Flash-Split: 2D Reflection Removal with Flash Cues and Latent Diffusion Separation

Supplementary Material

In this supplementary material, we evaluate an additional flash/no-flash baseline [6] on real scenes (Sec. 8), demonstrate the respective roles of the two stages of our method (latent separation and cross-latent decoding) (Sec. 9), analyze our method's robustness to misalignment (Sec. 10), report baseline's performance on flash images (Sec. 11), and provide more training and inference details (Sec. 12).

8. Additional Flash/No-Flash-Based Baseline

In our main paper we compared our results with a flash/no-flash baseline Lei et al. [31], this is the most recent method on flash/no-flash based reflection separation method. We take Lei et al. [31]'s official code implementation from GitHub for their method and use their pretrained network checkpoints. However, as shown in Fig. 7, 8, 9, 10 of the main paper, the reflection separation performance of Lei et al. [31] are not satisfactory.

We additionally add another flash/no-flash baseline, Chang et al. [6], which proposes a siamese dense network (SDN) for reflection removal with flash and no-flash image pairs. We also use their official implementation plus their pretrained checkpoints. We evaluated this method using the same scenes shown in the main paper. The results are shown in Fig. 17,18,19,20. These four scenes correspond to Fig. 7.8,9,10 of the main paper. While Chang et al. [7] outperforms Lei et al. [31] on real data, it still does not fully separate the transmission component from the input flash and no-flash images. Our method still achieves much better reflection separation performance.

9. Respective Roles of Our 2-Stage Separation

As mentioned in our main paper, we decouple the reflection separation problem into two consecutive stages: (1) iterative latent diffusion separation and (2) cross-latent decoding. More specifically, in Stage 1, we iteratively separate the reflection and transmission within the latent space; in Stage 2, we restore fine image details to the separated latents while keeping the reconstruction faithful to the original scene, by using separated latent from Stage 1 as guidance to extract the sharp image features from the unseparated input image.

The respective effects of the two stages are shown in Fig. 14 and 15. To visualize the intermediate results after Stage 1 (iterative latent diffusion separation), we decode the separated transmission/reflection latents using a vanilla decoder [43]. We can clearly see that the iterative latent diffusion separation in Stage 1 already performs a good separation of reflection and transmission. However, these intermediate results still suffer from hallucinations and blurriness, due to the under-determinedness of the decoding process. In Stage 2, our cross-latent decoding significantly improves the sharpness and faithfulness of the reconstructed images by leveraging the high-frequency details contained in the original input images.

In summary, Stage 1 separates the transmission and reflection, while Stage 2 enhances the details.

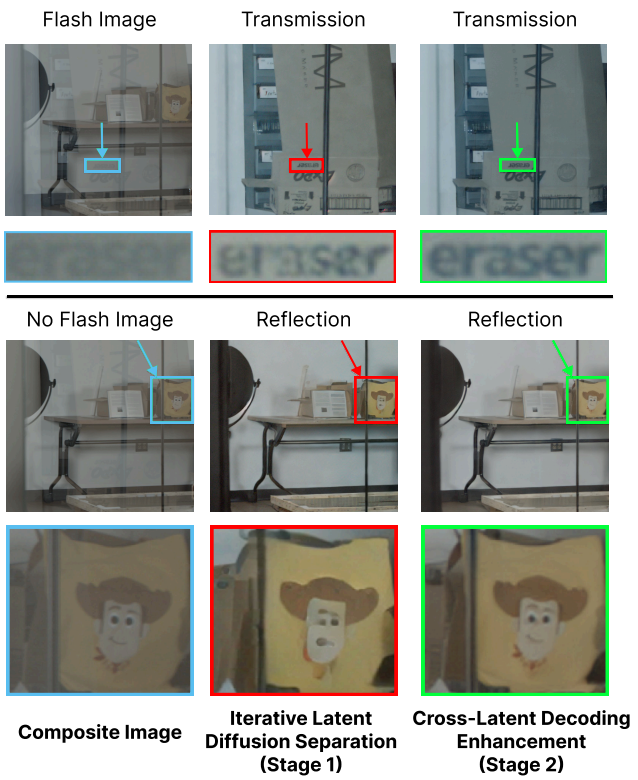

Figure 14. **Stage 1 for Separation; Stage 2 for Enhancement**. We visualize the intermediate results from our Iterative Latent Diffusion Separation in Stage I (middle column) and the final results from our Cross-latent Decoding in Stage 2 (right column). Stage 1 of our method performs good separation, and Stage 2 enhances the details while avoiding hallucinations. Note that in our method, Stage 1 only outputs the separated transmission/reflection latents, but in this figure, for the purpose of visualization, we decode the separated latents using a vanilla decoder from [43]. Additionally, note that the zoom-in texts ("eraser") shown in the top half of this figure have been flipped vertically for better readability.

