* dataset.

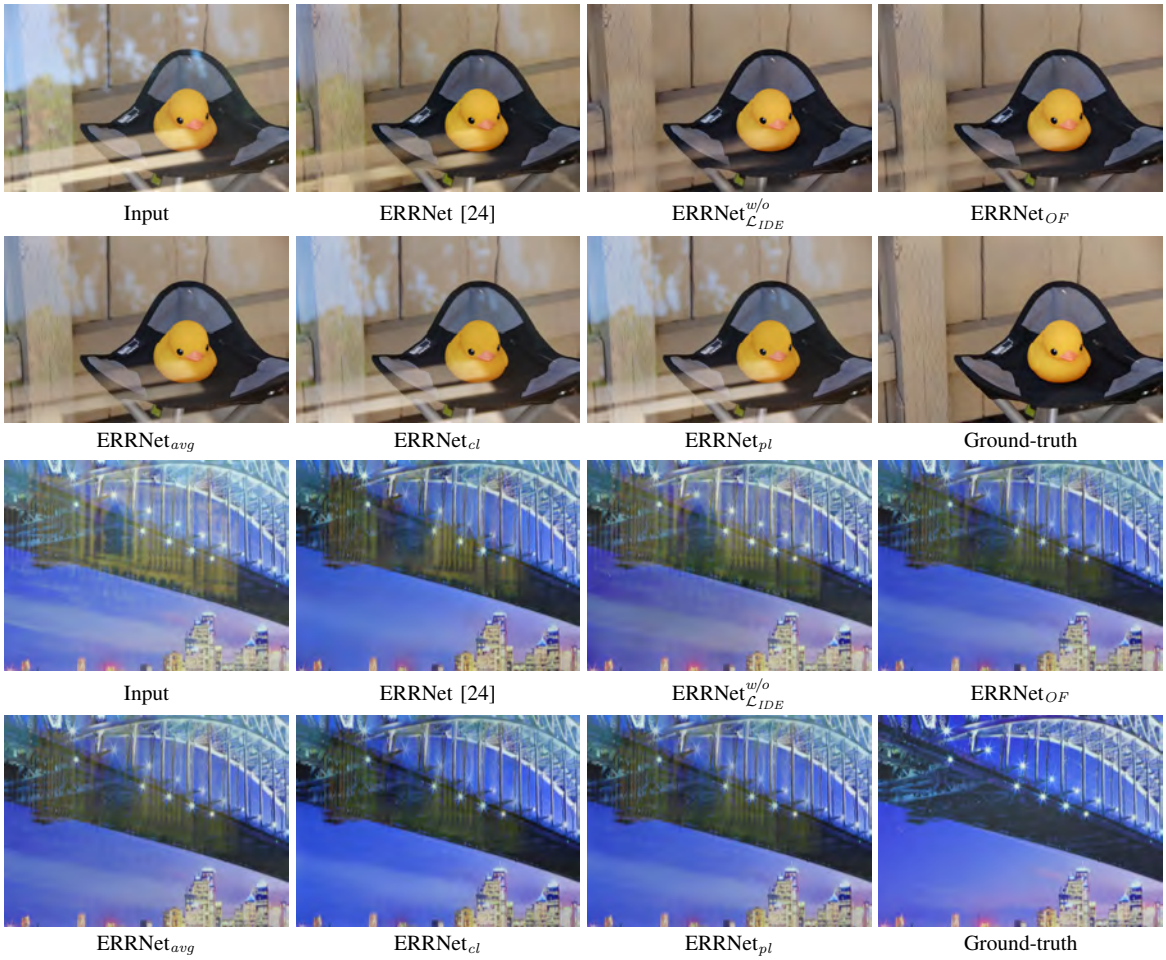


Figure 7: Visual results of ablation studies on IDE loss and RTAW module.

Testing Protocols. As shown in Tab. II, the backbone methods are trained and evaluated on various datasets. For a fair comparison, apart from the proposed IDE Loss $\mathcal{L}_{IDE}$, we train our method with only the training sets and loss functions it used given a backbone model, and evaluate the model performance following its testing protocol. Since only the model trained with aligned image pairs is given, we fine-tuned the ERRNet model following official instructions, and achieved an average PSNR index of 23.79 dB, which is higher than the performance (23.59 dB) reported by Wei *et al*. [24]. We also note that since the image size of *Real20* is too large for inference, following the backbone methods, the shorter side of *Real20* images is resized to 512, 400, and 300 pixels for evaluating ERRNet [24], IBCLN [32], and RAGNet [27], respectively.

