Domain Adaptation for Image-to-Image Translation Diffusion Models

Supplementary Material

1. Details of our methods

Algorithm 1 Source-Available Domain Adaptation For Diffusion Model with Domain Noise Alignment

Require: Source domain D_t, diffusion model $\epsilon_\theta(x_t, t, c)$, Timestep t_i
Ensure: Output list n_{all}

for each $c_t \in D_t$ **do**
  sample $x_T \in N(0,1)$, $n_t = ()$
  for each $t = T$ to 0 step t_i **do**
    $\epsilon_t = \epsilon_\theta(x_t, c, c_t)$, $n_t.append(\|\epsilon_t\|_2)$
    $x_{t-1} = scheduler.step(\epsilon_t, t, x_t).prev\_sample$
  end for
  $n_{all}.append(n_t)$
end for
$n_{all} = n_{all}.mean(dim = 0)$
return n_{all}

Require: Target image c_s, diffusion model $\epsilon_\theta(x_t, t, c)$, Source domain noise list n_{all}, Timestep t_i
Ensure: Output result x_0

Initialize $\delta_0 = 0$, $\delta_1 = 0$, $\lambda = 1$, $x_T \in N(0,1)$
for each $t = T$ to 1 step t_i **do**
  $\epsilon_t = \epsilon_\theta(x_t, c, c_s)$, $\delta_1 = \|\epsilon_t\|_2 / n_{all}[(T-t)/t_i]$
  if $\delta_0 \neq 0$ **then**
    $\lambda = \delta_1 - \delta_0 + 1 - sum_\lambda$
  end if
  $sum_\lambda = sum_\lambda + \lambda - 1$, $\delta_0 = \delta_1$
  $\hat{x_0} = (x_t - \sqrt{\bar{\beta}_t}\epsilon_t/\lambda)/\sqrt{\bar{\alpha}_t}$
  $x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x_0} + (1 - \sqrt{\bar{\alpha}_{t-1}})\epsilon_t$
end for
return x_0

1.1. For Source-Available DA Setting

We show our Domain Noise Alignment for Source-Available DA in Algorithm 1. Specifically, we pre-calculate the variance of noise predictions from the source domain. Then, we align the noise predictions from the target domain with those pre-calculated. Instead of directly scaling the noise prediction, we utilize the method for solving λ_t for each timestep: $\Delta N(t) - 1 \approx \int_t^T (\lambda_T - 1)dt + \int_t^{T-1}(\lambda_{T-1} - 1)dt + ... + \int_t^{t+1}(\lambda_{t+1} - 1)dt$, where $\Delta N(t) = \|\boldsymbol{\epsilon_\theta}(\boldsymbol{x}_t, t, c_s)\|_2 / \|\boldsymbol{\epsilon_\theta}(\boldsymbol{x}_t, t, c_t)\|_2$. We can approximate λ_t by leveraging the error between two adjacent timesteps, as described in Sec. 4.2 in the main text.

Algorithm 2 Source-Free Domain Adaptation For Diffusion Model with Domain Noise Alignment

Require: Target image c_s, diffusion model $\epsilon_\theta(x_t, t, c)$, Timestep t_i, Linear mask schedule P, Sample times k
Ensure: Output result x_0

Initialize $\delta_0 = 0$, $\delta_1 = 0$, $\lambda = 1$
sample $x_{T_1}, \ldots, x_{T_k} \in N(0,1)$,
for each $t = T$ to 1 step t_i **do**
  for each $i = 1$ to k **do**
    $\epsilon_{t_i} = \epsilon_\theta(x_{t_i}, c, c_s)$
    $\hat{x_{0_i}} = (x_t - \sqrt{\bar{\beta}_t}\epsilon_{t_i}/\lambda)/\sqrt{\bar{\alpha}_t}$
  end for
  $\gamma = cal\_consistency(\epsilon_{t_1}, \ldots, \epsilon_{t_k})$
  $p_t = P[T/t_i]$
  $M = quantile(stack[\hat{x_{0_1}}, \ldots, \hat{x_{0_k}}].var(0), p_t)$
  $\delta_1 = \|\epsilon_t\|_2 / \|\epsilon_t[M]\|_2$
  if $\delta_0 \neq 0$ **then**
    $\lambda = (\delta_1 - \delta_0 + 1 - sum_\lambda)\gamma$
  end if
  $sum_\lambda = sum_\lambda + \lambda - 1$
  $\delta_0 = \delta_1$
  for each $i = 1$ to k **do**
    $\hat{x_{0_i}} = (x_{t_i} - \sqrt{\bar{\beta}_t}\epsilon_{t_i}/\lambda)/\sqrt{\bar{\alpha}_t}$
    $x_{t-1_i} = \sqrt{\bar{\alpha}_{t-1}}\hat{x_{0_i}} + (1 - \sqrt{\bar{\alpha}_{t-1}})\epsilon_{t_i}$
  end for
end for
return $ensemble(x_{0_1}, \ldots, x_{0_k})$