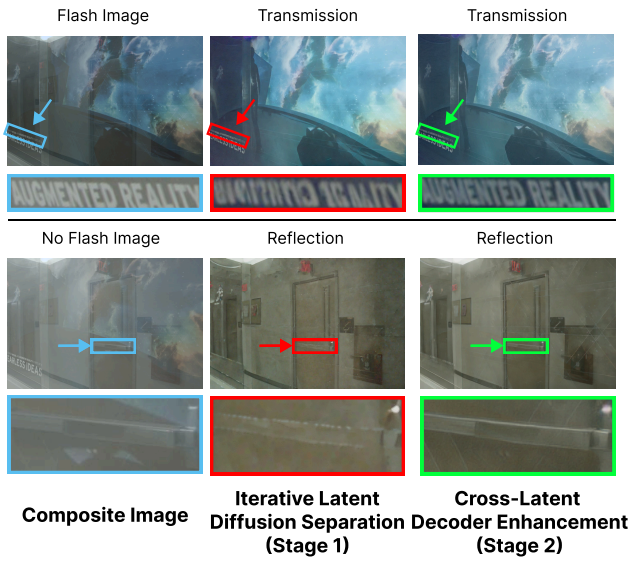

Figure 15. **Stage 1 for Separates; Stage 2 for Enhancement**. Same experiment as Fig. 14, but on a new scene. Stage 1 of our method perform good separation, and Stage 2 enhances the details while avoiding hallucinations. Note that the two small white triangles in the zoomed-in regions of the captured composite no-flash image (lower left corner) are from the transmitted scene, which aligns with our model's prediction.

10. Robustness Against Misalignment

To better understand our model's robustness to more severe misalignment, we intentionally increase the amount of misalignment between the captured flash and no-flash images, to a degree where our method fails. Fig. 16 shows that our method performs robustly against small to moderate camera motion (e.g., hand shake) ; however, in the case of extreme camera motion, (e.g., if the user is running or biking while capturing the flash/no-flash pair), our method might fail.

11. Software-based Methods Using Flash Image

The goal of this section is to show that our method performs better not because we use a camera flash, but rather because we use the cues from the flash/no-flash difference.

In our main paper, we visually compared our method with various software-based reflection removal methods. In those comparisons, we fed the no-flash images as inputs to the software-based methods. The rationale behind this choice is that these methods were trained on no-flash images, making the no-flash inputs in our real image evaluation more representative of their training distribution. Consequently, we believe that this approach provides a fair baseline for comparison.

However, one could argue that the flash images, which exhibit a stronger transmission component, might provide an advantage for software-based methods to better separate out the transmissions. To address this potential concern, we additionally run software-only baselines on the same scenes shown in the main paper, but using the *flash images* as inputs. The results are shown in Fig. 17,18, 19, 20. These four scenes correspond to Fig. 7.8,9,10 of the main paper.

In this case, our method still achieves much better reflection separation performance compared to software-based methods, which implies that, compared to the software-based methods, our method's superiority does not come from flash, but rather, comes from the flash/no-flash cues.

12. Additional Training and Inference Details

In this section, we provide additional details on the training and inference procedures for Stage 1 latent separation and Stage 2 cross-latent decoding. At a high level, our proposed pipeline is introduced in Fig. 5 and Sec. 3.3 of the main paper.

12.1. Training

Stage 1 Latent Separation. During Stage 1 latent separation training, we convert the flash and no-flash images to the latent space using the vanilla encoder from [43], and concatenate them in the latent channel dimension to form an input latent image z. We then take the target ground truth transmission/reflection images and encode them into ground truth image latents s_0. Now, we sample a noise image latent ϵ with the same dimension as the ground truth image latent. We then add the noise image to the ground truth image latent using a random noise level t:

$$s_t = \sqrt{\alpha_t} s_0 + (\sqrt{1-\alpha_t})\epsilon \tag{4}$$

Here $\{\alpha_t\}, t \in \{1, ..., T\}$ is the noise schedule specific to the diffusion model. We use the default DDPM [17] scheduler of the Stable Diffusion 2.1 model [43] with $T = 1000$ steps for training. We also use the annealed multi-resolution noise [23] instead of standard Gaussian noise [17].

