

DiffusionLight: Light Probes for Free by Painting a Chrome Ball

Pakkapon Phongthawee*¹
Amit Raj³

Worameth Chinchuthakun*^{1,2}
Varun Jampani⁴

Pramook Khungurn⁵

Nontaphat Sinsunthithet¹
Supasorn Suwajanakorn¹

¹ VISTEC

² Tokyo Tech

³ Google Research

⁴ Stability AI

⁵ Pixiv

<https://diffusionlight.github.io/>



Figure 1. We leverage a pre-trained diffusion model (Stable Diffusion XL) for light estimation by rendering an HDR chrome ball. In each scene, we show our normally exposed chrome ball on top and our underexposed version, which reveals bright light sources, on the bottom.

Abstract

We present a simple yet effective technique to estimate lighting in a single input image. Current techniques rely heavily on HDR panorama datasets to train neural networks to regress an input with limited field-of-view to a full environment map. However, these approaches often struggle with real-world, uncontrolled settings due to the limited diversity and size of their datasets. To address this problem, we leverage diffusion models trained on billions of standard images to render a chrome ball into the input image. Despite its simplicity, this task remains challenging: the diffusion models often insert incorrect or inconsistent objects and cannot readily generate chrome balls in HDR format. Our research uncovers a surprising relationship between the appearance of chrome balls and the initial diffusion noise map, which we

utilize to consistently generate high-quality chrome balls. We further fine-tune an LDR diffusion model (Stable Diffusion XL) with LoRA, enabling it to perform exposure bracketing for HDR light estimation. Our method produces convincing light estimates across diverse settings and demonstrates superior generalization to in-the-wild scenarios.

1. Introduction

Single-view lighting estimation is the problem of inferring the lighting conditions from an input image. In this work, we represent lighting as an environment map [8], which facilitates seamless insertion of virtual objects, including highly reflective ones. This problem is ill-posed because the environment map extends beyond the limited field of view of the input image. Moreover, the output must have a high dynamic range (HDR) to capture the true intensity of the incoming light. These difficulties have spurred numerous attempts to

*Authors contributed equally to this work.

regress HDR environment maps from LDR images.

A common strategy used in state-of-the-art techniques is to train a neural regressor with a dataset of HDR panoramas. For example, StyleLight [59] trains a GAN on thousands of panoramas and, at test time, uses GAN inversion to find a latent code that generates a full panorama whose cropped region matches the input image. Everlight [12] trains a conditional GAN on 200k panoramas to directly predict an HDR map from an input image. However, these training panoramas offer limited scene variety, often featuring typical viewpoints like those from a room’s center due to tripod use. Panoramas featuring elephants from inside a Safari Jeep, would be extremely rare (Figure 8). So, how can we estimate lighting in-the-wild for any image under any scenario?

Our key idea is to use a pre-trained large-scale text-to-image (T2I) diffusion model to render a chrome ball into the input image.* One clear advantage of our approach is the ability to leverage the image prior in the T2I diffusion models. Our method also does not assume a known camera pose, unlike many methods that outpaint panoramas [12, 59].

Inpainting a chrome ball into an image, while seemingly simple, poses challenges even for SOTA diffusion models with inpainting capabilities [6, 7, 63, 64] (see Figure 2). These models frequently fail to generate a chrome ball or generate one with undesirable patterns (e.g., a disco ball with tiny square mirrors) that do not convincingly reflect environmental lighting. Another key limitation is that these diffusion models were trained on low dynamic range (LDR) images and cannot produce HDR chrome balls.

To produce consistent, high-quality chrome balls, our solution involves three key ideas. First, we make inserting a ball reliable by using depth map conditioning [69] on top of a standard T2I diffusion model (Stable Diffusion XL [45]). Second, to better mimic genuine chrome ball appearance, we fine-tune the model using LoRA [30] on a small number of synthetically generated chrome balls. Third, we start the diffusion sampling process from a good initial noise map, and we propose an algorithm to find one. The last idea is based on surprising findings we discovered about diffusion model behavior and chrome ball appearance.

To generate HDR chrome balls, our idea is to generate and combine multiple LDR chrome balls with varying exposure values, similar to exposure bracketing. One naive method that requires no additional training is to use two text prompts: one for generating a standard chrome ball, and the other with “black dark” added to the text prompt. Alternatively, we propose utilizing our previous LoRA [30] to map continuous interpolations of the two text prompts to a target ball image with varying, known exposures. This enables specifying the exposure values of the generated balls at test time. While this fine-tuning requires a small number of panoramas for

training, the core task of producing reflective chrome balls still relies on the model initial’s capability, which remains generalizable to a broad range of scenes.

We evaluate our method against StyleLight [59], EverLight [12], Weber et al. [62], and EMLight [67] on standard benchmarks: Laval Indoor [20] and Poly Haven [3] datasets. Our method is competitive with StyleLight and achieves better performance on two out of three metrics across both datasets, while ranking second and third, when tested using a protocol in [12] on Laval Indoor. Note that the baselines were directly trained on the datasets, with some tailored to indoor scenes. When applied to more challenging, in-the-wild images beyond the benchmarks, our method still produces convincing results while the baselines fail to do so.

To summarize, our contributions are:

- A novel light estimation technique that generalizes across a wide variety of scenes based on a simple idea of inpainting a chrome ball using a pre-trained diffusion model.
- An iterative inpainting algorithm that enhances quality and consistency by leveraging our discovered relationship between the initial noise map and chrome ball appearances.
- A continuous LoRA fine-tuning technique for exposure bracketing to produce HDR chrome balls.

2. Related Work

Lighting estimation. While there exists light estimation methods that require specific light probes [13, 23, 37, 44, 61, 66] or naturally occurring elements [9, 43, 65] in the image, we focus our review on approaches that do not require such probes. These methods are designed to handle indoor [21, 22, 62, 67] or outdoor scenes [27, 28], or both [12, 34].

Unlike earlier approaches that rely on limited lighting models [21, 22, 27, 29, 31–33], modern techniques have shifted towards predicting 360° HDR environment maps, which are essential for tasks such as virtual insertion of highly reflective objects. A common strategy among these technique involves regressing an LDR input with a limited field of view into an HDR map with neural networks. Gardner et al. [20] use a CNN classifier to locate lights using a large dataset of LDR panoramas and then fine-tune the CNN to predict HDR maps using a smaller HDR dataset. Hold-Geoffroy et al. [28] first train an autoencoder for outdoor sky maps, then use another network to encode and decode an input image. Weber et al. [62] predict LDR maps along with parametric light sources [21], also with a CNN. Somanath et al. [52] introduce two loss functions based on randomly masked L1 and cluster classification to enhance estimation accuracy. Zhan et al. [67, 68] a two-step process involving a spherical light distribution predictor and an HDR map predictor. These methods are typically demonstrated in either indoor or outdoor settings due to the specific lighting models or the lack of sufficiently large and diverse HDR datasets.

To solve both indoor and outdoor settings, LeGendre et al.

*Note that the practice of *physically* placing a chrome ball into the scene dates back to the early days of computer graphics and photography [15].

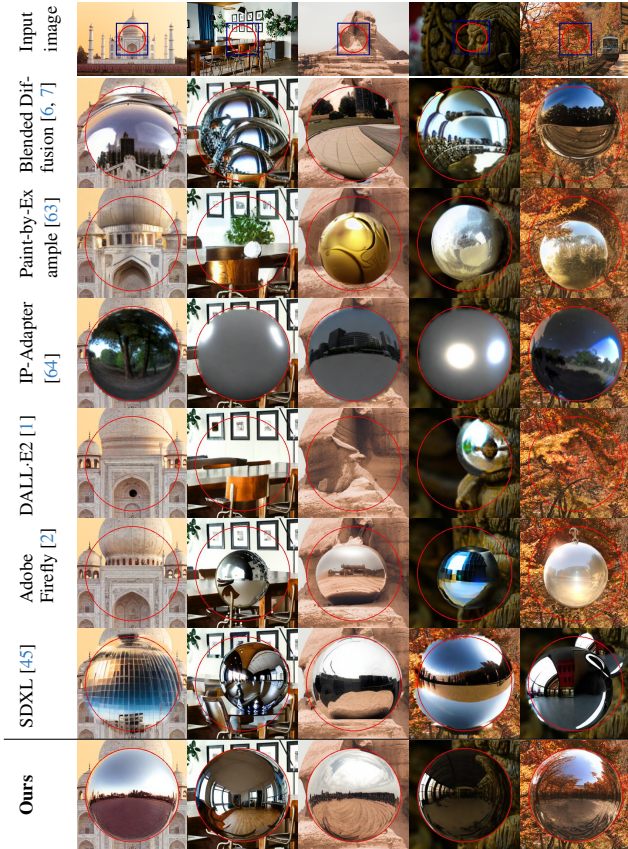


Figure 2. Chrome ball inpainting results from various methods. The red circle indicates the inpainted region, and we show a zoomed-in view of the blue crop. These diffusion models tend to produce distorted balls with undesirable textures, or completely fail to produce a ball and instead just reconstruct the original content. Our method addresses all these issues and precisely follows the inpainting mask.

[34] collected a new dataset of natural scenes captured using a mobile device with three reflective probing balls. Their method, DeepLight, regresses HDR lighting from an input image using a loss function that minimizes the difference between ground truth and rendered balls under the predicted lighting. Dastjerdi et al. [12]’s EverLight combines multiple indoor and outdoor datasets and predicts an editable lighting representation, which then conditions a GAN to generate a full HDR map. Some GAN-based techniques focus on outpainting an input image to a 360° panorama [4, 5, 11], but they perform poorly in light estimation due to the LDR prediction [12, 60]. In contrast, StyleLight by Wang et al. [59] trains a two-branch StyleGAN network to predict LDR and HDR maps from noise and, at test time, uses GAN inversion to predict an HDR map from an input image.

Despite many attempts to combine indoor and outdoor panoramas, the resulting datasets remain small and limited in diversity. In contrast, we leverage diffusion models trained on billions of images, leading to better generalization.

Image inpainting using diffusion models. Our method

relies on text-conditioned diffusion models to synthesize chrome balls. While there are many diffusion models for image editing or inpainting [6, 7, 41, 46, 50, 54, 57, 58, 63, 64], we only discuss work related to object insertion in images. Blended Diffusion [6, 7] allows arbitrary object insertion using text prompts for masked regions in an input image. This is done using guided sampling [16] based on the cosine distance between the CLIP embedding of the inpainted region and the prompt. Paint-by-Example [63] uses example images and their CLIP embeddings as prompts to condition diffusion models. ControlNet [69] and IP-Adapter [64] enable conditioning pre-trained diffusion models for image generation using both image and text prompts. Commercial products like DALL-E2 [1] and Adobe Firefly[2] also offer similar inpainting capabilities with text prompts.

Unfortunately, these methods fail to consistently produce chrome balls or produce ones that do not convincingly reflect the environmental lighting (see Figure 2).

Personalized text-to-image diffusion models. Our work enhances pre-trained diffusion models for consistent generation of known objects, which is related to *personalized* image generation. For this task, models are fine-tuned using single or a few reference object images while preserving their unique appearance. DreamBooth [51] uses a special token during fine-tuning to represent the object while maintaining the pre-trained distribution. Gal et al. [19] introduce a learned word in the embedding space for representing referenced objects, and Voynov et al. [56] adopt separate word embeddings per network layer. Additionally, there are studies exploring techniques to simplify fine-tuning of diffusion models [25, 30, 47], with LoRA [30] being a popular choice that enforces low-rank weight changes.

These methods can be adapted to our task by providing chrome ball images for fine-tuning. In fact, a portion of our pipeline can be viewed as a variant of DreamBooth with LoRA, albeit without the prior preservation loss.

3. Approach

Given a standard LDR input image, our goal is to estimate the scene’s lighting as an HDR environment map. Our solution is based on inserting a chrome ball into the image using a diffusion model, then unwraps it to an environment map. We tackle two key challenges of (1) how to consistently generate chrome balls and (2) how to use an LDR diffusion model to generate HDR chrome balls.

Overview. As shown in Figure 3, our key component is based on Stable Diffusion XL [45] with depth-conditioned ControlNet [69]. We first predict a depth map from the input image using an off-the-shelf depth prediction network [48, 49]. Then, we paint a circle both at the depth map’s center with the distance closest to the camera (visualized as white) and in an inpaint mask. We feed them along with the input

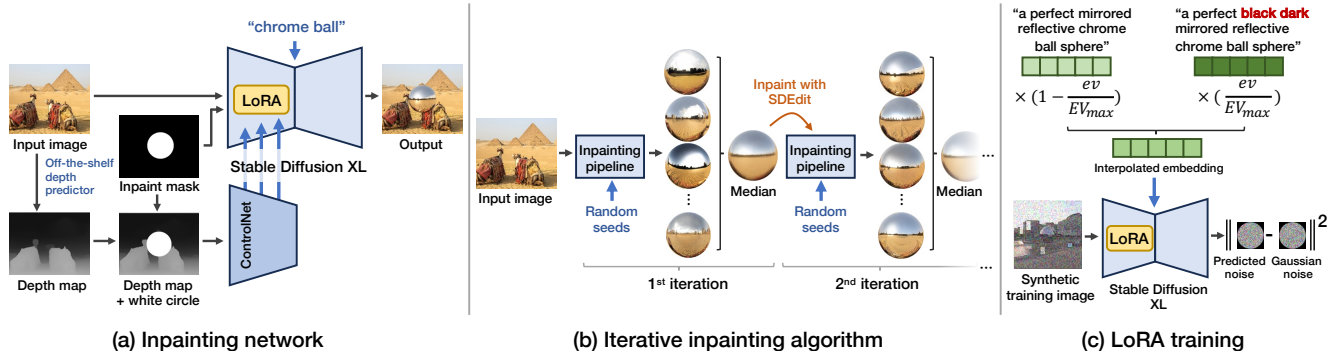


Figure 3. (a) We use Stable Diffusion XL [45] with depth-conditioned ControlNet [69] to inpaint a chrome ball. (b) Our iterative inpainting algorithm helps improve generation quality and consistency by constraining the initial noise through sample averaging (Section 3.2). (c) We train LoRA for exposure bracketing, which produces multiple LDR chrome balls with varying exposures for HDR merging (Section 3.3).

image and the prompt “a perfect mirrored reflective chrome ball sphere” to the diffusion model.