1.2. For Source-Free DA Setting

We show our Domain Noise Alignment and a noise estimation mechanism for Source-Free DA in Algorithm 2. Specifically, by performing multiple samplings of the initial noise in a batch, we can generate target images $x_0 \in R^{B\times C\times H\times W}$ and define high-confidence and low-confidence regions based on their variances along the batch dimension, where B represents the number of initial noise samples. The regions with lower variance are more reliable in prediction compared to other regions. Therefore, we use p as the percentage threshold to obtain regions with higher confidence and approximate the statistical properties of the source domain during the denoising process by leveraging the statistics of these selected regions

Table 1. Comparison results of diffusion models w/ and w/o our methods for Blind SR under different DA settings. The results are the average performance on four kinds of degradations(bicubic+jpeg, bicubic+noise, bicubic+blur, bicubic+jepg+noise+blur).

Method	Set14				BSD100			
	PSNR↑	LPIPS↑	MUSIQ↑	MANIQA↑	PSNR↑	LPIPS↑	MUSIQ↑	MANIQA↑
Source-Dependent Domain Adaptation								
DiffBIR	21.72	0.447	70.09	0.59	22.01	0.451	72.45	0.62
DiffBIR+ours	**22.27**	**0.456**	**71.42**	**0.59**	**22.69**	**0.459**	**72.91**	**0.64**
StableSR	21.98	0.455	68.23	0.56	22.27	0.455	70.12	0.58
StableSR+ours	**22.43**	**0.469**	**68.74**	**0.57**	**22.84**	**0.461**	**70.43**	**0.59**
Source-Free Domain Adaptation								
DiffBIR	21.72	0.447	70.09	0.59	22.01	0.451	72.45	0.62
DiffBIR+ours	**22.05**	**0.452**	**71.34**	**0.59**	**22.38**	**0.456**	**72.52**	**0.63**
StableSR	21.98	0.455	68.23	0.56	22.27	0.455	70.12	0.58
StableSR+ours	**22.14**	**0.463**	**68.42**	**0.57**	**22.45**	**0.458**	**70.33**	**0.58**

2. More Implementation Details

2.1. Dataset Details

Depth Estimation. We mainly test our Method on nuScenes and RobotCar_Night datasets, both of which contain images under diverse conditions. nuScenes is a large autonomous driving dataset comprising 1000 video clips collected in diverse road scenes and weather conditions. These scenes are pretty challenging, with a fair amount of unexpected regions. RobotCar_Night is a large-scale autonomous driving dataset that includes driving videos captured on a consistent route during various weather conditions, traffic conditions, and times of day and night.

Moreover, we also test our method on more commonly used datasets, like NYUv2 and KITTI. The NYUv2 dataset is a high-quality dataset widely used for depth estimation and indoor scene understanding, released by New York University. It contains 1,449 pairs of RGB-D images, captured from diverse indoor scenes (such as offices, living rooms, kitchens, etc.), with depth information acquired using the Microsoft Kinect sensor. The KITTI depth estimation dataset comprises 93,000 images, which are frames extracted from continuous video sequences, encompassing a variety of urban scenes and driving conditions. These images are paired with their corresponding sparse depth maps (generated by LiDAR), providing a rich resource of training and testing data for depth estimation tasks.

Blind Super-Resolution. We mainly test our Method on RealSR and DRealSR datasets. RealSR and DRealSR are two significant datasets designed for real-world super-resolution (SR) tasks, aiming to address the shortcomings of traditional super-resolution methods in real-world scenarios. For ease of comparison, we only utilize their validation datasets for testing, where RealSR contains 100 paired images and DRealSR contains 93 paired images. The low-resolution (LR) images have a resolution of 128×128, while the high-resolution (HR) images have a resolution of 512×512.