Our UNet then takes the input latents from the flash/no-flash input images (z) and noised ground truth latents (s_t) from the ground truth transmission/reflection images and predicts a noise $\hat{\epsilon}$. Our training objective is to minimize the L2 loss between the injected noise ϵ and the noise predicted by the UNet $\hat{\epsilon}$. Note that the ground truth transmission/reflection images are only used for training, and not used for inference (see Sec. 12.2 for details).

We use the exact simulated and real datasets as proposed in Lei et al. [31], which contains sets of flash/no-flash pairs and corresponding ground truth transmission and reflection images. The input images are randomly cropped to 384×384 sized patches for training. To simulate misaligned flash/no-flash image pairs, we follow Lei et al. [31] and keep the no-flash images intact and do a

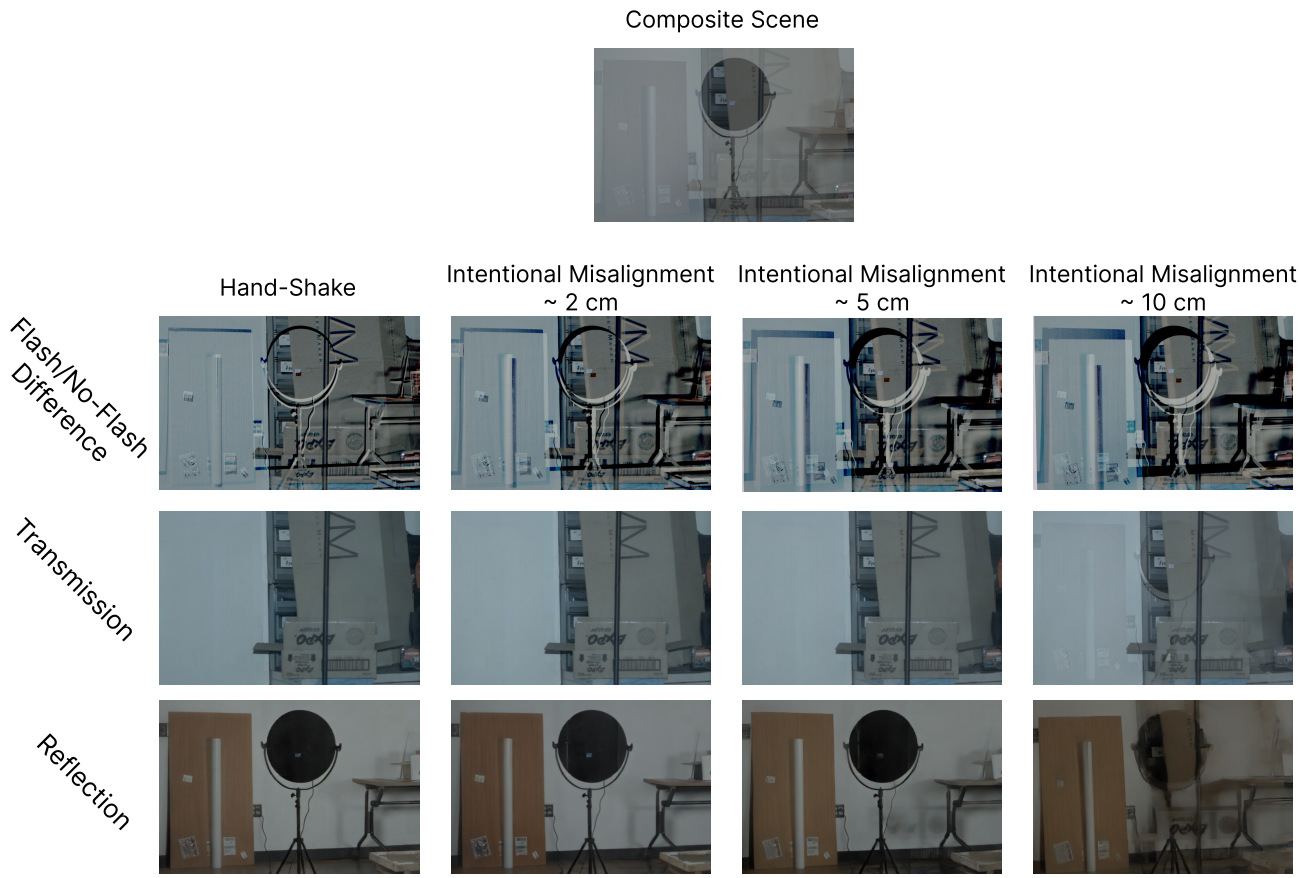

Figure 16. **Our Method's Robustness to Different Degrees of Misalignment**. While our model effectively handles misaligned flash and no-flash images due to handshake, we intentionally further increase the flash/no-flash misalignment to find out when will our model fail. Note that we assume the scene to be static and the misalignment comes from camera motion. The results show our method's robustness against estimated misalignment of 2 and 5 centimeters, respectively. However, when the misalignment exceeds 10 centimeters, our method's performance deteriorates. This shows that our method performs robustly against small to moderate camera motion (e.g., hand shake) while baselines completely fail; however, in the case of very severe camera motion, (e.g., if the user is running or biking while capturing the flash/no-flash pair), our method might fail.

monocular-depth-guided image misalignment to generate a misaligned flash image.

Stage 2 Cross-Latent Decoder. Our cross-latent decoder is trained to learn a mapping from the latents separated by our Stage 1 (iterative latent diffusion separation) to the ground truth transmission/reflection images, using unseparated input images as guidance.