We make two improvements to the above base model. First, we propose a technique called ‘iterative inpainting’ to locate a neighborhood of good initial noise maps that lead to consistent and high-quality chrome balls (Section 3.2). Second, to further improve the generated appearance and generate multiple LDR images for exposure bracketing, we fine-tune the diffusion model using LoRA [30] on a set of synthetically generated chrome balls with varying exposures (Section 3.3). To explain our method in detail, we first cover background and standard notations of diffusion models.

3.1. Preliminaries

Diffusion models [26] form a family of generative models that can transform a prior distribution (Gaussian distribution) to a target data distribution p_{data} by learning to revert a Gaussian diffusion process. Following the convention in [53], it is represented by a discrete-time stochastic process $\{\mathbf{x}_t\}_{t=0}^T$ where $\mathbf{x}_0 \sim p_{\text{data}}$, and $\mathbf{x}_t \sim \mathcal{N}(\mathbf{x}_{t-1}; \sqrt{\alpha_t/\alpha_{t-1}}\mathbf{x}_{t-1}, (1-\alpha_t/\alpha_{t-1})\mathbf{I})$. The decreasing scalar function α_t , with constraints that $\alpha_0 = 1$ and $\alpha_T \approx 0$, controls the noise level through time. It can be shown that

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\epsilon, \text{ where } \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (1)$$

A diffusion model is a neural network ϵ_θ trained to predict from \mathbf{x}_t the noise ϵ that was used to generate it according to (1). The commonly employed, simplified training loss is

$$\mathcal{L} = \mathbb{E}_{\mathbf{x}_0, t, \epsilon} \|\epsilon_\theta(\sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\epsilon, t, \mathbf{C}) - \epsilon\|_2^2, \quad (2)$$

where \mathbf{C} denotes conditioning signals such as text. The trained network can then be used to convert a Gaussian noise sample to a data sample through several sampling methods [26, 53, 70]. In this paper, we use Stable Diffusion [45, 50], which operates on latent codes of images according to a variational autoencoder (VAE). As such, we use \mathbf{x}_t to denote images and \mathbf{z}_t to denote latent codes.

LoRA fine-tuning. Instead of fine-tuning each full weight matrix $\mathbf{W}_i \in \mathbb{R}^{m \times n}$, LoRA[30] optimizes a learnable low-rank residual matrix $\Delta\mathbf{W}_i = \mathbf{A}_i\mathbf{B}_i$, where $\mathbf{A}_i \in \mathbb{R}^{m \times d}$, $\mathbf{B}_i \in \mathbb{R}^{d \times n}$, and $d \ll m, n$. The final weight matrix is given by $\mathbf{W}'_i = \mathbf{W}_i + \alpha\Delta\mathbf{W}_i$, where α is the “LoRA scale.”

3.2. Iterative inpainting for improving quality

We found that the base depth-conditioned Stable Diffusion model could reliably *insert* a chrome ball as opposed to merely recovering the masked out content. However, the chrome balls often contain undesirable patterns and fail to convincingly reflect environment lighting (Figure 4).

Our first improvement stems from a few observations: The same initial noise map leads to the generation semantically similar inpainted areas *regardless* of the input image. For instance, there exists a “disco” noise map that consistently produces a disco ball *across* different input images, while a good noise map almost always produces a reflective chrome ball (Figure 4). When searching for images of a “chrome ball” on the Internet, not all results match the specific reflective chrome ball we want. So, the encoded semantics found within the noise map are perhaps understandable, as text prompts alone cannot encode all visual appearances of “chrome ball.” Here, adding “disco” to the negative prompt may fix this specific instance, but many other failure modes are not as easy to describe and exclude via text prompts.

Another observation is that the average of multiple ball samples tends to approximate the overall lighting reasonably well, but the average ball itself is too blurry and not as useful.

Based on these insights, we propose a simple algorithm to automatically locate a good noise neighborhood by sample averaging. Specifically, we first inpaint N chrome balls onto an input image using different random seeds. Then, we calculate a pixel-wise median ball and composite it back to the input image. Let us denote this composited image by \mathbf{B} . To sample a better chrome ball, we apply SDEdit [41][†] by adding noise to \mathbf{B} to simulate the diffusion process up to time

[†]Commonly known as “image-to-image” by Stable Diffusion users [17].

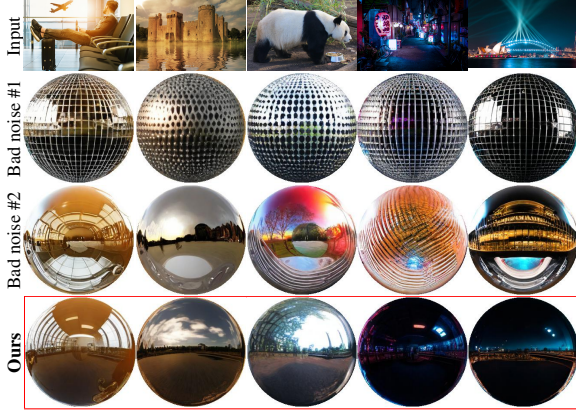


Figure 4. Observations: the initial noise map encodes some semantic patterns. We found certain noise maps to consistently produce a disco chrome ball or bad patterns *across* input images. In contrast, our results produce consistently clean chrome balls.

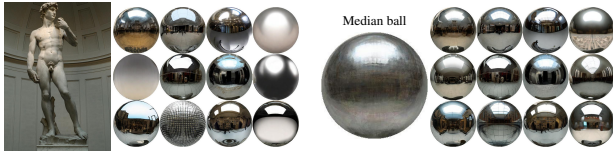


Figure 5. Balls before (left) and after (right) iterative inpainting.



Figure 6. Simply adding “black dark” to the prompt allows the overexposed sun to emerge. However, we need a way to specify the target EV, which is addressed by our LoRA (Section 3.3).

step t : $\mathbf{B}' = \sqrt{\alpha_t}\mathbf{B} + \sqrt{1 - \alpha_t}\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $t < T$, the maximum timestep. Then, we continue denoising \mathbf{B}' starting from t to 0. In our implementation, $t = 0.8T$.

We can repeat the process by using SDEdit to generate another set of N chrome balls from the output and recompute another median chrome ball. This repetition not only minimizes artifacts and spurious patterns from incorrect ball types but also enhances the consistency of light estimation.

In our implementation, we perform two iterations of median computation. This involves generating N chrome balls with standard diffusion sampling to compute the first median, performing N SDEdits from the first median to compute the second, and doing one last SDEdit to generate the output.

3.3. Predicting HDR chrome balls

The main issue with using pre-trained diffusion models for HDR prediction is that they have never seen HDR images. Nonetheless, these models can still indirectly learn about HDR and the wide range of luminance through examples of under and overexposed images in their training sets. This ability is evident in our experiment where adding ‘black

dark’ to our text prompt can reduce the overexposed white sky, allowing the round sun to emerge on outdoor scenes (see Figure 6). To leverage this ability, we propose to use the exposure bracketing technique by inpainting multiple chrome balls with different exposure values and combining them to produce a linearized HDR output. Our idea is to train a LoRA to steer the sampling process such that the output conforms to the appearances associated with specific exposure compensation values (EVs).

Training set. We construct our LoRA training set using HDR panoramas synthetically generated from Text2Light [10] to avoid direct access to scenes in benchmark datasets. As illustrated in Figure 3, each training pair consists of a random EV, denoted by ev , and a panorama crop with a chrome ball rendered with $EV=ev$ at the center. This crop is constructed by projecting a full HDR panorama to a small field-of-view image and then tone-mapping to LDR *without* exposure compensation (EV0). The chrome ball is rendered using the panorama as the environment map in Blender [18], but its luminance is scaled by 2^{ev} before being tone-mapped to LDR. Following [59], we use a simple $\gamma=2.4$ tone-mapping function and map the 99th percentile intensity to 0.9.

Here, we assume that the typical output images from the diffusion model have a mean EV of zero. For light estimation purposes, our focus is on recovering high-intensity light sources, which are crucial for relighting and are captured in underexposed or negative EV images. Therefore, we randomly sample the EV values from $[EV_{\min}, 0]$.

Training. To generate a chrome ball with a specific EV, we condition our model on an interpolation of two text prompts as a function of ev . The two prompts are the original prompt and the original with “black dark” added. We denote their text embeddings by ξ_o and ξ_d , respectively. The resulting embedding is given by:

$$\xi^{ev} = \xi_o + (ev/EV_{\min})(\xi_d - \xi_o). \quad (3)$$

We train our LoRA with a masked version of the standard L_2 loss function computed only on the chrome ball pixels given by a mask \mathbf{M} :

$$\mathcal{L} = \mathbb{E}_{\mathbf{z}_o, t, \epsilon, ev} [\|\mathbf{M} \odot (\epsilon_\theta(\mathbf{z}_t^{ev}, t, \xi^{ev}) - \epsilon)\|_2^2], \quad (4)$$

where \mathbf{z}_t^{ev} is computed using Equation (1) from our training image with $EV=ev$. We choose to train a single LoRA as opposed to multiple LoRAs for individual EVs because it helps preserve the overall scene structure across exposures due to weight sharing. Refer to Appendix C for details.

LDR balls generation and HDR merging. We generate chrome balls with multiple EVs = $\{-5, -2.5, 0\}$, each using their own median ball computation (Section 3.2). While our LoRA can maintain the overall scene structure across exposures, some details do become altered. As a result, using

standard HDR merging algorithms can lead to ghosting artifacts when details in each LDR are not fully aligned. As our primary goal is to gather high-intensity light sources from underexposed images to construct a useful light map, we can merge the luminances while retaining the chroma from the normally exposed EV0 image to reduce ghosting.

In particular, we first identify overexposed regions in each LDR image with a simple threshold of 0.9, assuming the pixel range is between 0 and 1. Then, the luminance values in these regions are replaced by the exposure-corrected luminances from lower EV images. This luminance replacement is performed in pairs, starting from the lowest EV to the normal EV0 image, detailed in Algorithm 2 in Appendix A.

4. Experiments

Implementation details. We fine-tuned SDXL [45] for multi-exposure generation using a rank-4 LoRA [30]. We trained our LoRA on 1,412 HDR panoramas synthetically generated by Text2Light [10] for 2,500 steps with a learning rate of 10^{-5} and a batch size of 4. The process took about 5 hours on an NVIDIA RTX 3090Ti. During training, we sampled timestep $t \sim U(900, 999)$ as we found that the light information is determined at the early stage of the denoising process. When applying iterative inpainting (Section 3.2), we generated $N = 30$ chrome balls per each median computation iteration. We use UniPC [70] sampler with 30 sampling steps, a guidance scale of 5.0, and a LoRA scale of 0.75.

Datasets. We evaluated our approach on two standard benchmarks: Laval Indoor HDR [20] and Poly Haven [3]. The latter covers both indoor and outdoor settings.

Evaluation metrics. Following previous work [59, 67], we used three scale-invariant metrics: scale-invariant Root Mean Square Error (si-RMSE) [24], Angular Error [34], and normalized RMSE. The normalization for the last metric is done by mapping the 0.1st and 99.9th percentiles to 0 and 1, following [40]. We chose these metrics instead of standard RMSE because each benchmark dataset has its own specific range and statistics of light intensity, but our method was not trained on any of them.

4.1. Evaluation on benchmark datasets

We adopt two different evaluation protocols used in the literature: from each input LDR image, we generate an HDR panorama of size 128×256 pixels and use it to render (1) three spheres with different materials (gray-diffuse, silver-matte, and silver-mirror spheres) [20, 21, 59] or (2) an array of diffuse spheres [12, 62]. Then, we computed the evaluation metrics on these renderings. Many studies do not publish their source code and use only one of the protocols, resulting in missing baselines’ scores in some experiments.

Evaluation on three spheres. We compared our method to StyleLight [59] on (1) 289 panoramas from the Laval



Figure 7. Qualitative results on benchmark datasets. For each input image, we show the rendered chrome ball (1st row) and the corresponding environment map (2st row) from each method. (I: iterative, LR: LoRA).

Indoor dataset and (2) 500 panoramas from Poly Haven dataset. It is important to note that StyleLight was trained on the Laval *Indoor* dataset; its scores on Poly Haven are provided solely as a reference to demonstrate how existing methods perform in out-of-distribution scenarios. Following StyleLight’s protocol, we created one input image from each panorama by cropping it to a size of 192×256 with a vertical FOV of 60° and then applying tone-mapping, setting the 99th percentile to 0.9 and using $\gamma = 2.4$. In only our pipeline, we upscale the image while keeping the aspect ratio for SDXL.