Moreover, we also test our method on synthetic datasets Set14 and BSD100. Following the setting of previous work, we adopt several synthetic degradations to the HR images in these two datasets, and report the average performance. We choose eight kinds of degradations: bicubic, bicubic+noise, bicubic+blur, bicubic+jpeg, bicubic+noise+blur, bicubic+noise+jpeg, bicubic+blur+jpeg, bicubic+noise+blur+jpeg.

Optical Flow. We mainly test our Method on FCDN and Sintel datasets. FCDN (FlyingChairs-DarkNoise) is a novel dataset proposed by Zheng, specifically designed to enhance optical flow estimation in low-light environments. This dataset is built upon the open-source optical flow dataset, FlyingChairs, and is synthesized by incorporating noise characteristics and automatic white balance (AWB) features under low-light conditions. Specifically, the authors first estimated the noise characteristics in low-light settings (primarily based on Poisson and Gaussian distributions) and then integrated these noise characteristics into the FlyingChairs dataset, resulting in the creation of the FCDN dataset. The Sintel dataset comprises 35 scenes with a total of 1,628 frames, where each scene contains between 20 to 50 frames. The frame rate is 24 fps, and the resolution of the images is 1024×436.

Semantic Segmentation. We mainly test our Method on ACDC and Dark Zurich datasets. ACDC (Adverse Conditions Dataset with Correspondences) is a semantic segmentation dataset focused on extreme visual scenarios, primarily designed for intelligent driving applications. The dataset contains 4,006 images, evenly distributed across four common adverse weather conditions: fog, nighttime, rain, and snow. Each image is accompanied by high-quality pixel-level annotations, as well as corresponding images captured under normal conditions in nearly identical scenes. Addi-

tionally, a binary mask is provided to distinguish between clear and uncertain semantic content within the images. Dark Zurich is an image dataset designed for semantic segmentation, focusing on scenes captured during nighttime, dusk, and daytime. The dataset comprises a total of 8,779 images, each accompanied by the camera's GPS coordinates. These coordinates are utilized to establish cross-time correspondences, matching each nighttime or dusk image with its corresponding daytime counterpart.

2.2. Testing Details

Depth Estimation. We adopt Marigold and Lotus as baseline. Both of them are fine-tuned from Stable Diffusion to utilize the generative prior, helping generate more accurate depth maps. Marigold employs the v-prediction paradigm, while Lotus adopts the x_0 prediction paradigm. When collecting noise at different time steps, we utilize the transformation method between diffusion prediction paradigms to obtain the noise prediction:

$$\boldsymbol{\epsilon} = \sqrt{\bar{\alpha}_t}\boldsymbol{v}_{\boldsymbol{\theta}} + \sqrt{\bar{\beta}_t}\boldsymbol{x}_t \tag{1}$$

$$\epsilon = (\boldsymbol{x}_t - \sqrt{\bar{\beta}_t}\boldsymbol{x}_{\boldsymbol{\theta}})/\sqrt{\bar{\alpha}_t} \tag{2}$$

Blind Estimation. We adopt DiffBIR and StableSR as the baseline. For DiffBIR, it adopts a pre-processing Network to obtain a coarse HR image. The pre-processing Network is trained on the first stage with synthetic data. Although the pre-processing Network helps the diffusion backnone generate more realistic images, it may encounter unseen degradations that cannot be effectively removed. So the images after the pre-processing network still has large domain gap. For StableSR, it uses a time-aware encoder to encode LR images and then embed the information with cross-attention. Both of them use post-processing modules to help improve fidelity. Considering that the effect of the post-processing modules is not completely controllable, we do not adopt them. Instead, we use the CFG technique to help balance the fidelity and reality of the output HR images:

$$\boldsymbol{\epsilon}_{\boldsymbol{\theta}} = w\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) + (1-w)\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, \Phi) \tag{3}$$

Where $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}$ represents the denoising network, c represents the condition and Φ represents non-condition generation.

Optical Flow. We adopt Open-DDVM as the baseline. Open-DDVM uses a two-stage refinement method to generate an optical flow map with two input images x_1 and x_2 from coarse to fine. It first utilizes the pretrained diffusion model to generate the coarse optical flow map with low-resolution inputs. Then x_2 is back warped with the coarse output to obtain the estimated x_1 which is later compared with x_0 to select the unrecovered regions. The unrecovered regions are then re-input to the diffusion model to refine the output. However, it is enough to collect the statistics of noise prediction on the first stage and calculate λ_t with these.