The architecture of our cross-latent decoder is modified from the pre-trained VAE component in [43] by adding skip connections with zero convolutions. We trained separate cross-latent decoders for reflection and transmission. For transmission, we use the input flash image as the composite image, since the flash image contains a higher proportion of transmission compared to the no-flash image. Conversely, for reflection, we use the input no-flash image as the composite image, since the no-flash image contains a higher proportion of reflection compared to the flash image. Our cross-latent decoder takes in both the unseparated input image and the separated latent from Stage 1 as inputs, and outputs a separated RGB image. We train the model by minimizing the difference between the decoded and the ground truth transmission/reflection image. We use an equally weighted sum of L1, SSIM [4], and LPIPS [66] losses to supervise the training. We take the separated transmission/reflection latents from Stage 1 and group them with the ground truth transmission/reflection, as well as the input flash/no-flash images to form our Stage 2 training data. Specifically, we take the misaligned training images crops with size 384×384 from Lei et al. [31] as the input and run inference on our trained Stage 1 model for 20 DDIM [46] denoise iterations. See Sec. 12.2 on Stage 1 inference for more details.

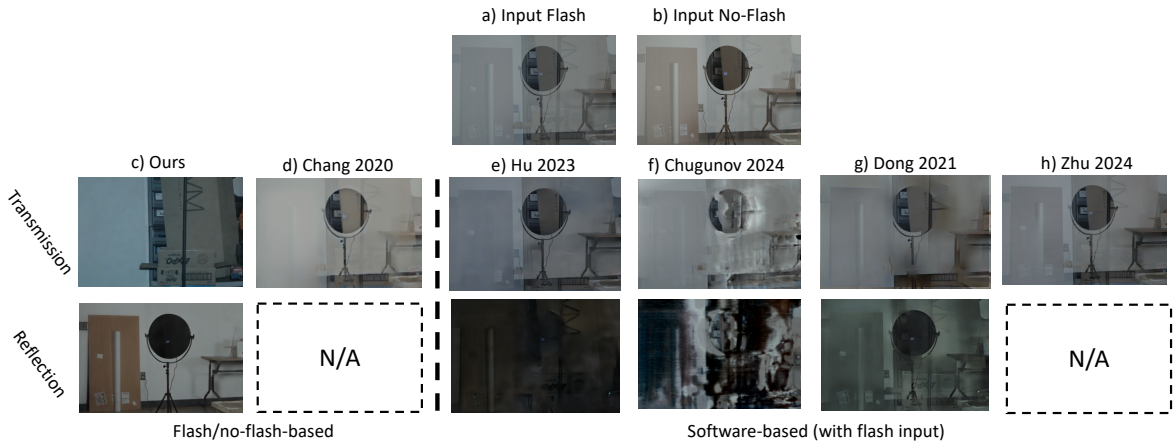

Figure 17. **Real Experiment: The Lab Scene.** We compare with an additional flash/no-flash-based baseline Chang et al. [7]. Chang et al. [7] can only predict the transmission, not the reflection, thus the "N/A". Although Chang et al. [7] achieves better results than Lei et al. [31] on the real data, it still cannot completely separate the transmission component from the input flash/no-flash images. The software-based results shown in the real experiment are obtained using the no-flash image as the input. This figure provides additional results to Fig. 7 of the main paper.