B. Comparison with Backbone Methods

As shown in Tab. III, the peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) indices of two AdaNEC methods are provided for comparing against each backbone method. We can see that, the `OF` manner generally can achieve an average PSNR gain of about 0.45∼0.7 dB, and the performance on all datasets is boosted, showing the effectiveness of our AdaNEC$_{\texttt{OF}}$ method. On the other hand, considering the computation amount and inference efficiency, the `NI` manner also achieves considerable performance gain on most datasets. It is worth noting that, IBCLN [32] is a recurrent-based method, which is not friendly to the proposed AdaNEC$_{\texttt{NI}}$ method, and only minor performance gain (0.07 dB) is achieved. While on all other backbones, both AdaNEC$_{\texttt{OF}}$ and AdaNEC$_{\texttt{NI}}$ achieve considerable performance improvements.

As for SSIM, both AdaNEC$_{\texttt{OF}}$ and AdaNEC$_{\texttt{NI}}$ achieve higher SSIM index than the backbone methods in all conditions, showing the promoted reflection removal ability. For ERRNet [24] and RAGNet [24], we also evaluate them on the *Nature20* dataset, which is not considered by these methods, as shown by the gray shaded regions in Tab. III.

As shown in Figs. 3 to 5, we also evaluate the proposed method qualitatively. It can be seen that, when incorporated with our AdaNEC method from a domain generalization perspective, the backbone methods can be enhanced in terms of their reflection removal ability and output image quality. We have also tested on the whole *Real45* dataset, and show the results in Fig. 6.

Table III: Quantitative comparison against the state-of-the-art backbones. For calculating the PSNR and SSIM indices, we follow the testing protocol of the backbone methods. Note that the results of ERRNet [24] and RAGNet [27] on *Nature20* are given for reference (highlighted with gray shadow), and the average indices are calculated on the other four datasets (*Real20*, *Wild*, *Postcard*, and *Solid*) according to the official setting. The better results between AdaNEC methods with or without $\bar{w}_i$ are marked with **bold**, and the best results are marked with red. The image amount of each dataset is provided in the parentheses.

Method	*Real20* (20)		*Wild* (55)		*Postcard* (199)		*Solid* (200)		*Nature20* (20)		Average (474/494)	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
ERRNet [24]	22.07	0.781	25.13	0.889	22.76	0.864	24.62	0.898	20.86	0.757	23.79	0.877
ERRNet$_{\texttt{OF}}$ w/o $\bar{w}_i$	22.70	**0.791**	24.60	0.865	**23.15**	0.871	24.91	0.895	20.46	0.756	24.04	0.877
ERRNet$_{\texttt{OF}}$	**22.80**	0.790	**25.26**	**0.890**	23.08	**0.874**	**25.26**	**0.904**	**20.99**	**0.768**	**24.24**	**0.885**
ERRNet$_{\texttt{NI}}$ w/o $\bar{w}_i$	22.37	0.787	24.63	0.865	23.14	0.874	24.75	0.894	20.53	0.757	23.96	0.878
ERRNet$_{\texttt{NI}}$	**22.81**	**0.791**	**25.69**	**0.895**	**23.56**	**0.884**	**25.13**	**0.902**	**21.20**	**0.771**	**24.44**	**0.889**
IBCLN [32]	21.86	0.762	24.71	0.886	23.39	0.875	24.87	0.893	23.57	0.783	24.08	0.875
IBCLN$_{\texttt{OF}}$ w/o $\bar{w}_i$	**22.53**	**0.794**	25.44	0.888	24.19	0.888	25.05	0.899	24.59	0.818	24.63	0.886
IBCLN$_{\texttt{OF}}$	22.52	0.789	**25.77**	**0.897**	**24.27**	**0.889**	**25.24**	**0.900**	**24.74**	**0.820**	**24.78**	**0.888**
IBLCN$_{\texttt{NI}}$ w/o $\bar{w}_i$	**22.41**	**0.793**	25.20	0.887	**23.52**	**0.887**	24.77	**0.897**	24.56	0.816	**24.21**	**0.885**
IBCLN$_{\texttt{NI}}$	22.04	0.782	**25.35**	**0.894**	23.34	**0.887**	**24.85**	**0.897**	**24.59**	**0.818**	24.17	**0.885**
RAGNet [27]	22.95	0.793	25.52	0.880	23.67	0.879	26.15	0.903	21.21	0.765	24.90	0.886
RAGNet$_{\texttt{OF}}$ w/o $\bar{w}_i$	23.32	0.806	**25.87**	0.895	25.08	0.900	**26.19**	0.907	21.40	0.775	25.57	0.898
RAGNet$_{\texttt{OF}}$	**23.34**	**0.807**	25.85	**0.896**	**25.20**	**0.903**	26.17	**0.908**	**21.48**	0.776	**25.60**	**0.900**
RAGNet$_{\texttt{NI}}$ w/o $\bar{w}_i$	**23.43**	**0.810**	25.52	0.887	23.91	0.884	**26.26**	**0.906**	21.31	0.770	25.07	0.891
RAGNet$_{\texttt{NI}}$	23.18	0.802	**26.25**	**0.899**	**24.90**	**0.906**	25.66	0.903	21.44	**0.777**	**25.31**	**0.900**