Semantic Segmentation. We adopt DDP as the baseline. DDP relies on a category list to predict the semantic map. When we test DDP on the target domain, It is imperative to ensure that the categories in the source domain align consistently with those in the target domain. Therefore, we select the common categories between the source domain and the target domain to create a category table, and then only evaluate the performance on these shared categories.

Table 2. Comparison results of diffusion models w/ and w/o our methods for depth estimation under source-dependent DA setting.

Method	NYUv2		KITTI	
	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑
Marigold	8.19	89.8	12.43	85.19
Marigold+ours	**7.26**	**90.9**	**11.97**	**85.56**
Lotus	8.34	89.2	11.15	86.22
Lotus+ours	**7.61**	**90.4**	**10.64**	**86.81**

3. More Quantitative Results

3.1. Depth Estimation

The performance of our method on two more commonly used datasets, NYUv2 and KITTI, is shown in Table 2. Although their divergence from the source domain is not substantial, our method still achieves significant improvements on these two datasets.

3.2. Blind Super-Resolution

The performance of our method on two synthetic datasets, Set14 and BSD100, are shown in Table 1. Notably, although the pre-processing network is trained on synthetic data, it still struggles when the degradations are more complex. We show the consistency of predictions when diffusion models encounter heavier degradations in Figure, illustrating the domain gap still exists. Therefore, our method is also effective on synthetic datasets.

4. More Visualizations

We present visual comparison results across multiple tasks and datasets, demonstrating that our method can effectively help diffusion-based dense prediction models perform more robustly under extreme cross-domain conditions.