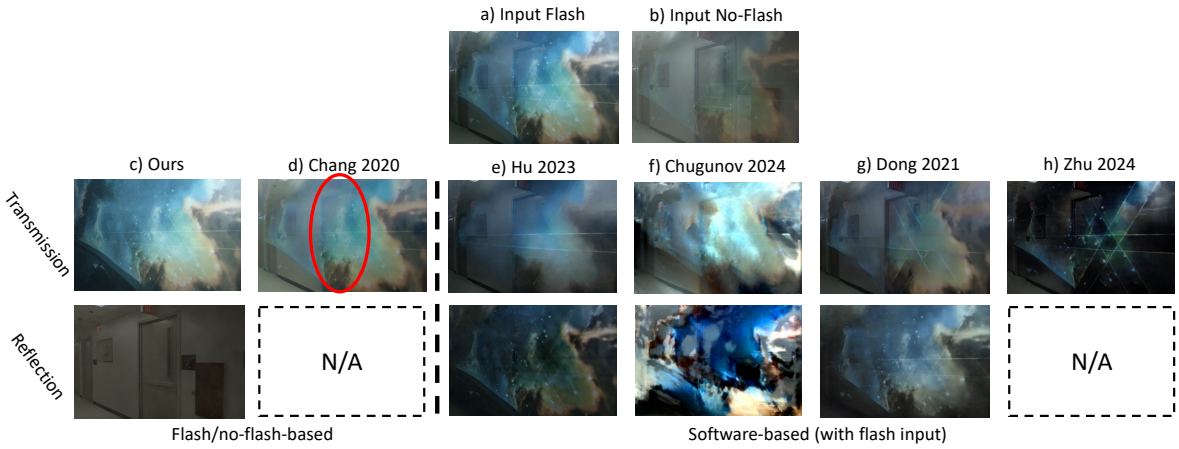

Figure 18. **Real Experiment: The Poster Scene.** We compare with an additional flash/no-flash-based baseline Chang et al. [7]. Chang et al. [7] can only predict the transmission, not the reflection, thus the "N/A". Although Chang et al. [7] achieves better results than Lei et al. [31] on the real data, it still cannot completely separate the transmission component from the input flash/no-flash images. We circle the areas where Chang et al. [7] did not correctly separate the door in the reflection. The software-based results shown in the real experiment are obtained using the no-flash image as the input. This figure provides additional results to Fig. 8 of the main paper.

12.2. Inference

After training, our diffusion model can be used to recover transmission/reflection images from any flash no-flash pair. We convert the flash and no-flash images to the latent space using the vanilla encoder from [43], and concatenate them in the latent channel dimension to obtain the input latent image z. Our output prediction latent for transmission/reflection s is initialized from random Gaussian noise. We iteratively denoise the separated reflection/transmission images using our trained dual-branch UNet under the guidance of input flash/no-flash images. At each denoising iteration, we concatenate the input and output prediction latent images and feed them to the UNet. We then update the prediction latent based on the predicted noise of our UNet and the current time step.

$$s_{t-1} = DDIM(s_t, \hat{\epsilon}_t, t) \tag{5}$$

Here t is the denoising timestep for the current iteration. This denoising timestep corresponds to the amount of noise contained in the output latent and decreases with every subsequent denoising iteration. s_t is the output separated transmission/reflection image at timestep t, $\hat{\epsilon}_t$ is the noise predicted by the UNet at timestep t, and s_{t-1} is the output separated transmission/reflection latent image at the next timestep $t-1$ ready for the next iteration. We use the DDIM [46] scheduler for inference, which uses skipping step updates to enable fewer denoising iterations and faster