Table IV: Ablation studies on domain experts and IDE loss. The upper half shows the PSNR indices of domain experts trained with a single dataset, *i.e.*, *Syn*$_{CEIL}$, *Real89*, and *Unaligned*, where the results better than the backbone ERRNet [24] model are marked with **bold**. The lower half shows the performance of the model trained only with reflection removal losses (*i.e.*, without $\mathcal{L}_{IDE}$).

Methods	*Real20*	*Wild*	*Postcard*	*Solid*	Average
	PSNR↑	PSNR↑	PSNR↑	PSNR↑	PSNR↑
ERRNet [24]	22.08	25.13	22.76	24.62	23.79
Syn$_{CEIL}$	20.22	**25.31**	22.45	24.39	23.51
Real89	**22.84**	24.58	20.24	24.52	22.66
Unaligned	20.07	24.41	**23.25**	24.56	23.80
ERRNet$_{\texttt{OF}}$ (w/o $\mathcal{L}_{IDE}$)	22.80	24.52	20.06	24.65	22.63
ERRNet$_{\texttt{OF}}$	22.80	25.26	23.08	25.26	24.24

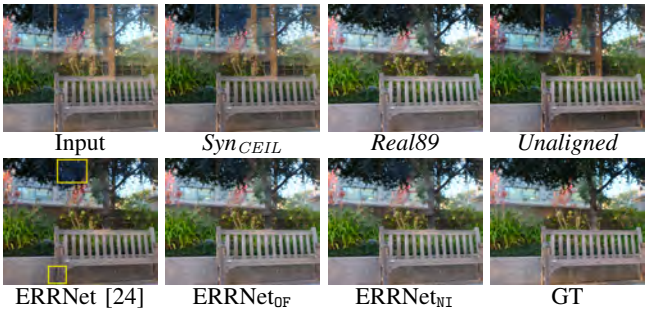


Figure 8: Visual comparison of ERRNet-based domain experts on *Real20* dataset. Please zoom in for better observation.

VII. Ablation Study and Analysis

A. Ablation Study

In this part, we conduct some ablation studies to show the contribution and influence of different parts of the proposed AdaNEC method. Considering the training efficiency and relatively simple structure, the ablation studies are conducted on ERRNet [24].

Performance of domain experts. As shown in Tab. IV, when trained with a single dataset, the performance of the domain experts may be enhanced on some datasets (*e.g.*, generally the performance on *Real20* will be better when trained with *Real89*, since they are the training/testing set from the same dataset), and the final results may approximate these enhanced performance. For *Solid*, all these domain experts perform slightly worse than the backbone model. However, a decent performance gain is obtained when incorporating with AdaNEC, which shows the generalization ability of the proposed method. We also show the visual results in Fig. 8, and the AdaNEC-boosted models can generate better reflection removal results.

In-domain Expert (IDE) Loss. As illustrated in Sec. II-D, the IDE loss is introduced from a straightforward idea. On the one hand, it explicitly requires that the reflection removal should rely more on the in-domain expert, which accords with the idea of determining the weights according to the expertise level in domain generalization. On the other hand, if the RTAW module is trained only with the reflection removal losses, it will keep pursuing extreme weights for some domain experts, resulting in large and unstable values in the RTAW outputs. When the IDE loss is applied, the weight of other domain experts will be constrained to a reasonable range, and we empirically find that the training of RTAW module will become more stable. As shown in Tab. IV, when the IDE loss is discarded, the training of RTAW is usually unstable and finally results in different order of magnitude of the expertise levels (v_i). Therefore, the final result heavily relies on a specific domain expert. We can see that the performance of ERRNet$^{w/o}_{\mathcal{L}_{IDE}}$ is very close to the domain expert trained on *Real89* dataset, indicating the biased training.