Table 1 shows that our method outperforms StyleLight in terms of Angular Error and Normalized RMSE on Laval indoor dataset, with significantly lower Angular Error: 49.5% (diffuse), 27.8% (matte), and 12.4% (mirror). Our method is also effective in Poly Haven outdoor scenes, while Style-

Dataset	Method	Scale-invariant RMSE ↓			Angular Error ↓			Normalized RMSE ↓		
		Diffuse	Matte	Mirror	Diffuse	Matte	Mirror	Diffuse	Matte	Mirror
Laval Indoor [20]	StyleLight (reported by the paper)	0.11	0.29	0.55	2.41	2.96	4.30	-	-	-
	StyleLight (reproduced using official code)	0.13	0.31	0.55	4.24	4.74	6.78	0.23	0.40	0.51
	SDXL [†]	0.20	0.45	0.72	5.87	6.20	8.28	0.32	0.47	0.51
	SDXL [†] + iterative (ours)	0.15	0.39	0.67	3.58	4.55	7.05	0.25	0.41	0.47
	SDXL [†] + LoRA (ours)	0.15	0.38	0.65	3.47	3.86	6.15	0.25	0.40	0.45
Poly Haven [3]	SDXL [†] + iterative + LoRA (ours)	0.14	0.33	0.60	2.14	3.42	5.94	0.20	0.36	0.43
	StyleLight (reproduced using official code)	0.17	0.44	0.64	3.53	4.44	7.12	0.23	0.41	0.49
	SDXL [†]	0.22	0.59	0.80	2.98	4.25	5.74	0.30	0.50	0.54
	SDXL [†] + iterative (ours)	0.16	0.50	0.73	2.57	4.18	5.31	0.22	0.44	0.50
	SDXL [†] + LoRA (ours)	0.16	0.50	0.72	2.35	3.56	4.43	0.24	0.43	0.47
	SDXL [†] + iterative + LoRA (ours)	0.14	0.45	0.66	2.14	3.60	4.29	0.20	0.39	0.43

Table 1. Comparison using the three-sphere evaluation protocol between StyleLight [59], simple inpainting with SDXL [45] and depth-conditioned ControlNet [69] (“SDXL[†]” in the table), and ablated versions of our method. The best and second-best are color coded.

Method	si-RMSE ↓	Angular Error ↓
EverLight [12]	0.091	6.36
StyleLight [59]	0.123	7.09
Weber et al. [62]	0.081	4.13
EMLight [67]	0.099	3.99
Ours	0.090	5.25

Table 2. Scores on indoor array-of-spheres protocol (Section 4.1)

Sphere	Method	si-RMSE ↓	Angular Error ↓	Normalized RMSE ↓
Diffuse	StyleLight	0.143	3.741	0.236
	Ours	0.135	2.337	0.219
Matte	StyleLight	0.347	4.492	0.429
	Ours	0.359	3.483	0.369
Mirror	StyleLight	0.606	7.655	0.544
	Ours	0.644	5.988	0.438

Table 3. Scores on the random-camera protocol (Section 4.2).

Dataset	Method	RMSE ↓	si-RMSE ↓	Angular Error ↓
Laval Indoor [20]	StyleLight	0.246	0.271	5.814
	Ours	0.187	0.303	4.412
Poly Haven [3]	StyleLight	0.241	0.324	6.291
	Ours	0.179	0.275	4.567

Table 4. Scores on LDR environment maps (Section 4.3).

Light’s performance drops with a large 39.7% gap in Angular Error for mirror spheres. Qualitative results are in Figure 7 and Appendix E.1. Note that we used StyleLight’s official code to produce these scores; however, discrepancies exist with those reported in the paper. (See Appendix J for details and our discussion with StyleLight’s authors on this issue).

Evaluation on array of spheres. We compared our approach with StyleLight, Everlight [12], EMLight [67], and Weber et al. [62]. We used 224 panoramas (the same ones used to

evaluate Everlight) from the Laval Indoor dataset. For each panorama, we generated 10 input LDR images by centering the panorama at certain azimuthal angles and cropping it to 50° FOV, following Weber et al. [62]. As a result, the metrics were computed from 2240 input-output pairs. In Table 2, our method ranks after Weber et al. and EMLight; however, it outperforms Everlight and StyleLight, despite not being explicitly trained on the dataset.

4.2. Evaluation on unknown camera parameters

Evaluation protocols in the last section crop HDR panoramas at fixed camera angles and FOVs. In real-world settings, however, we often do not know the camera parameters of a photograph. This evaluation considers more challenging scenarios that reflect this situation better. In particular, to generate an input LDR image, we randomly sample the FOV from the interval [30°, 150°], the elevation from [−45°, 45°], and the azimuth from all 360°. We then crop an HDR panorama accordingly. We generate one LDR image from 289 HDR panoramas of the Laval Indoor dataset and compare our method with Stylelight using the generated input-output pairs and the three-sphere protocol. Table 3 shows that our method outperforms StyleLight in Angular Error and Normalized RMSE and remains competitive in si-RMSE.

4.3. Evaluation on LDR panoramas

We demonstrate that our method can generate more plausible panoramas than SOTA approaches. In this evaluation, we generated one LDR input image from each panorama in the test dataset as in the three-sphere protocol. However, we compared output LDR panoramas *directly* to the ground truth. Again, since each dataset uses its own brightness scale unknown to our method, comparison was done in the LDR image domain, where the scale is explicitly defined. Specifically, we compared our panoramas to those from StyleLight at a resolution of 256 × 512 after performing tone-mapping as described in [59]. Table 4 shows that our method out-

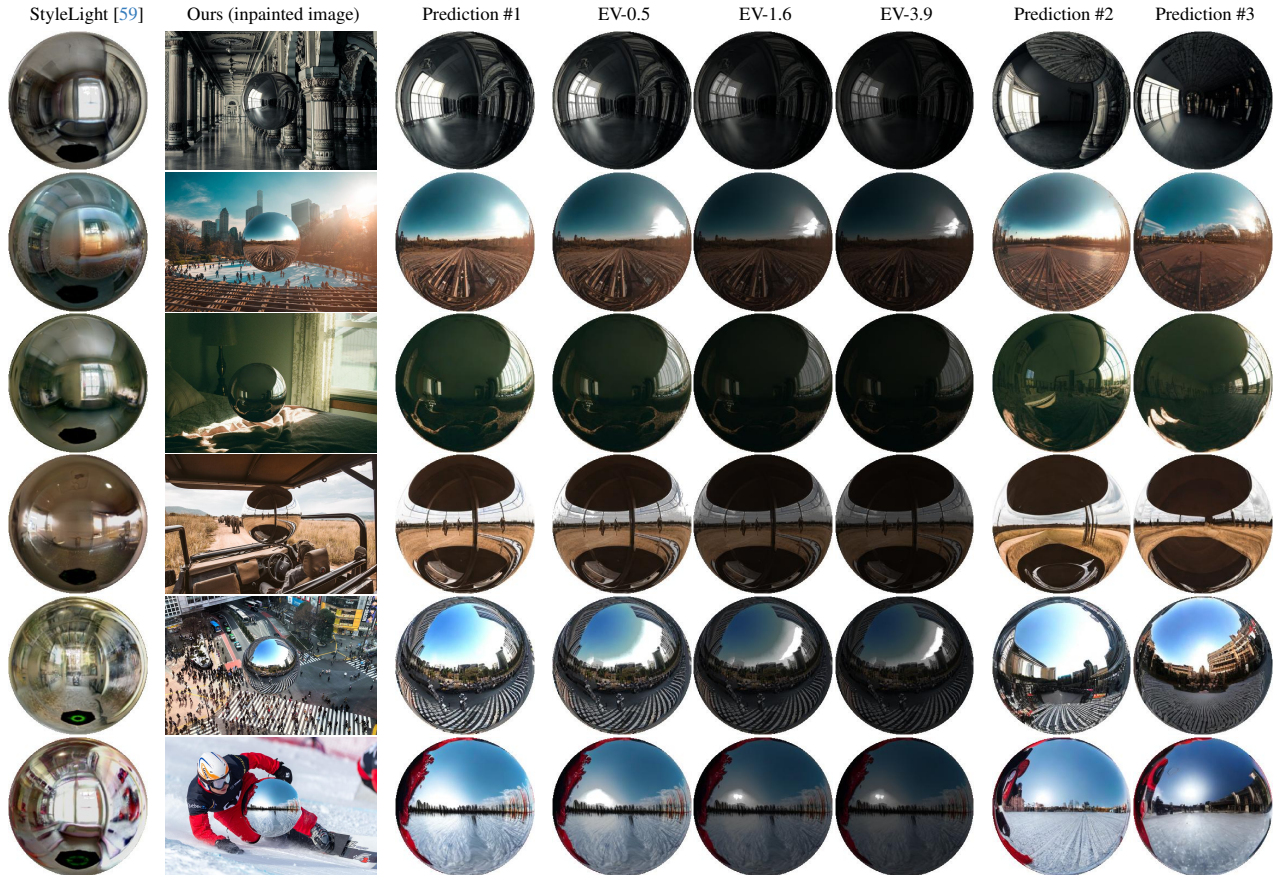


Figure 8. Qualitative results for in-the-wild scenes. We show chrome balls generated from our pipeline along with their HDR outputs, rendered at different EVs in columns 3 to 6. Additional plausible chrome balls are depicted in columns 7 and 8.

performs StyleLight with respect to nearly all metrics and datasets, suggesting that our method can leverage the strong generative prior in pre-trained diffusion models.

4.4. Qualitative results for in-the-wild scenes

We present additional qualitative results on diverse in-the-wild scenes available on Unsplash (www.unsplash.com) and other websites under CC4.0 license in Figure 8 and Appendix E.2. Compared to other existing techniques, our method can produce chrome balls that “reflect” what is in the input image: the car’s ceiling, the zebra crossing, and the red garment of the snowboarder, as well as reveal overexposed details, such as window frames and the sun (see also Figure 1).

4.5. Ablation studies

We perform an ablation study on our iterative inpainting and LoRA using Laval Indoor and Poly Haven datasets. Table 1 shows that our full method surpasses all ablated versions on all metrics except Angular Error on matte balls in Poly Haven. Studies on the size and number of balls, as well as the trade-off between running time and quality are in Appendix B. Studies on LoRA scale and timesteps are in Appendix C.

4.6. Limitations

Given the absence of focal length or FOV information, we assume orthographic projection when converting a chrome ball to an environment map, which may not reflect the projection model rendered by the diffusion model. Our chrome balls occasionally fail to reflect surrounding environments in overhead or bird’s eye-view images, shown in Appendix I. Our method is currently slow with iterative inpainting and diffusion sampling, but utilizing sampling-efficient diffusion models [36, 38, 39] can directly improve its speed.

5. Conclusion

This paper presents a novel HDR light estimation approach by inpainting a chrome ball into the scene using a pretrained LDR diffusion model. To consistently render high-quality chrome balls, we propose an iterative algorithm to locate a suitable initial noise neighborhood and apply continuous LoRA fine-tuning for exposure bracketing and generating HDR chrome balls. Our method performs competitively with the state of the art in both indoor and outdoor settings and marks the first work that achieves good generalization to in-the-wild images.