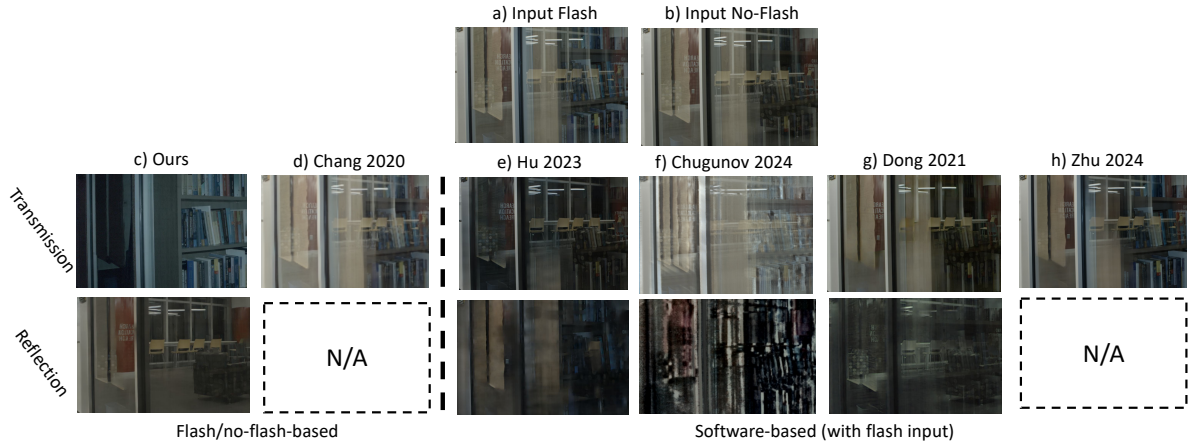

Figure 19. **Real experiment: the Office Scene.** We compare with an additional flash/no-flash-based baseline Chang et al. [7]. Chang et al. [7] can only predict the transmission, not the reflection, thus the "N/A". Although Chang et al. [7] achieves better results than Lei et al. [31] on the real data, it still cannot completely separate the transmission component from the input flash/no-flash images. The software-based results shown in the real experiment are obtained using the no-flash image as the input. This figure provides additional results to Fig. 9 of the main paper.

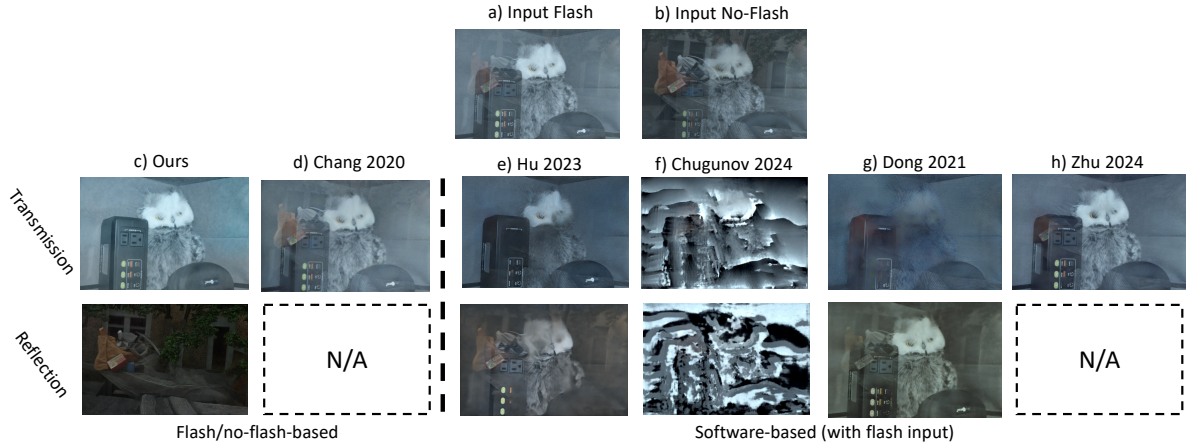

Figure 20. **Real experiment: the Outdoor Scene.** Chang et al. [7] can only predict the transmission, not the reflection, thus the "N/A". Although Chang et al. [7] achieves better results than Lei et al. [31] on the real data, it still cannot completely separate the transmission component from the input flash/no-flash images. The software-based results shown in the real experiment are obtained using the no-flash image as the input. This figure provides additional results to Fig. 10 of the main paper.

inference. We use 50 denoising iterations for inference.