References

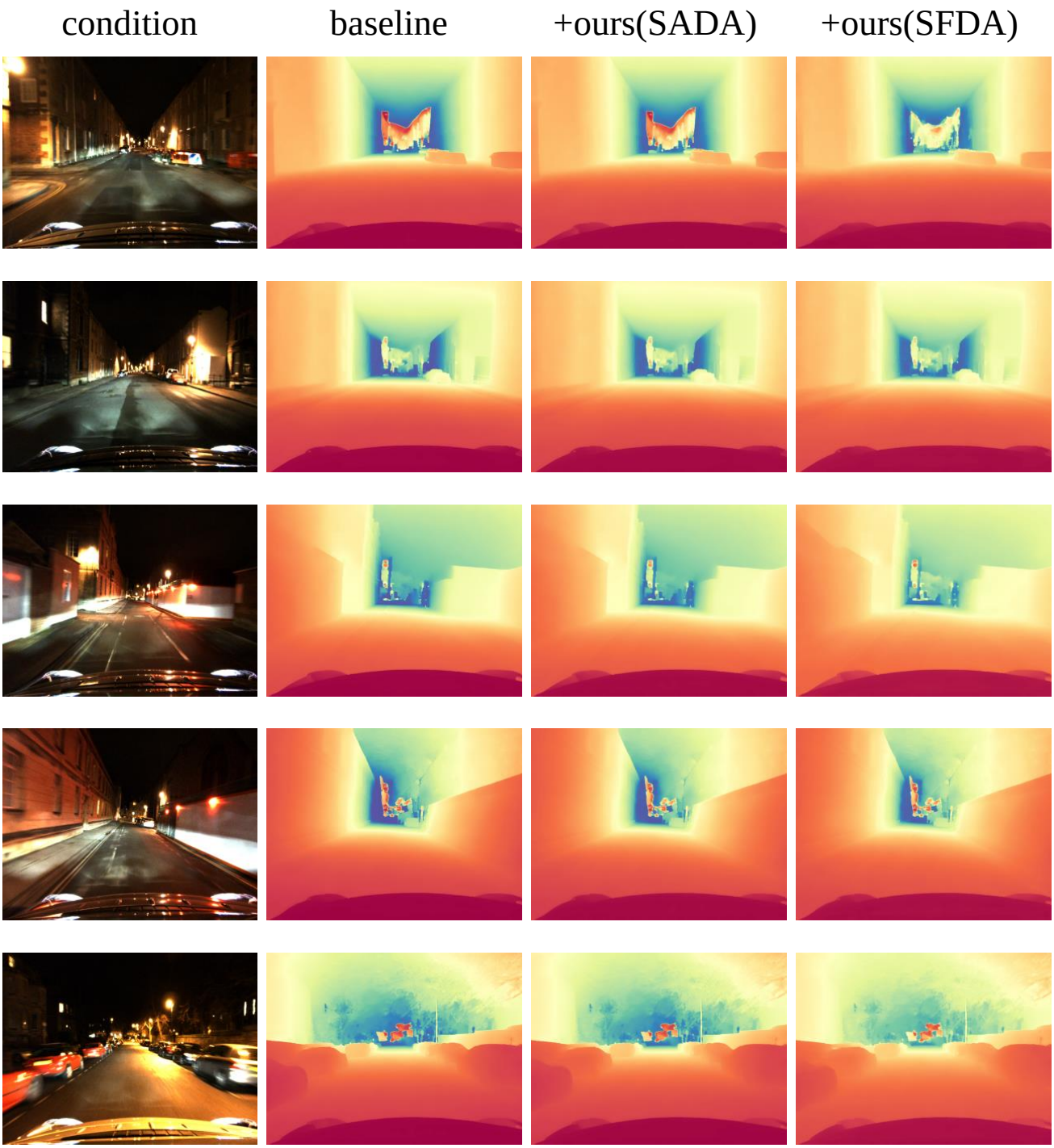

Figure 1. Competitive results of models w/ and w/o our methods in depth estimation.