References

- [1] Dall-e 2. <https://openai.com/dall-e-2>, 2022. Accessed: Nov 16, 2023. [3](#), [15](#), [16](#), [21](#), [22](#)
- [2] Adobe firefly. <https://www.adobe.com/sensei/generative-ai/firefly.html>, 2023. Accessed: Nov 5, 2023. [3](#), [15](#), [16](#), [21](#), [22](#)
- [3] Poly haven. <https://polyhaven.com/>, 2023. Accessed: Nov 8, 2023. [2](#), [6](#), [7](#), [13](#), [15](#), [16](#)
- [4] Naofumi Akimoto, Seito Kasai, Masaki Hayashi, and Yoshimitsu Aoki. 360-degree image completion by two-stage conditional gans. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 4704–4708. IEEE, 2019. [3](#)
- [5] Naofumi Akimoto, Yuhi Matsuo, and Yoshimitsu Aoki. Diverse plausible 360-degree image outpainting for efficient 3dcg background creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11441–11450, 2022. [3](#)
- [6] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18208–18218, 2022. [2](#), [3](#), [15](#), [16](#), [21](#), [22](#)
- [7] Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *ACM Trans. Graph.*, 42(4), 2023. [2](#), [3](#), [15](#), [16](#), [21](#), [22](#)
- [8] James F. Blinn and Martin E. Newell. Texture and reflection in computer generated images. *Commun. ACM*, 19(10):542–547, 1976. [1](#)
- [9] Dan A Calian, Jean-François Lalonde, Paulo Gotardo, Tomas Simon, Iain Matthews, and Kenny Mitchell. From faces to outdoor light probes. In *Computer Graphics Forum*, pages 51–61. Wiley Online Library, 2018. [2](#)
- [10] Zhaoxi Chen, Guangcong Wang, and Ziwei Liu. Text2light: Zero-shot text-driven hdr panorama generation. *ACM Transactions on Graphics (TOG)*, 41(6):1–16, 2022. [5](#), [6](#), [13](#)
- [11] Mohammad Reza Karimi Dastjerdi, Yannick Hold-Geoffroy, Jonathan Eisenmann, Siavash Khodadadeh, and Jean-François Lalonde. Guided co-modulated gan for 360° field of view extrapolation. In *2022 International Conference on 3D Vision (3DV)*, pages 475–485. IEEE, 2022. [3](#)
- [12] Mohammad Reza Karimi Dastjerdi, Jonathan Eisenmann, Yannick Hold-Geoffroy, and Jean-François Lalonde. Everlight: Indoor-outdoor editable hdr lighting estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7420–7429, 2023. [2](#), [3](#), [6](#), [7](#)
- [13] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, page 189–198, New York, NY, USA, 1998. Association for Computing Machinery. [2](#)
- [14] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Anirudha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects, 2023. [17](#)
- [15] Paul Debevec. Reflection Mapping History from Gene Miller. <https://www.pauldebevec.com/ReflectionMapping/miller.html>. Accessed: 2023-11-17. [2](#)
- [16] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021. [3](#)
- [17] Hugging Face. Image-to-image. [4](#)
- [18] Blender Foundation. Home of the blender project - free and open 3d creation software. [5](#)
- [19] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022. [3](#)
- [20] Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. Learning to predict indoor illumination from a single image. *ACM Trans. Graph.*, 36(6), 2017. [2](#), [6](#), [7](#), [13](#)
- [21] Marc-André Gardner, Yannick Hold-Geoffroy, Kalyan Sunkavalli, Christian Gagné, and Jean-François Lalonde. Deep parametric indoor lighting estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7175–7183, 2019. [2](#), [6](#)
- [22] Mathieu Garon, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, and Jean-François Lalonde. Fast spatially-varying indoor lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6908–6917, 2019. [2](#)
- [23] Stamatios Georgoulis, Konstantinos Rematas, Tobias Ritschel, Mario Fritz, Tinne Tuytelaars, and Luc Van Gool. What is around the camera? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5170–5178, 2017. [2](#)
- [24] Roger Grosse, Micah K. Johnson, Edward H. Adelson, and William T. Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2335–2342, 2009. [6](#)
- [25] Ligong Han, Yinxiao Li, Han Zhang, Peyman Milanfar, Dimitris Metaxas, and Feng Yang. Svdif: Compact parameter space for diffusion fine-tuning. *arXiv preprint arXiv:2303.11305*, 2023. [3](#)
- [26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 6840–6851, 2020. [4](#), [16](#), [17](#)
- [27] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Sunil Hadap, Emiliano Gambaretto, and Jean-François Lalonde. Deep outdoor illumination estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7312–7321, 2017. [2](#)
- [28] Yannick Hold-Geoffroy, Akshaya Athawale, and Jean-François Lalonde. Deep sky modeling for single image outdoor lighting estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6927–6935, 2019. [2](#)

- [29] Lukas Hosek and Alexander Wilkie. An analytic model for full spectral sky-dome radiance. *ACM Transactions on Graphics (TOG)*, 31(4):1–9, 2012. 2
- [30] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 2, 3, 4, 6
- [31] Kevin Karsch, Varsha Hedau, David Forsyth, and Derek Hoiem. Rendering synthetic objects into legacy photographs. *ACM Transactions on graphics (TOG)*, 30(6):1–12, 2011. 2
- [32] Kevin Karsch, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, Hailin Jin, Rafael Fonte, Michael Sittig, and David Forsyth. Automatic scene inference for 3d object compositing. *ACM Transactions on Graphics (TOG)*, 33(3):1–15, 2014.
- [33] Jean-François Lalonde, Alexei A Efros, and Srinivasa G Narasimhan. Estimating the natural illumination conditions from a single outdoor image. *International Journal of Computer Vision*, 98:123–145, 2012. 2
- [34] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. Deeplight: Learning illumination for unconstrained mobile mixed reality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019. 2, 3, 6
- [35] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022. 16
- [36] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. InstafLOW: One step is enough for high-quality diffusion-based text-to-image generation. *arXiv preprint arXiv:2309.06380*, 2023. 8
- [37] Stephen Lombardi and Ko Nishino. Reflectance and illumination recovery in the wild. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):129–141, 2015. 2
- [38] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference, 2023. 8
- [39] Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. Lcm-lora: A universal stable-diffusion acceleration module. *arXiv preprint arXiv:2311.05556*, 2023. 8
- [40] Demetris Marnerides, Thomas Bashford-Rogers, Jonathan Hatchett, and Kurt Debattista. Expandnet: A deep convolutional neural network for high dynamic range expansion from low dynamic range content, 2019. 6
- [41] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. 3, 4, 12
- [42] Rang MH Nguyen and Michael S Brown. Why you should forget luminance conversion and do something better. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6750–6758, 2017. 12
- [43] Ko Nishino and Shree K Nayar. Eyes for relighting. *ACM Transactions on Graphics (TOG)*, 23(3):704–711, 2004. 2
- [44] Jeong Joon Park, Aleksander Holynski, and Steven M Seitz. Seeing the world in a bag of chips. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1417–1427, 2020. 2
- [45] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023. 2, 3, 4, 6, 7, 15, 16, 17, 21, 22, 33
- [46] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10619–10629, 2022. 3
- [47] Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. *arXiv preprint arXiv:2306.07280*, 2023. 3
- [48] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 3
- [49] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ArXiv preprint*, 2021. 3
- [50] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 3, 4
- [51] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022. 3, 14
- [52] Gowri Somanath and Daniel Kurz. Hdr environment map estimation for real-time augmented reality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11298–11306, 2021. 2
- [53] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020. 4, 16, 17
- [54] Luming Tang, Nataniel Ruiz, Chu Qinghao, Yuanzhen Li, Aleksander Holynski, David E Jacobs, Bharath Hariharan, Yael Pritch, Neal Wadhwa, Kfir Aberman, and Michael Rubinstein. Realfill: Reference-driven generation for authentic image completion. *arXiv preprint arXiv:2309.16668*, 2023. 3
- [55] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. 12, 14
- [56] Andrey Voynov, Qinghao Chu, Daniel Cohen-Or, and Kfir Aberman. P+: Extended textual conditioning in text-to-image generation. 2023. 3
- [57] Bram Wallace, Akash Gokul, Stefano Ermon, and Nikhil Naik. End-to-end diffusion latent optimization improves classifier guidance. *arXiv preprint arXiv:2303.13703*, 2023. 3
- [58] Bram Wallace, Akash Gokul, and Nikhil Naik. Edict: Exact diffusion inversion via coupled transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22532–22541, 2023. 3

- [59] Guangcong Wang, Yinuo Yang, Chen Change Loy, and Ziwei Liu. Stylelight: Hdr panorama generation for lighting estimation and editing. In *European Conference on Computer Vision*, pages 477–492. Springer, 2022. [2](#), [3](#), [5](#), [6](#), [7](#), [8](#), [12](#), [16](#), [23](#), [24](#), [25](#), [26](#), [27](#), [28](#), [29](#), [30](#)
- [60] Jionghao Wang, Ziyu Chen, Jun Ling, Rong Xie, and Li Song. 360-degree panorama generation from few unregistered nfov images. *arXiv preprint arXiv:2308.14686*, 2023. [3](#)
- [61] Henrique Weber, Donald Prévost, and Jean-François Lalonde. Learning to estimate indoor lighting from 3d objects. In *2018 International Conference on 3D Vision (3DV)*, pages 199–207. IEEE, 2018. [2](#)
- [62] Henrique Weber, Mathieu Garon, and Jean-François Lalonde. Editable indoor lighting estimation. In *European Conference on Computer Vision*, pages 677–692. Springer, 2022. [2](#), [6](#), [7](#), [16](#)
- [63] Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. Paint by example: Exemplar-based image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18381–18391, 2023. [2](#), [3](#), [15](#), [16](#), [21](#), [22](#)
- [64] Hu Ye, Jun Zhang, Sibao Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. 2023. [2](#), [3](#), [15](#), [16](#), [21](#), [22](#)
- [65] Renjiao Yi, Chenyang Zhu, Ping Tan, and Stephen Lin. Faces as lighting probes via unsupervised deep highlight extraction. In *Proceedings of the European Conference on computer vision (ECCV)*, pages 317–333, 2018. [2](#)
- [66] Hong-Xing Yu, Samir Agarwala, Charles Herrmann, Richard Szeliski, Noah Snavely, Jiajun Wu, and Deqing Sun. Accidental light probes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12521–12530, 2023. [2](#)
- [67] Fangneng Zhan, Changgong Zhang, Yingchen Yu, Yuan Chang, Shijian Lu, Feiying Ma, and Xuansong Xie. Emlight: Lighting estimation via spherical distribution approximation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3287–3295, 2021. [2](#), [6](#), [7](#)
- [68] Fangneng Zhan, Yingchen Yu, Changgong Zhang, Rongliang Wu, Wenbo Hu, Shijian Lu, Feiying Ma, Xuansong Xie, and Ling Shao. Gmlight: Lighting estimation via geometric distribution approximation. *IEEE Transactions on Image Processing*, 31:2268–2278, 2022. [2](#)
- [69] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. [2](#), [3](#), [4](#), [7](#), [16](#)
- [70] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *NeurIPS*, 2023. [4](#), [6](#), [14](#), [16](#)

A. Implementation Details

A.1. Inpainting algorithm

The pseudocode of the iterative inpainting algorithm described in Section 3.2 is given in Algorithm 1. Our implementation uses $\gamma = 0.8$, $k = 2$, and $N = 30$. The algorithm repeatedly invokes the INPAINT function, which stands for an inpainting algorithm based on SDEdit [41] as implemented in the Diffusers library [55]. For completeness, we include its pseudocode in Algorithm 0. This algorithm requires no modification to the diffusion model and resembles standard diffusion sampling except the ‘‘imputing’’ step in Line 16.

Algorithm 0: Inpainting using Diffusion Model

```

1: function UPDATE( $\mathbf{z}, t, \epsilon$ )
2:   return  $\sqrt{\alpha_{t-1}} \left( \frac{\mathbf{z} - \sqrt{1-\alpha_t}\epsilon}{\sqrt{\alpha}} \right) + \sqrt{1-\alpha_{t-1}}\epsilon$ 
3: end function
4:
5: // input: latent code of input image  $\mathbf{z}_I$ ,
6: // initial latent code  $\mathbf{z}$ , // inpainting mask  $M$ 
7: // conditioning signal (e.g. text, depth map)  $\mathbf{C}$ 
8: // timestep to start denoising (denoising-start).
9: // output: Input image with an inpainted chrome ball.
10: function INPAINT( $\mathbf{z}_I, \mathbf{z}, \mathbf{M}, \mathbf{C}, \text{denoising-start} = T$ )
11:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
12:   for  $i \in \{\text{denoising-start}, \dots, 1\}$  do
13:      $\epsilon \leftarrow \epsilon_\theta(\mathbf{z}, t, \mathbf{C})$ 
14:      $\mathbf{z} \leftarrow \text{UPDATE}(\mathbf{z}, t, \epsilon)$ 
15:      $\mathbf{z}'_I \leftarrow \sqrt{\alpha_t}\mathbf{z}_I + \sqrt{1-\alpha_t}\epsilon$ 
16:      $\mathbf{z} \leftarrow (1 - \mathbf{M}) \odot \mathbf{z}'_I + \mathbf{M} \odot \mathbf{z}$ 
17:   end for
18:    $\epsilon \leftarrow \epsilon_\theta(\mathbf{z}, 0, \mathbf{C})$ 
19:    $\mathbf{z} \leftarrow \text{UPDATE}(\mathbf{z}, 0, \epsilon)$ 
20:   return DECODE( $\mathbf{z}$ )
21: end function

```

A.2. HDR merging algorithm

Algorithm 2 merges a bracket of LDR images to create an HDR image. As mentioned in the main paper, we merge in the luminance space to avoid ghosting artifacts. Our luminance conversion assumes sRGB [42] and gamma value of 2.4, following [59].