Inference continues to Stage 2 where we take the separated transmission/reflection latent outputs from Stage 1 and feed them to the decoder of their respective cross-latent decoders. Finally, the Stage 2 cross-latent decoders output the refined transmission/reflection RGB images.

13. Additional Results for Analysis of Latent Diffusion Separation

We show additional results from real world scenes in [61] for the VAE and pretraining analysis in Sec. 4 of the main paper. We can again see that the model using VAE trained from scratch (Fig. 21e) achieves almost comparable performance to the model using pretrained SD (Fig. 21f), despite some minor artifacts. Notably, this model still performs better than the flash/no-flash baseline method [31] (Fig. 21b). On the other hand, the models without the VAE (Fig. 21c & d) cannot effectively separate the reflection. These findings further support our main paper's analysis (Section 4), emphasizing that the effectiveness of latent diffusion-based separation significantly contributes to the overall success of our method.

14. Additional Ablations on Pretraining

In this section we augment our analysis in Sec. 4 of the main paper by showing further ablation studies on the effect of the pretrained diffusion model on our method.

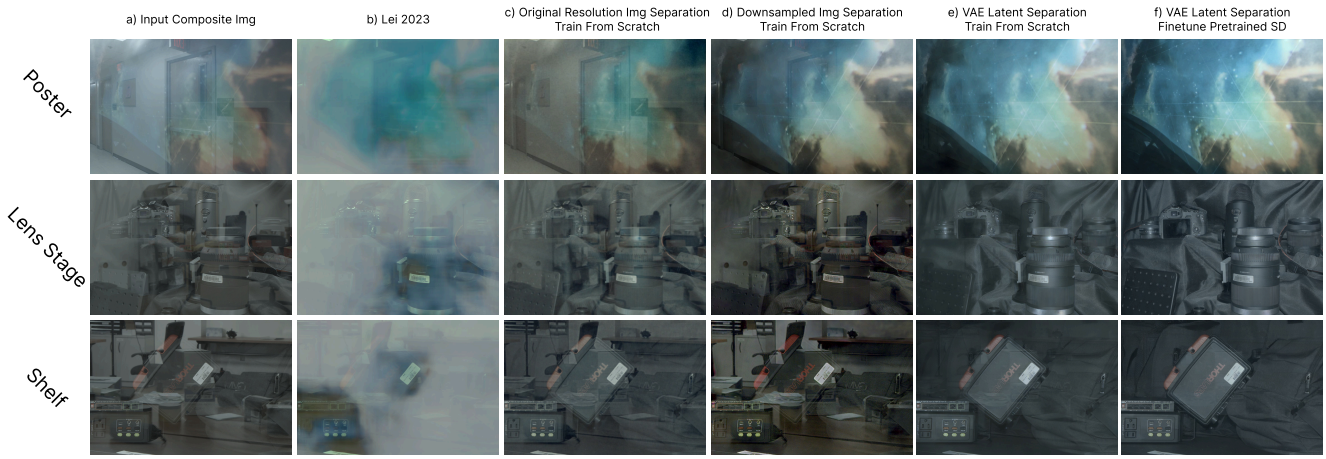

Figure 21. **Additional Real World Results for the VAE and Pretraining analysis**. We show further comparisons on additional real world scenes from [61] between models in our VAE and pretraining analysis in Sec. 4 of the main paper. Real experiment results show that the VAE is important for effective reflection separation. Notably, the model using VAE trained from scratch (d) achieves almost comparable performance to the model using pretrained Stable Diffusion (SD) [43] (e), despite some minor artifacts, while outperforming the flash/no-flash baseline (b) [31].

	PSNR ↑	SSIM ↑	LPIPS ↓
Stable Diffusion v1.1	31.75	0.964	0.048
Stable Diffusion v1.4	31.58	0.964	0.049
Stable Diffusion v2	31.61	0.963	0.048
Stable Diffusion v2.1	31.84	0.963	0.047

Table 2. **Quantitative Comparison of Transmission Prediction With Different Pretrained Base Diffusion Models**. We trained variants of our model using different pretrained Stable Diffusion (SD) [43] versions without the inter-branch cross attention. Newer versions of the SD model are additionally trained on top of older models and therefore should contain more prior information. However, quantitative results on the real dataset of [31] indicate that performance remains similar when using the legacy models (v1.1 and v1.4) compared to the newer v2 and v2.1 versions.