Furthermore, we also apply the IDE loss on high-level domain generalization tasks for further verification (*i.e.*, image classification). The experiments are conducted with the

Table V: Applying our method on image classification-based domain generalization tasks.

Method	PACS					Office-Home				
	Art	Cat	Pho	Skt	Avg	Art	Clp	Prd	Rel	Avg
Vanilla [45]	77.0	75.9	96.0	69.2	79.5	58.9	49.4	74.3	76.2	64.7
MMD-AAE [61]	75.2	72.7	96.0	64.2	77.0	56.5	47.3	72.1	74.8	62.7
CCSA [62]	80.5	76.9	93.6	66.8	79.4	59.9	49.9	74.1	75.7	64.9
JiGen [63]	79.4	75.3	96.0	71.6	80.5	53.0	47.5	71.5	72.8	61.2
CrossGrad [64]	79.8	76.8	96.0	70.2	80.7	58.4	49.4	73.9	75.8	64.4
Epi-FCR [65]	82.1	77.0	93.9	73.0	81.5	-	-	-	-	-
DMG [66]	76.9	**80.4**	93.4	75.2	81.5	-	-	-	-	-
DAEL [45]	84.6	74.4	95.6	78.9	83.4	59.4	55.1	74.0	75.7	66.1
Ours (w/o $\mathcal{L}_{IDE}$)	**84.8**	76.0	**96.4**	79.5	84.2	59.6	55.6	75.6	75.7	66.6
Ours	84.6	78.2	95.9	**79.9**	**84.7**	**60.4**	**55.7**	**75.6**	**76.2**	**67.0**

Table VI: Ablation studies on RTAW. The indices better than the backbone model are highlighted with green and those worse than the backbone model are highlighted with red (the difference of PSNR $<$ 0.05dB and SSIM $<$ 0.004 has been ignored).

Methods	*Real20*		*Wild*		*Postcard*		*Solid*		Average	
	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
ERRNet [24]	22.08	0.781	25.13	0.889	22.76	0.864	24.62	0.898	23.79	0.877
ERRNet$_{average}$	21.50	0.760	25.63	0.899	22.95	0.880	25.26	0.904	24.17	0.887
ERRNet$_{single}$	21.79	0.772	25.14	0.888	23.17	0.870	25.20	0.899	24.20	0.880
ERRNet$_{parallel}$	22.11	0.782	24.91	0.882	23.50	0.873	24.89	0.894	24.19	0.879
ERRNet$_{\texttt{OF}}$	22.80	0.790	25.26	0.890	23.08	0.874	25.26	0.904	24.24	0.885

DAEL [45] backbone. First we add the RTAW module[7] to the DAEL framework for predicting the weights (rather than the original simple averaging in DAEL). The IDE loss is then applied for comparison. As shown in Tab. V, the proposed IDE loss also works well on high-level domain generalization tasks. We follow the DAEL training settings in these experiments.

RTAW Module. We also conduct ablation studies on the structure of RTAW module, and three variants are compared as follows. 1) ERRNet$_{average}$: A plain average operation, the simplest way to perform model combination. 2) ERRNet$_{parallel}$: In this case, the feature extractor $\mathcal{F}$ and the CDAM module are removed from RTAW (see Fig. 1), resulting in N feature extractors deployed in parallel, where each model predicts the expertise level for a specific domain expert. 3) ERRNet$_{single}$: We further simplify the RTAW module by using only one feature extractor, which is composed of five convolution layers and a fully-connected layer.

As shown in Tab. VI, even it performs well on some datasets, the generalization ability is not guaranteed when a plain average operation is taken, resulting in severe performance degradation of ERRNet$_{avg}$ on the *Real20* dataset. ERRNet$_{single}$ and ERRNet$_{parallel}$ also face the problem of performance degradation on some datasets. On the contrary, the proposed method with cross-domain attention module can generalize better to all target domains, showing the superiority of our RTAW.