B. Ablation on Iterative Inpainting Algorithm

B.1. Inpainting iterations

Figure 10 presents additional results demonstrating how our iterative inpainting algorithm improves the consistency and quality of the generated chrome balls after one iteration. Figure 11 presents results with more iterations. Note that the

Algorithm 1: Iterative Inpainting Algorithm

```

1: function INPAINTBALL( $\mathbf{I}, \mathbf{M}, \mathbf{C}, \gamma, T = 1000$ )
2:    $\mathbf{z} \leftarrow \text{ENCODE}(\mathbf{I})$ 
3:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
4:    $\mathbf{z}_{\gamma T} \leftarrow \sqrt{\alpha_{\gamma T}}\mathbf{z} + \sqrt{1-\alpha_{\gamma T}}\epsilon$ 
5:   return
   INPAINT( $\mathbf{z}, \mathbf{z}_{\gamma T}, \mathbf{M}, \mathbf{C}, \text{denoising-start} = \gamma T$ )
6: end function
7:
8: // input: Input image  $\mathbf{I}$ , binary inpainting mask  $\mathbf{M}$ ,
9: // Conditioning signal (e.g. text, depth map)  $\mathbf{C}$ 
10: // denoising strength  $\gamma$ , number of balls for median  $N$ ,
11: // number of median iterations  $k$ .
12: // output: Input image with an inpainted chrome ball.
13: function ITERATIVEINPAINT( $\mathbf{I}, \mathbf{M}, \mathbf{C}, \gamma, k, N$ )
14:   for  $i \in \{1, \dots, k\}$  do
15:     for  $j \in \{1, \dots, N\}$  do
16:        $\gamma' \leftarrow \gamma$  if  $i > 1$  else 1.0
17:        $\mathbf{B}_j \leftarrow \text{INPAINTBALL}(\mathbf{I}, \mathbf{M}, \mathbf{C}, \gamma')$ 
18:     end for
19:      $\mathbf{B} \leftarrow \text{PIXELWISEMEDIAN}(\mathbf{B}_1, \dots, \mathbf{B}_N)$ 
20:      $\mathbf{I} \leftarrow (1 - \mathbf{M}) \odot \mathbf{I} + \mathbf{M} \odot \mathbf{B}$ 
21:   end for
22:   return INPAINTBALL( $\mathbf{I}, \mathbf{M}, \gamma$ )
23: end function

```

Algorithm 2: HDR Merging Algorithm

```

1: function LUMINANCE( $\mathbf{I}, ev, \gamma = 2.4$ )
2:   return  $\mathbf{I}^\gamma \cdot [0.21267, 0.71516, 0.07217]^\top (2^{-ev})$ 
3: end function
4:
5: // input: LDR images and a list of exposure values in
6: // descending order, where  $ev_0 = 0$ . E.g.,  $[0, -2.5, -5]$ 
7: // output: A linearized HDR image.
8: function MERGELDRS( $\mathbf{I}_0, \dots, \mathbf{I}_{n-1}, ev_0, \dots, ev_{n-1}$ )
9:    $\mathbf{L} \leftarrow \text{LUMINANCE}(\mathbf{I}_{n-1}, ev_{n-1})$ 
10:  for  $i \in \{n-2, n-1, \dots, 0\}$  do
11:     $\mathbf{L}_i \leftarrow \text{LUMINANCE}(\mathbf{I}_i, ev_i)$ 
12:     $\mathbf{M} \leftarrow \text{CLIP}\left(\frac{2^{ev_i}\mathbf{L}_i - 0.9}{0.1}, 0, 1\right) \odot \mathbb{1}(\mathbf{L} > \mathbf{L}_i)$ 
13:     $\mathbf{L} \leftarrow (1 - \mathbf{M}) \odot \mathbf{L}_i + \mathbf{M} \odot \mathbf{L}$ 
14:  end for
15:  return  $\mathbf{I}_0^\gamma \odot \left( \frac{\mathbf{L}}{\mathbf{L}_0} \right)$ 
16: end function

```

experiments in our main paper were limited to two iterations due to our resource constraints.

B.2. Trade-off between running time and quality

We experimented with various numbers of balls N and iterations k in the iterative inpainting algorithm on the validation set comprising 100 scenes from the Laval Indoor dataset

#Iterations (k)	#Balls (N)	Time (mins)	si-RMSE ↓	Angular Error ↓	Normalized RMSE ↓
1	5	2.5	0.631	5.367	0.416
	15	7.5	0.615	5.229	0.410
	30	15	0.609	5.210	0.405
2	5	5	0.634	5.393	0.417
	15	15	0.615	5.263	0.409
	30	30	0.607	5.248	0.403

Table 5. Ablation study on the number of iterations k and balls N . We report running time and quality metrics on 200 mirror balls in the validation set. See Figure 9 for qualitative results.

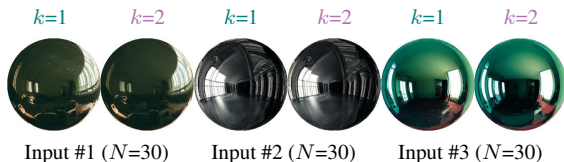


Figure 9. Qualitative results for the two configurations in Table 5, marked with teal and purple colors. We can halve the running time with minimal quality degradation by decreasing the number of iterations k to one.

[20] and 100 scenes from the Poly Haven dataset [3]. As shown in Table 5, increasing either of them generally leads to more accurate results. Notably, while two iterations of iterative inpainting ($k = 2$) deliver the best score, reducing the number of iterations to one ($k = 1$) halves the running time with minimal quality degradation as shown in Figure 9. Note that we opted for $N = 30$ and $k = 2$, which takes about 30 minutes per image on an RTX 3090 Ti GPU, because of resource constraints.

B.3. Inpainting ball size

We investigated the effects of ball diameter (i.e., white circle) on the depth maps used by ControlNet. Specifically, we analyze and compare the efficacy of various ball sizes, ranging from 128 to 512 pixels in diameter, as illustrated in Figure 12. The result shows that increasing the ball size from 256 to 384 or 512 still results in realistic balls, but they do not reflect the environment as convincingly. This is likely because the original input content seen by the model is reduced. On the other hand, smaller balls can capture the lighting well but are less detailed and not as useful.

C. Ablation on LoRA Training

C.1. Additional details on training set generation

To generate training panoramas from text prompts as mentioned in Section 3.3, we use Text2Light [10], using prompts from its official GitHub repository[‡] and additional

[‡]<https://github.com/FrozenBurning/Text2Light>

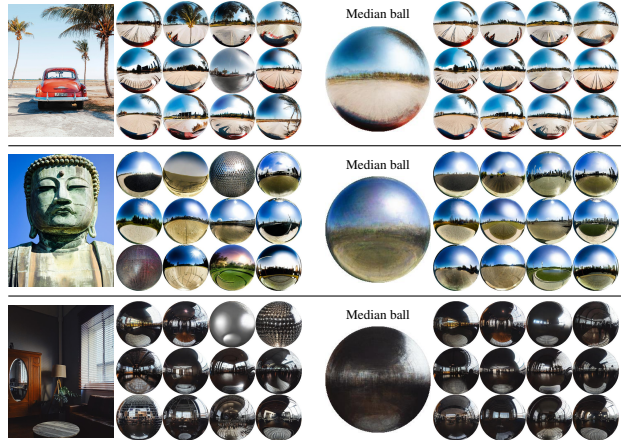


Figure 10. Chrome balls before (left) and after (right) one iteration of our iterative inpainting algorithm. Notice how poor chrome balls are fixed and the light estimation becomes more consistent.

prompts generated by Chat-GPT 3.5[§] from short instructions and examples. To eliminate near-duplicate samples, we use the perceptual hashing algorithm implemented in the imagededup package[¶]. This process yielded a dataset of 1,412 unique HDR panoramas at resolution of 2048×4196 pixels. We used orthographic projection and a 60° field of view.

C.2. Range of timesteps for LoRA training

For LoRA training, we sampled from timesteps 900-999 as we observed that the overall lighting information is determined earlier in the sampling process (see Figure 13). This choice helped speed up training. In Table 6, we compare this choice to training from 0-999 and 500-999 given the same number of training iterations and report scores on the same validation set of 200 scenes as in Appendix B.2. Our choice of 900-999 yielded the best performance across all three metrics.

C.3. LoRA scale

We conducted an experiment to assess the effect of using different LoRA scales. Here, the LoRA scale refers to the α value in the weight update equation: $\mathbf{W}' = \mathbf{W} + \alpha \Delta \mathbf{W}$, where \mathbf{W}' is the new weight for inference, \mathbf{W} is the original weight of SDXL, $\Delta \mathbf{W}$ is the weight update from LoRA. (See Section 3.1 for a brief background on LoRA.) In Table 7, we report scores computed on EV0 LDR chrome balls evaluated on scenes in Poly Haven, which were never part of Text2Light’s training set. We selected the LoRA scale of 0.75, which has the best si-RMSE and angular error scores, for our implementation.

[§]<https://chat.openai.com/>

[¶]<https://github.com/ideal0/imagededup>

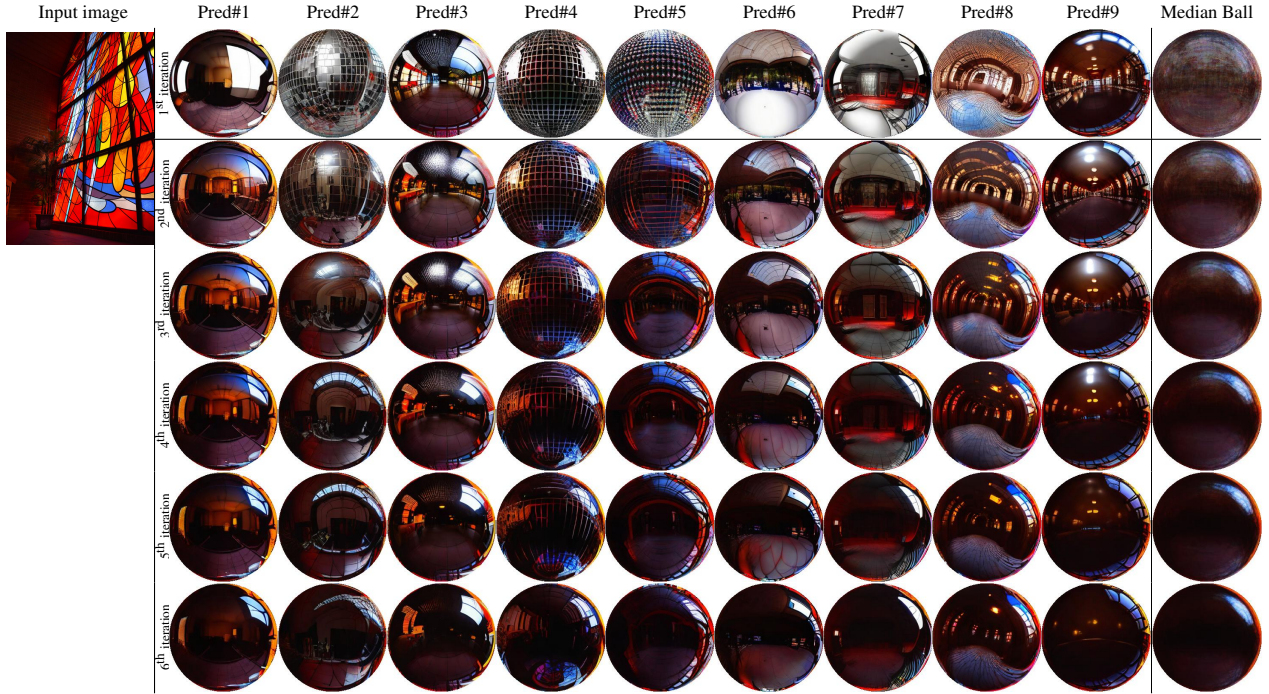


Figure 11. Repeatedly applying our iterative inpainting algorithm gradually produces chrome balls with better light estimation and fix degenerate balls such as Pred#2, Pred#4, and Pred#5.

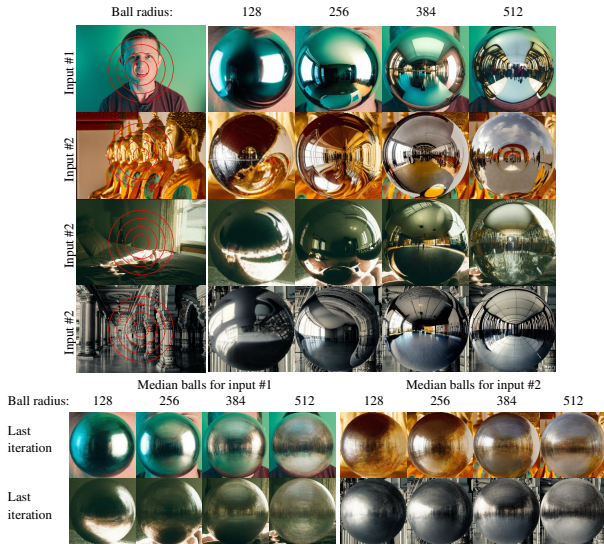


Figure 12. Results when varying the ball size (without LoRA).

C.4. Training a single continuous LoRA v.s. multiple LoRAs for exposure bracketing

As described in Section 3.3, we train a single *continuous* LoRA for multiple EVs by conditioning it on an interpolated text prompt embedding instead of training multiple LoRAs for individual EVs. This strategy helps preserve the overall scene structure across exposures due to weight sharing.

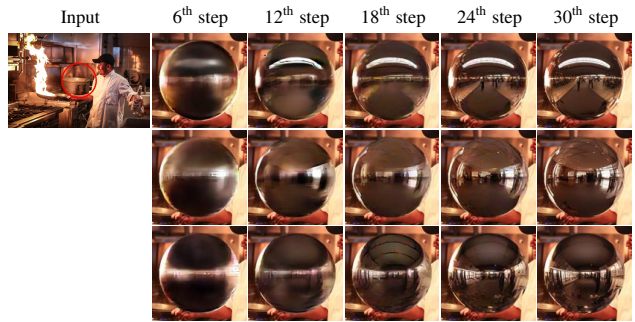


Figure 13. The overall lighting is determined at early sampling steps. Here, we visualize intermediate predictions at various steps during 30-step sampling with UniPC [70]. These intermediate predictions, or the predicted \mathbf{z}_0 , can be computed from \mathbf{z}_t at any timestep t using Equation 1. Each row corresponds to a different random seed.

To show this, we conducted an experiment comparing results from our LoRA and three separately trained LoRAs at EVs 0, -2.5, and -5.0. Following the commonly adopted training pipeline [51] implemented in the Diffusers library [55], our three LoRAs are trained with the prompt containing the ‘sks’ token: “a perfect sks mirrored reflective chrome ball sphere.” We use the same hyperparameters, random seeds, and HDR panoramas during training. We present results without our iterative algorithm to isolate the effect of LoRA in Figure 14. Note that we use a lora scale of 1.0 to apply



Figure 14. Our proposed continuous LoRA training (Cont. LoRA) yields chrome balls with higher detail consistency across different EVs than results obtained from using three separate LoRAs (3 LoRAs).