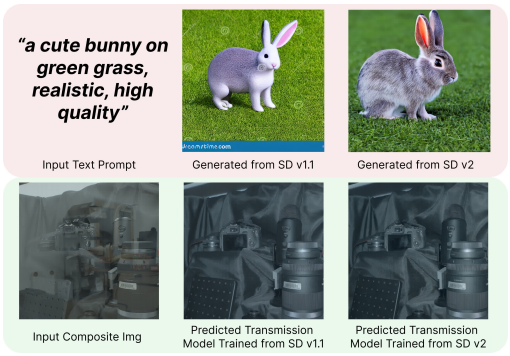

Figure 22. **Qualitative Comparison of Separation with Different Stable Diffusion (SD) [43] Versions**. While legacy SD versions (SD v1.1) are less capable of text-to-image generation compared to newer models (SD v2), when repurposed for reflection separation using our method, both models produce near-identical reconstructions.

14.1. Ablation on the Pretrained Model Versions

To study the effect of pretraining, we analyzed how starting from different pretrained Stable Diffusion (SD) versions influence our model's performance. Newer SD versions (v2 and v2.1) are trained on older models and use higher-resolution image data, suggesting that they incorporate additional prior knowledge. However, training our method with various SD versions yielded largely consistent results (Tab. 2). Qualitative results on real-world data from [61] also revealed near-identical outcomes (Fig. 22), indicating that the better generative capabilities of newer SD models do not necessarily enhance reflection separation. We believe this supports our conclusion from Sec. 4 of the main paper that latent diffusion separation, rather than pretraining, drives our method's success.

14.2. Ablation on Pretrained Model by Partially Freezing Pretrained Weights

We conduct an additional ablation study on the effect of pretraining, where we freeze components of the pretrained Stable Diffusion v2 [43] UNet and apply our method. To purely test the performance of the diffusion UNet, all of our models are trained without inter-branch cross-attention. We show quantitative results in Tab. 3 testing on the real dataset of [31]. Only freezing the midblock achieves the best closest performance compared to fine-tuning the full UNet. In contrast, freezing the upblocks leads to the largest performance degradation, highlighting that UNet upsampling layers play a critical role in refining detailed spatial and

	PSNR ↑	SSIM ↑	LPIPS ↓
Full UNet	31.61	0.963	0.048
Frozen Downblocks	30.92	0.955	0.057
Frozen Midblock	31.42	0.960	0.050
Frozen Upblocks	30.02	0.947	0.062

Table 3. **Quantitative Comparison of Transmission Prediction when Freezing Weights of the Pretrained Stable DIffusion UNet**. We trained variants of our model starting from pretrained Stable Diffusion v2 (SD v2) [43], selectively freezing different components of the pretrained diffusion UNet. All of our models are trained without inter-branch cross-attention to purely test the performance of the diffusion UNet. When testing on the real dataset of [31]. Freezing the midblock achieves the closest performance to fully fine-tuning the model, indicating midblock weights are robust and require less adaptation, whereas freezing upblocks results in the largest performance degradation, suggesting that decoder layers critically require fine-tuning for transmission prediction.

structural outputs required specifically for transmission prediction.

15. Perpendicular Capture

Figure 23. **Perpendicular Capture.** We show a new scene where the camera is in a car pointing outside, *almost* perpendicular to the side window. Our model can still remove reflections. While in theory, a *perfectly* perpendicular capture may cause flare, we empirically found it very easy to circumvent by slightly adjusting the viewpoint. In this case, our method(right) still archieves better reflection removal comapred to the single image baseline [21].

In Sec. 3.1 of the main paper, we introduced that flash/no-flash photography will work if the glass is not *exactly* perpendicular to the camera viewing direction. In this case, the flash illumination reflects away from the camera sensor upon hitting the reflective surface, thereby preventing lens flare from appearing on the captured image. In practice, we found that this condition is easy to fulfill. As we show in Fig. 23, even in the case where the camera is *almost* perpendicular to the window, the captured image will not have flare and our method can remove reflections effectively. While the flash illumination will cause secondary reflections (e.g., light that hits the glass, bounces to the reflected scene, and then gets reflected elsewhere), we rarely observe secondary reflections in practice. Therefore, it is reasonable to assume that there are minimal changes to the intensity of the reflected scene.