Domain-level AdaNEC. In Sec. II-E, we propose to replace the per-sample weight w_i with the average weight $\bar{w}_i$ of the domain (testing set), and perform a domain-level AdaNEC. To show the effectiveness of the domain-level AdaNEC method, the results using w_i are also provided in Tab. III. The better results between AdaNEC methods with or without $\bar{w}_i$ are marked with bold. As one can see, it can bring some extra performance gain in most cases, verifying the effectiveness of domain-level AdaNEC methods.

[7]Here we reuse the name RTAW for convenience. Since the DAEL training is based on pre-trained ResNet [67] models, we also use a pre-trained ResNet model as the RTAW module for faster convergence.

B. Visual Results of Ablation Studies

In this section, we show visual results of ablation studies on IDE loss and RTAW module in Fig. 7. The first two rows indicate an example from *Real20* dataset, and the ERRNet$^{w/o}_{\mathcal{L}_{IDE}}$ shows a similar performance with our ERRNet$_{OF}$, while others are obviously worse than these two methods. The last two rows show a hard example from *Postcard* dataset, and the performance of ERRNet$^{w/o}_{\mathcal{L}_{IDE}}$ degrades seriously, while other methods can generate better results than it. The visual results further verify the conclusions drawn in Sec. 5.1, showing the superior generalization ability of the proposed method.

References

[1] Szeliski R, Avidan S, Anandan P. Layer extraction from multiple images containing reflections and transparency. In: IEEE Conference on Computer Vision and Pattern Recognition. 2000, 246–253

[2] Sarel B, Irani M. Separating transparent layers through layer information exchange. In: European Conference on Computer Vision. 2004, 328–341

[3] Gai K, Shi Z, Zhang C. Blind separation of superimposed moving images using image statistics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(1): 19–32

[4] Li Y, Brown M S. Exploiting reflection change for automatic reflection removal. In: IEEE International Conference on Computer Vision. 2013, 2432–2439

[5] Guo X, Cao X, Ma Y. Robust separation of reflection from multiple images. In: IEEE Conference on Computer Vision and Pattern Recognition. 2014, 2187–2194

[6] Xue T, Rubinstein M, Liu C, Freeman W T. A computational approach for obstruction-free photography. ACM Transactions on Graphics, 2015, 34(4): 1–11

[7] Yang J, Li H, Dai Y, Tan R T. Robust optical flow estimation of double-layer images under transparency or reflection. In: IEEE Conference on Computer Vision and Pattern Recognition. 2016, 1410–1419

[8] Sun C, Liu S, Yang T, Zeng B, Wang Z, Liu G. Automatic reflection removal using gradient intensity and motion cues. In: ACM International Conference on Multimedia. 2016, 466–470

[9] Punnappurath A, Brown M S. Reflection removal using a dual-pixel sensor. In: IEEE Conference on Computer Vision and Pattern Recognition. 2019, 1556–1565

[10] Liu Y L, Lai W S, Yang M H, Chuang Y Y, Huang J B. Learning to see through obstructions. In: IEEE Conference on Computer Vision and Pattern Recognition. 2020, 14215–14224

[11] Agrawal A, Raskar R, Nayar S K, Li Y. Removing photography artifacts using gradient projection and flash-exposure sampling. In: ACM SIGGRAPH. 2005, 828–835

[12] Lei C, Chen Q. Robust reflection removal with reflection-free flash-only cues. In: IEEE Conference on Computer Vision and Pattern Recognition. 2021, 14811–14820

[13] Sarel B, Irani M. Separating transparent layers through layer information exchange. In: European Conference on Computer Vision. 2004, 328–341

[14] Wieschollek P, Gallo O, Gu J, Kautz J. Separating reflection and transmission images in the wild. In: European Conference on Computer Vision. 2018, 89–104

[15] Lei C, Huang X, Zhang M, Yan Q, Sun W, Chen Q. Polarized reflection removal with perfect alignment in the wild. In: IEEE Conference on Computer Vision and Pattern Recognition. 2020, 1750–1758

[16] Li Y, Brown M S. Single image layer separation using relative smoothness. In: IEEE Conference on Computer Vision and Pattern Recognition. 2014, 2752–2759

[17] Wan R, Shi B, Tan A H, Kot A C. Depth of field guided reflection removal. In: IEEE International Conference on Image Processing. 2016, 21–25

[18] Levin A, Weiss Y. User assisted separation of reflections from a single image using a sparsity prior. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2007, 29(9): 1647–1654