Sphere	Denoising step	si-RMSE ↓	Angular Error ↓	Normalized RMSE ↓
Diffuse	$x_0 : x_{999}$	0.194	3.322	0.292
	$x_{500} : x_{999}$	0.188	3.260	0.284
	$x_{900} : x_{999}$	0.156	2.956	0.246
Matte	$x_0 : x_{999}$	0.449	4.121	0.436
	$x_{500} : x_{999}$	0.452	4.008	0.438
	$x_{900} : x_{999}$	0.385	3.575	0.371
Mirror	$x_0 : x_{999}$	0.727	6.292	0.483
	$x_{500} : x_{999}$	0.730	6.277	0.479
	$x_{900} : x_{999}$	0.656	5.464	0.431

Table 6. Ablation study on sampled timesteps for LoRA training.

LoRA scale	RMSE ↓	si-RMSE ↓	Angular Error ↓
0.00	0.232	0.327	6.189
0.25	0.220	0.307	6.287
0.50	0.211	0.303	6.180
0.75	0.204	0.300	6.109
1.00	0.199	0.303	6.267

Table 7. Ablation study on LoRA scales

the same weight residual matrices obtained from the training. Our LoRA produces chrome balls with better structure consistency, particularly at EV-5.0.

D. More Comparison with SOTA Inpainting Techniques

In Figure 2 in Section 2, we provide a qualitative comparison between our approach and existing SOTA diffusion-based inpainting methods: Blended Diffusion [6, 7], Paint-by-Example [63], IP-Adapter [64], DALL-E2 [1], Adobe Firefly [2], and SDXL [45]. In this section, we describe the experimental settings for these methods. Additionally, we investigate the behavior of each using different random seeds.

D.1. Experimental setups

Blended Diffusion, IP-Adapter, and SDXL shared the same text prompt: “a perfect mirrored reflective chrome ball sphere.”. We used negative prompt “matte, diffuse, flat, dull” when executing methods that can accept one: IP-Adapter and SDXL. We provided Paint-by-Example and IP-Adapter with reference chrome balls from five randomly selected HDR environment maps in Poly Haven dataset [3] as shown in Figure 15. We used the official OpenAI API ¹ for DALL-E2, and we used the Generative Fill feature in Photoshop for Adobe Firefly. We followed the default configurations in the methods’ official implementations as described in Table 8.

D.2. Behavior under different random seeds

We show inpainting results of our method and other baselines using different random seeds in Figure 21 and Figure

¹<https://platform.openai.com/docs/guides/images/edits-dall-e-2-only>

Method	Sampler	#step	cfg
Blended Diffusion	DDIM [26]	50	7.5
Paint-by-Example	PLMS [35]	50	5.0
IP-Adapter	UniPC [70]	30	5.0
SDXL	UniPC [70]	30	5.0

Table 8. Sampler, number of sampling steps, and classifier-guidance scale (cfg) used in different SOTA methods.



Figure 15. Chrome balls used as inputs for Paint-by-Example [63] and IP-Adapter [64]. We generate them from five randomly selected HDR environment maps from Poly Haven dataset [3].

22. What we observed in general was that Blend Diffusion [6, 7] produced distorted balls. Paint-by-Example [63] failed to reproduce mirrored chrome balls altogether. IP-Adapter [64] replicated textures and details of the example chrome balls, making it unsuitable for light estimation. DALL-E2 [1] often simply reconstructed most of the masked-out content. Adobe Firefly [2] had a similar problem with DALL-E2 [1], albeit more severe (see Figure 22). Moreover, none of these techniques precisely followed the inpainting mask. Our proposed method can address all these issues and consistently inpaint high-quality chrome balls.

E. Additional Qualitative Results

E.1. Benchmark datasets

This section provides qualitative results for the experiments in Section 4 in the main paper.

Evaluation on three spheres. We show spheres with three material types—mirror, matte, and diffuse—rendered using the inferred environment maps from the following methods:

1. The ground truth
2. StyleLight [59]
3. Stable Diffusion XL [45] with depth-conditioned ControlNet [69] (SDXL[†])
4. SDXL[†] with our LoRA (SDXL[†]+LR)
5. SDXL[†] with our iterative inpainting (SDXL[†]+I)
6. SDXL[†] with our LoRA and iterative inpainting (SDXL[†]+LR+I)

Qualitative results for the Laval indoor dataset are in Figure 26-25 and for Poly Haven in Figure 27-29. As discussed in the main paper, we start with SDXL[†] to which we add our LoRA (LR) and iterative inpainting (I) to obtain our proposed algorithm, SDXL[†]+LR+I. The methods designated

SDXL[†]+LR and SDXL[†]+I are ablated versions of our algorithm.

Evaluation on an array of spheres. We show renderings of an array of spheres on a plane using our estimated lighting, following the evaluation protocol from Weber et al. [62]. We display 24 random test images from a total of 2,240 test images of the Laval Indoor dataset in Figure 31. Because the scale of the HDR images our method generates and that of the test set are different, for visualization purposes, we scale each output image so that its 0.1st and 99.9th percentiles of pixel intensity match those of the ground truth. Note that this scaling does not affect the quantitative scores reported in the main paper since the metrics are already scale-invariant. The last row shows challenging test scenes featuring only plain, solid backgrounds without any shaded objects. Estimating lighting from such input images is highly ill-posed and multi-modal. As a result, visual assessment or evaluations using pixel-based metrics, as used in this protocol, may not be meaningful for such cases.

E.2. In-the-wild images

We present additional qualitative results for in-the-wild scenes in Figure 32. Our method produces high-quality chrome balls that harmonize well with diverse scenes and lighting environments, such as a dim hallway under red neon lighting, an underwater tunnel with blue-tinted sunlight, a close-up shot of food, an outdoor view by a coastline, and a bird’s-eye view from a tall building. Our method also works on non-realistic images, such as paintings or cartoons, where visual cues like shading and shadows are present (see Figure 33).

F. Stochastic vs Deterministic Sampling

In Section 3.2, we discuss the relationship between initial noise maps and semantic patterns of chrome balls. This mapping is deterministic only when using samplers derived from probability flow ODE, such as DDIM [53] and UniPC [70]. The “disco” noise map would less consistently produce “disco” balls if we adopted stochastic samplers such as DDPM [26], which introduce noise during the sampling process. This section investigates whether degeneration of chrome balls is caused by deterministic sampling and whether stochastic sampling can help mitigate this issue.

We conducted an experiment comparing results from DDIM and DDPM using different numbers of sampling steps. Our results suggest that neither of these schemes consistently reduces occurrence of bad chrome balls. Specifically, using stochastic samplers may occasionally produce better chrome balls than deterministic ones at sufficiently high numbers of sampling steps (see Figure 16). Unfortunately, they still yield “disco” balls when starting sampling with the “disco” noise map, as illustrated in Figure 17.

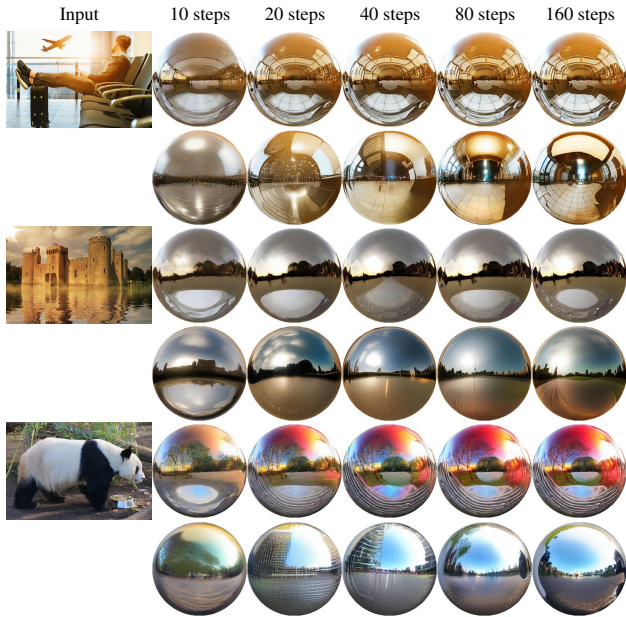


Figure 16. Comparison between chrome balls generated using DDIM [53] (1st row) and DDPM [26] (2nd row) with different sampling steps. DDPM can sometimes mitigate the occurrence of bad patterns originating from bad initial noise maps when using high sampling steps. Prompt: “a chrome ball”.

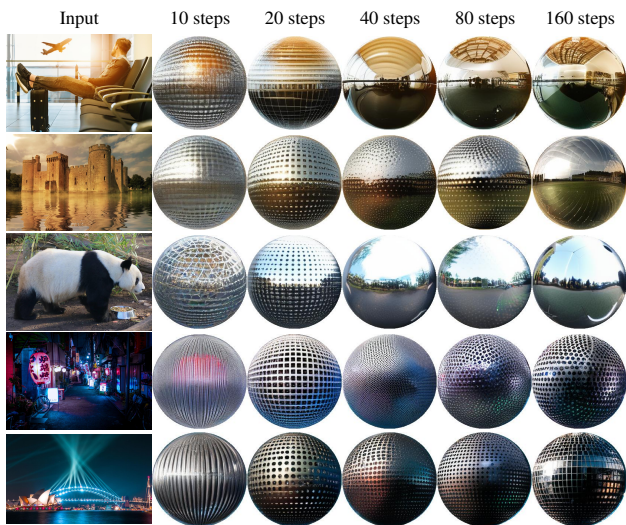


Figure 17. Chrome balls generated from the “disco” noise map using DDPM [26] with different sampling steps. Switching to DDPM instead of deterministic samplers still produces “disco” balls, even at high sampling steps. Prompt: “a chrome ball”.

G. Spatially-Varying Light Estimation

In this work, we inpaint a chrome ball in the input’s center to represent global lighting in the scene and do not model any spatially varying effects by assuming orthographic pro-

jection. Nonetheless, our preliminary study suggests that the output from our inpainting pipeline does change according to where the chrome ball is inpainted, as illustrated in Figure 18. This behavior can be leveraged for spatially-varying light estimation. To correctly infer spatially-varying, omnidirectional lighting, one needs to also infer the scene geometry, the depth of the inpainted chrome ball and camera parameters such as the focal length from the input image. These problems are interesting areas for future work.

H. Virtual Object Insertion

Virtual object insertion is a downstream application that requires light estimation. In Figure 19, we present qualitative results for two objects from Objaverse-XL [14], rendered with HDR environment maps obtained through our method.

I. Additional Failure Cases

We present additional failure cases in Figure 20. Our method occasionally fails to produce chrome balls that accurately reflect surrounding environments in overhead or bird’s-eye view images. For instance, the curvature of the horizon line in the scene with balloons is incorrect. While our method performs reasonably well for some non-realistic images like paintings, it struggles with images in drastically different styles, such as some cartoons and Japanese-style animations, which significantly differ from the training data of SDXL [45]. Switching to a fine-tuned model, such as AnimateXL **, to leverage a more specialized generative prior can improve its performance on specific image styles.

J. StyleLight’s Score Discrepancies

We used StyleLight’s official implementation to produce their scores in Table 1. However, there are discrepancies with their reported scores due to unknown implementations of their metrics. We discussed this with the authors on GitHub †† before the submission deadline, and they clarified that additional masking of black regions and rotation of panoramas were performed before evaluation. Despite implementing these additional steps, we still could not match the scores. The authors further suggested that we apply a consistent post-processing technique to all baselines for a fair comparison, which resulted in the scores we reported. To ensure transparency, we have made our evaluation code available at <https://github.com/DiffusionLight/DiffusionLight-evaluation>.