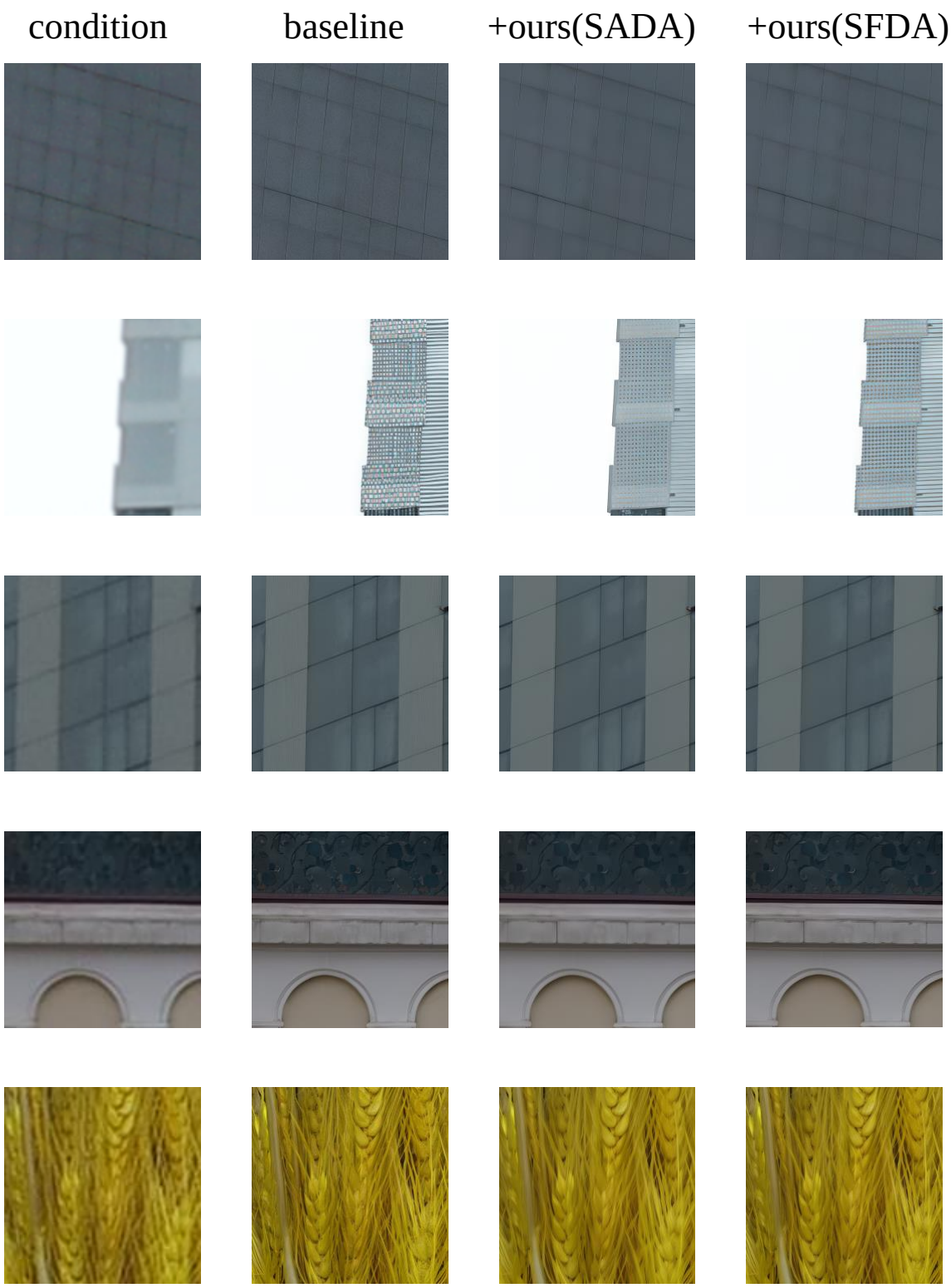

Figure 2. Competitive results of models w/ and w/o our methods in Blind SR.

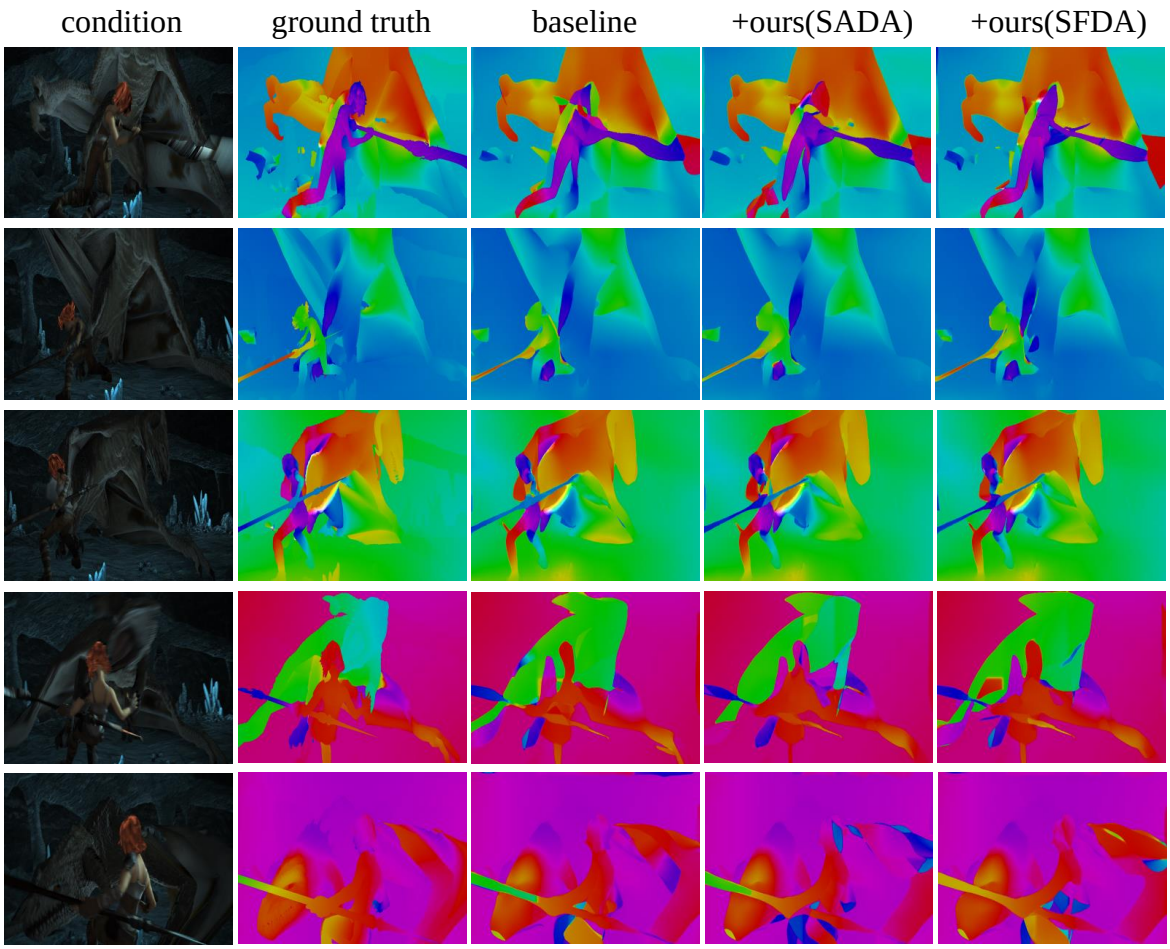

Figure 3. Competitive results of models w/ and w/o our methods in optical flow.