[19] Arvanitopoulos N, Achanta R, Susstrunk S. Single image reflection suppression. In: IEEE Conference on Computer Vision and Pattern Recognition. 2017, 4498–4506

[20] Wan R, Shi B, Duan L Y, Tan A H, Gao W, Kot A C. Region-aware reflection removal with unified content and gradient priors. IEEE Transactions on Image Processing, 2018, 27(6): 2927–2941

[21] Shih Y, Krishnan D, Durand F, Freeman W T. Reflection removal using ghosting cues. In: IEEE Conference on Computer Vision and Pattern Recognition. 2015, 3193–3201

[22] Fan Q, Yang J, Hua G, Chen B, Wipf D. A generic deep architecture for single image reflection removal and image smoothing. In: IEEE International Conference on Computer Vision. 2017, 3238–3247

[23] Zhang X, Ng R, Chen Q. Single image reflection separation with perceptual losses. In: IEEE Conference on Computer Vision and Pattern Recognition. 2018, 4786–4794

[24] Wei K, Yang J, Fu Y, Wipf D, Huang H. Single image reflection removal exploiting misaligned training data and network enhancements. In: IEEE Conference on Computer Vision and Pattern Recognition. 2019, 8178–8187

[25] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014

[26] Dong Z, Xu K, Yang Y, Bao H, Xu W, Lau R W. Location-aware single image reflection removal. In: IEEE International Conference on Computer Vision. 2021, 5017–5026

[27] Li Y, Liu M, Yi Y, Li Q, Ren D, Zuo W. Two-stage single image reflection removal with reflection-aware guidance. arXiv preprint arXiv:2012.00945, 2020

[28] Chang Y, Jung C. Single image reflection removal using convolutional neural networks. IEEE Transactions on Image Processing, 2018, 28(4): 1954–1966

[29] Feng X, Pei W, Jia Z, Chen F, Zhang D, Lu G. Deep-masking generative network: A unified framework for background restoration from superimposed images. IEEE Transactions on Image Processing, 2021, 30: 4867–4882

[30] Chen W T, Chen K Y, Chen I H, Fang H Y, Ding J J, Kuo S Y. Missing recovery: Single image reflection removal based on auxiliary prior learning. IEEE Transactions on Image Processing, 2022, 32: 643–656

[31] Yang J, Gong D, Liu L, Shi Q. Seeing deeply and bidirectionally: A deep learning approach for single image reflection removal. In: European Conference on Computer Vision. 2018, 654–669

[32] Li C, Yang Y, He K, Lin S, Hopcroft J E. Single image reflection removal through cascaded refinement. In: IEEE Conference on Computer Vision and Pattern Recognition. 2020, 3565–3574

[33] Prasad B, Boregowda L R, Mitra K, Chowdhury S, others . V-DESIRR: Very fast deep embedded single image reflection removal. In: IEEE International Conference on Computer Vision. 2021, 2390–2399

[34] Wen Q, Tan Y, Qin J, Liu W, Han G, He S. Single image reflection removal beyond linearity. In: IEEE Conference on Computer Vision and Pattern Recognition. 2019, 3771–3779

[35] Kim S, Huo Y, Yoon S E. Single image reflection removal with physically-based training images. In: IEEE Conference on Computer Vision and Pattern Recognition. 2020, 5164–5173

[36] Zheng Q, Shi B, Chen J, Jiang X, Duan L Y, Kot A C. Single image reflection removal with absorption effect. In: IEEE Conference on Computer Vision and Pattern Recognition. 2021, 13395–13404

[37] Wan R, Shi B, Duan L Y, Tan A H, Kot A C. Benchmarking single-image reflection removal algorithms. In: IEEE International Conference on Computer Vision. 2017, 3922–3930

[38] Wan R, Shi B, Li H, Duan L Y, Tan A H, Kot A C. CoRRN: Cooperative reflection removal network. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019, 42(12): 2969–2982

[39] Lei C, Huang X, Qi C, Zhao Y, Sun W, Yan Q, Chen Q. A categorized reflection removal dataset with diverse real-world scenes. arXiv preprint arXiv:2108.03380, 2021

[40] Blanchard G, Lee G, Scott C. Generalizing from several related classification tasks to a new unlabeled sample. Advances in Neural Information Processing Systems, 2011, 24: 2178–2186

[41] Ding Z, Fu Y. Deep domain generalization with structured low-rank constraint. IEEE Transactions on Image Processing, 2017, 27(1): 304–313