**<https://huggingface.co/Linaqruf/animate-xl>

††<https://www.kaggle.com/datasets/mylesoneill/tagged-anime-illustrations/data>

‡‡<https://github.com/Wangcong/StyleLight/issues/9>

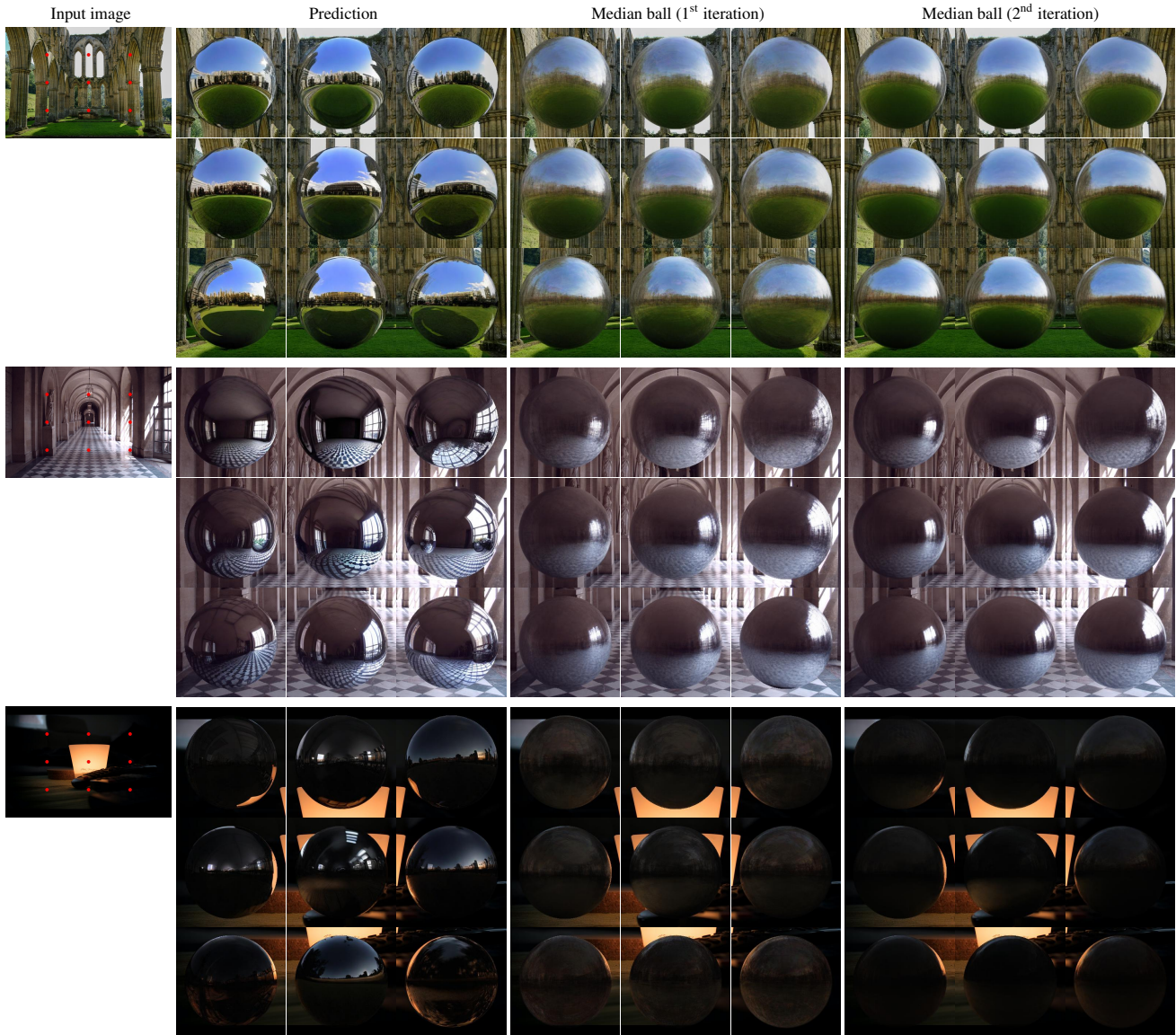


Figure 18. We show the spatially varying effects of painting a chrome ball at nine different locations, specified by red dots in the input images. For each input image, we present predictions from a random seed and median balls at the 1st and 2nd iterations. The effects can be seen in the changes of the curvature the horizon line, the size of the window, and the position of the light reflected from the lamp. These effects are more apparent in median balls as the number of iterations increases.



Figure 19. We synthetically render each 3D object into input images using our estimated lighting.



(a) Overhead and bird's-eye view images



(b) Images with significant style difference from natural photos



(c) Macro and close-up images



(d) High-key and low-key images with solid backgrounds

Figure 20. **Failure modes:** (a) Our method may produce unrealistic chrome balls in overhead and bird's-eye view images, leading to incorrect curvature in the horizon line. (b) The chrome balls may not harmonize well with inputs whose styles differ significantly from natural photos. (c) Macro and close-up images can lack sufficient shading cues, leading to less convincing estimates. (d) Images with empty or solid backgrounds often cause our method to hallucinate some details onto the balls, and the balls may appear too dark, especially on a white background. These images are from Unsplash (www.unsplash.com), Kaggle ^{††}, or other websites under the CC 4.0 license.

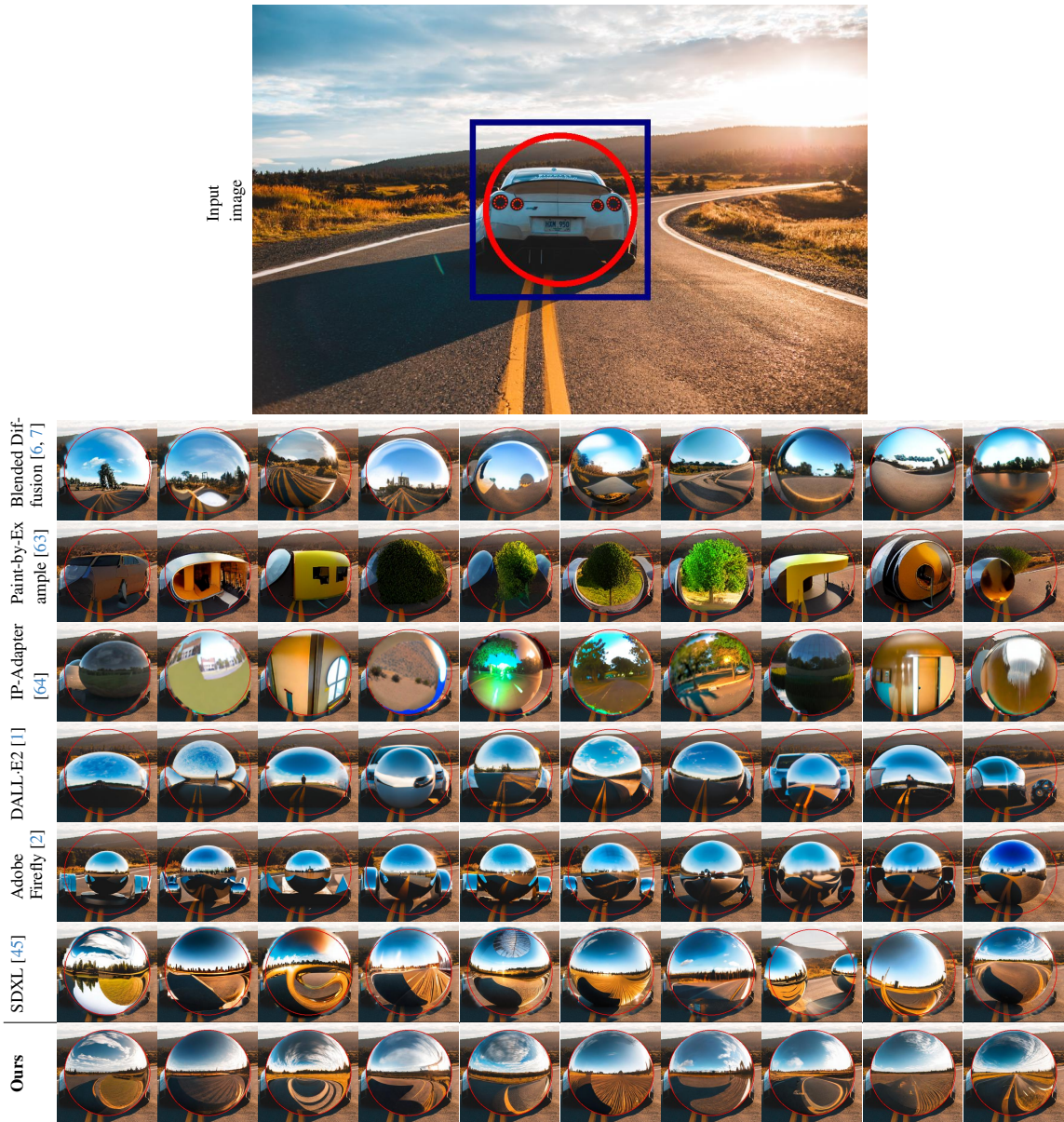


Figure 21. Chrome ball inpainting results from various methods. The red circle indicates the inpainted region, and we show a zoomed-in view of the blue crop. Each row contains results from ten different random seeds.

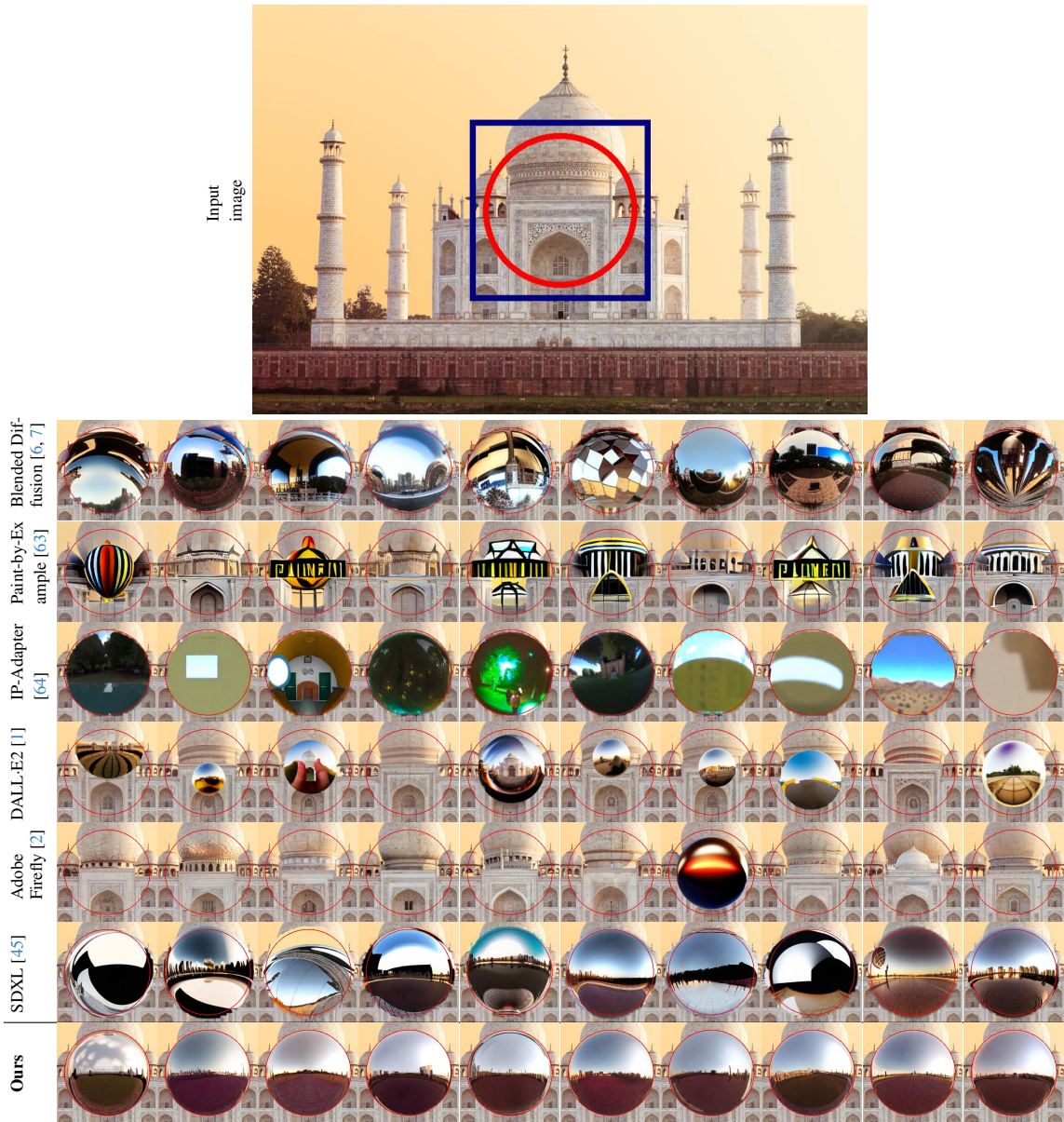


Figure 22. Chrome ball inpainting results from various methods. The red circle indicates the inpainted region, and we show a zoomed-in view of the blue crop. Each row contains results from ten different random seeds.

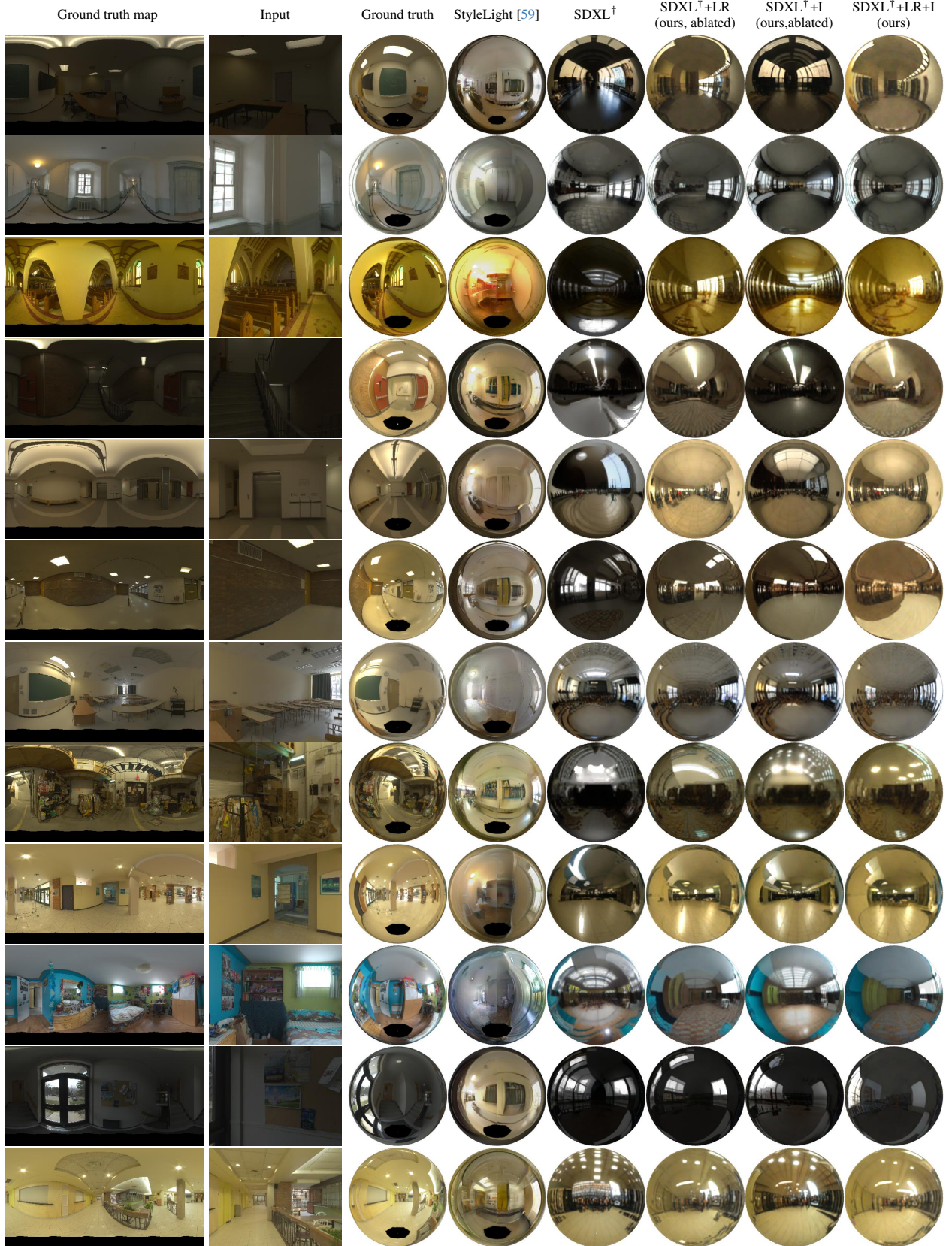


Figure 23. Qualitative results for the Laval indoor dataset using mirror balls.