[42] Mancini M, Bulo S R, Caputo B, Ricci E. Best sources forward: domain generalization through source-specific nets. In: IEEE International Conference on Image Processing. 2018, 1353–1357

[43] D'Innocente A, Caputo B. Domain generalization with domain-specific aggregation modules. In: German Conference on Pattern Recognition. 2018, 187–198

[44] Wang S, Yu L, Li K, Yang X, Fu C W, Heng P A. Dofe: Domain-oriented feature embedding for generalizable fundus image segmentation on unseen datasets. IEEE Transactions on Medical Imaging, 2020, 39(12): 4237–4248

[45] Zhou K, Yang Y, Qiao Y, Xiang T. Domain adaptive ensemble learning. IEEE Transactions on Image Processing, 2021, 30: 8008–8018

[46] Wang J, Lan C, Liu C, Ouyang Y, Zeng W, Qin T. Generalizing to unseen domains: A survey on domain generalization. arXiv preprint arXiv:2103.03097, 2021

[47] Zhou K, Liu Z, Qiao Y, Xiang T, Loy C C. Domain generalization: A survey. arXiv preprint arXiv:2103.02503, 2021

[48] Chen H, Lundberg S, Lee S I. Checkpoint ensembles: Ensemble methods from a single training process. arXiv preprint arXiv:1710.03282, 2017

[49] Shoshan A, Mechrez R, Zelnik-Manor L. Dynamic-net: Tuning the objective without re-training for synthesis tasks. In: IEEE International Conference on Computer Vision. 2019, 3215–3223

[50] Wang X, Yu K, Dong C, Tang X, Loy C C. Deep network interpolation for continuous imagery effect transition. In: IEEE Conference on Computer Vision and Pattern Recognition. 2019, 1692–1701

[51] Parkin R E. A note on the extinction coefficient and absorptivity of glass. Solar Energy, 2015, 114: 196–197

[52] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł, Polosukhin I. Attention is all you need. In: Advances in Neural Information Processing Systems. 2017, 5998–6008

[53] Li D, Yang Y, Song Y Z, Hospedales T M. Deeper, broader and artier domain generalization. In: IEEE International Conference on Computer Vision. 2017, 5542–5550

[54] Johnson J, Alahi A, Fei-Fei L. Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision. 2016, 694–711

[55] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial nets. In: Advances in Neural Information Processing Systems. 2014, 2672–2680

[56] Good I J. Rational decisions. Journal of the Royal Statistical Society. Series B (Methodological), 1952, 14(1): 107–114

[57] Mannor S, Peleg D, Rubinstein R. The cross entropy method for classification. In: International Conference on Machine Learning. 2005, 561–568

[58] Jolicoeur-Martineau A. The relativistic discriminator: a key element missing from standard gan. arXiv preprint arXiv:1807.00734, 2018

[59] Everingham M, Van Gool L, Williams C K, Winn J, Zisserman A. The pascal visual object classes (voc) challenge. International Journal of Computer Vision, 2010, 88(2): 303–338
[60] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, others . Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems. 2019, 8026–8037
[61] Li H, Pan S J, Wang S, Kot A C. Domain generalization with adversarial feature learning. In: IEEE Conference on Computer Vision and Pattern Recognition. 2018, 5400–5409
[62] Motiian S, Piccirilli M, Adjeroh D A, Doretto G. Unified deep supervised domain adaptation and generalization. In: IEEE International Conference on Computer Vision. 2017, 5715–5725
[63] Carlucci F M, D'Innocente A, Bucci S, Caputo B, Tommasi T. Domain generalization by solving jigsaw puzzles. In: IEEE Conference on Computer Vision and Pattern Recognition. 2019, 2229–2238
[64] Shankar S, Piratla V, Chakrabarti S, Chaudhuri S, Jyothi P, Sarawagi S. Generalizing across domains via cross-gradient training. arXiv preprint arXiv:1804.10745, 2018
[65] Li D, Zhang J, Yang Y, Liu C, Song Y Z, Hospedales T M. Episodic training for domain generalization. In: IEEE International Conference on Computer Vision. 2019, 1446–1455
[66] Chattopadhyay P, Balaji Y, Hoffman J. Learning to balance specificity and invariance for in and out of domain generalization. In: European Conference on Computer Vision. 2020, 301–318
[67] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. 2016, 770–778