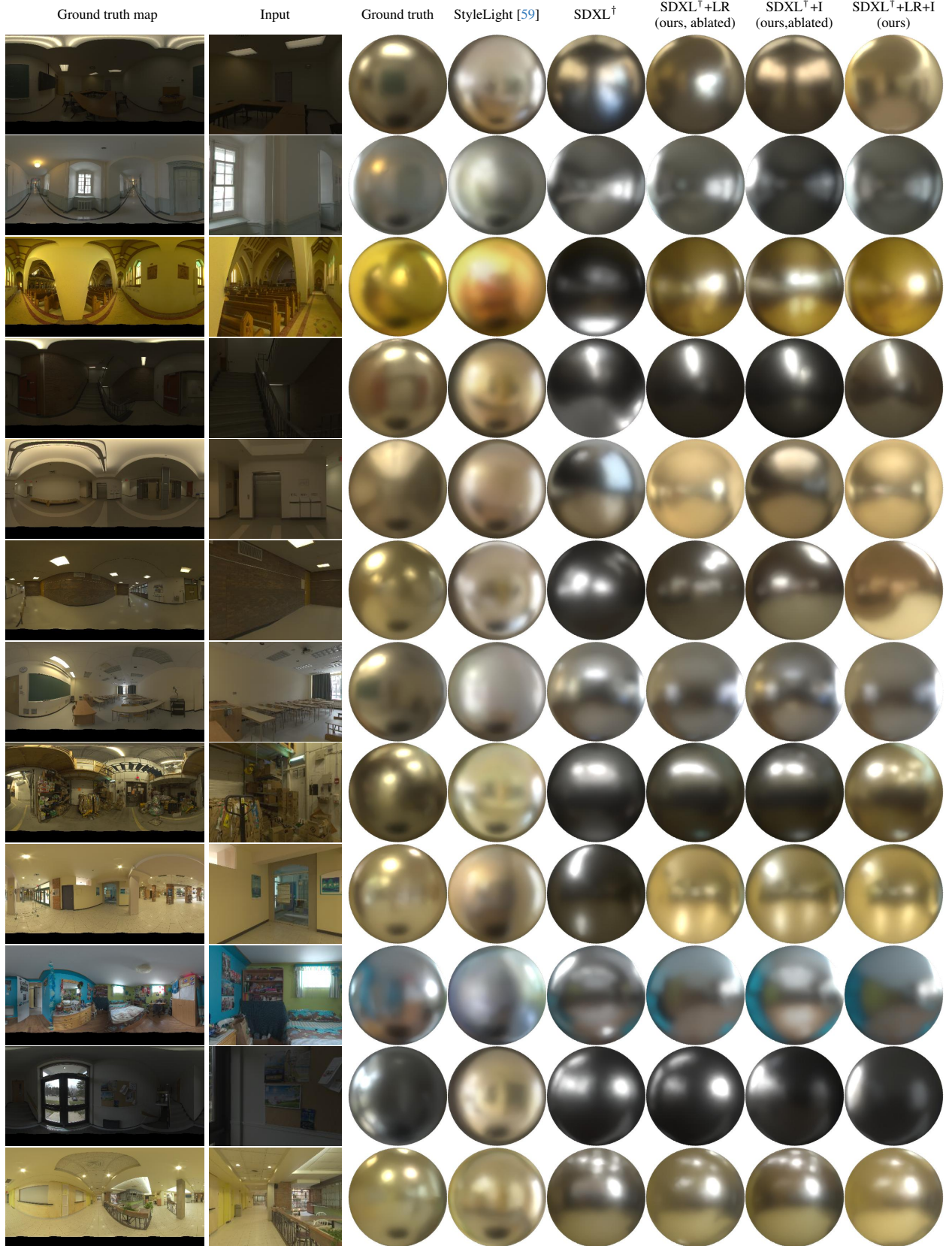


Figure 24. Qualitative results for the Laval indoor dataset using matte balls.

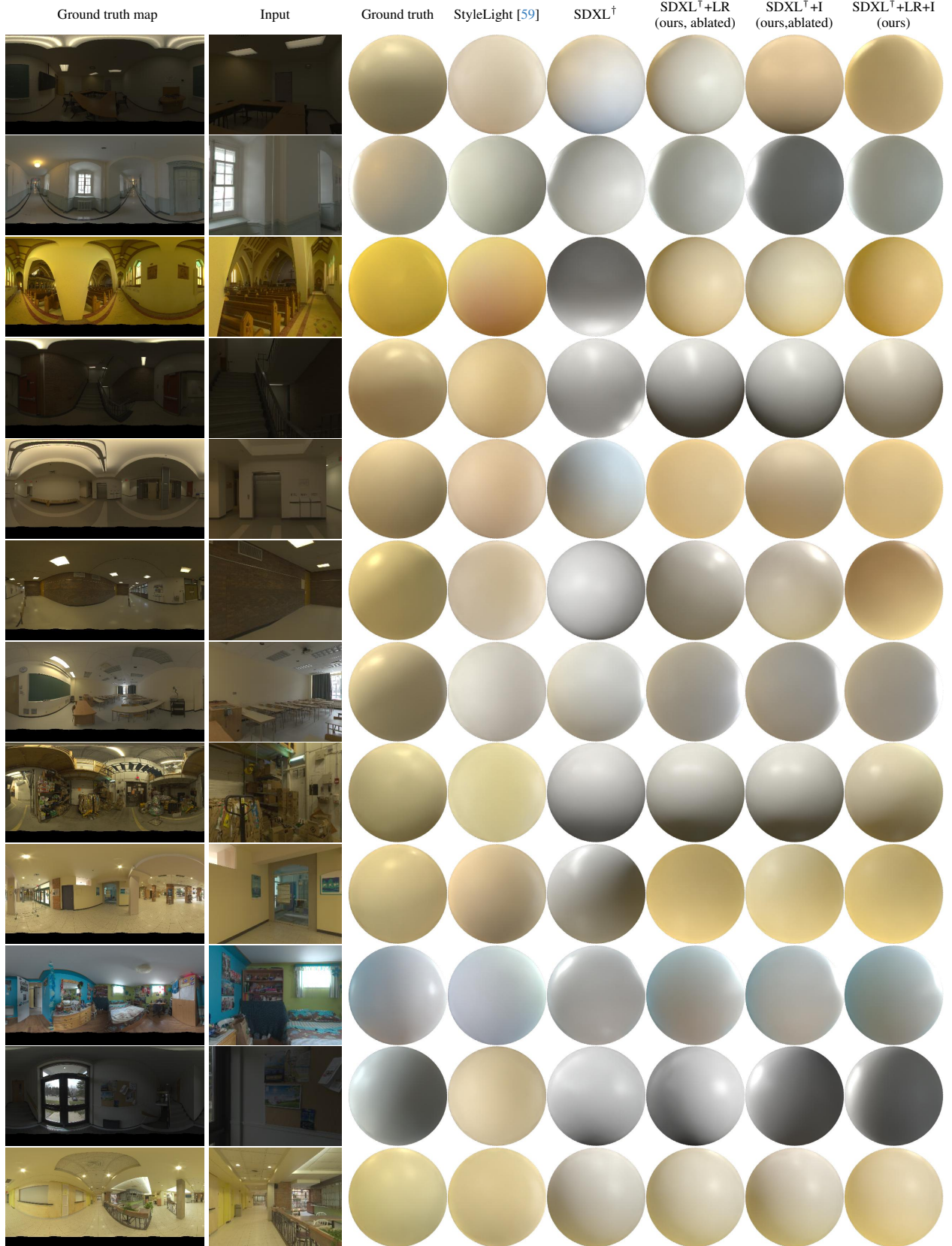


Figure 25. Qualitative results for the Laval indoor dataset using diffuse balls.

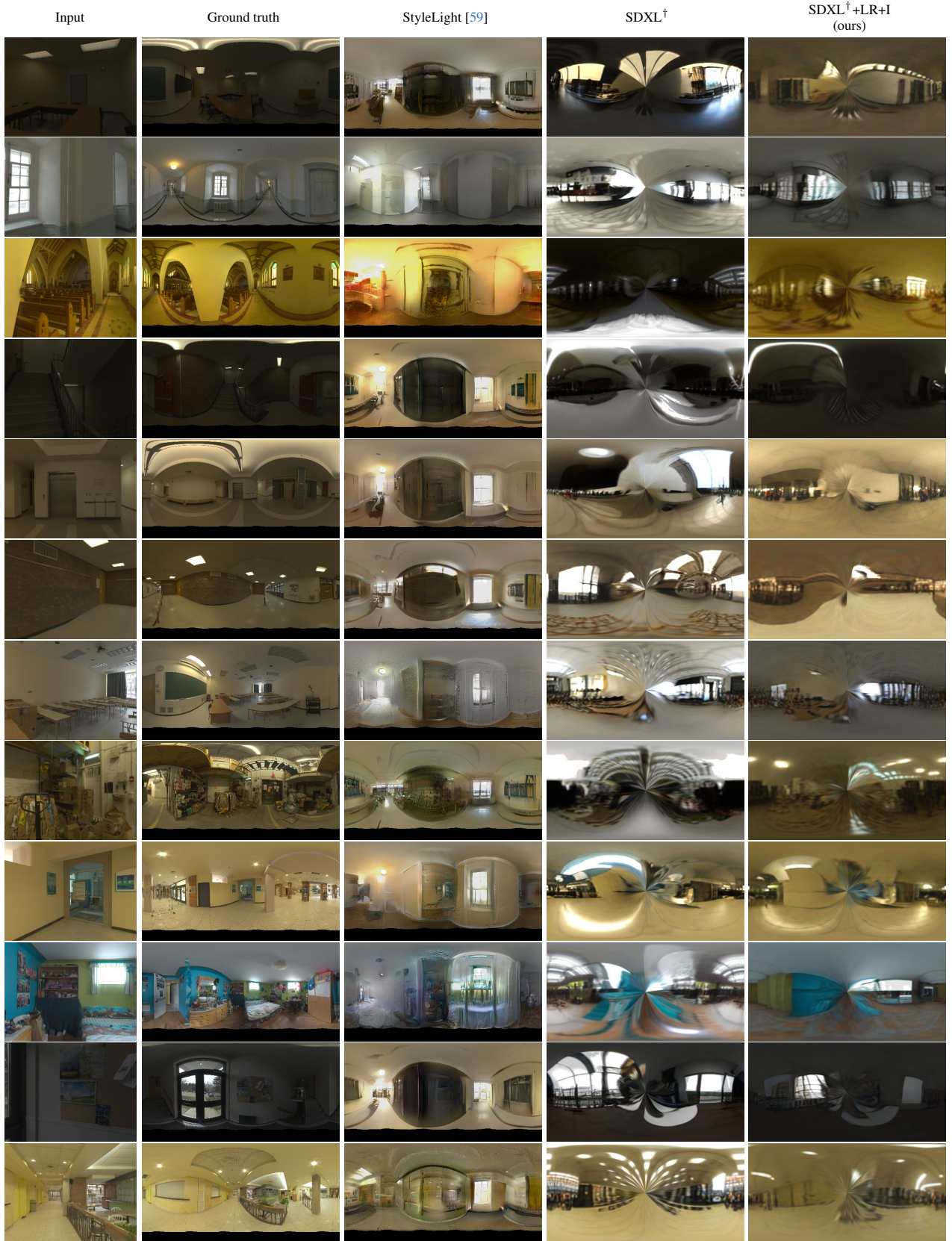


Figure 26. Unwarped equirectangular maps for the Laval indoor dataset.

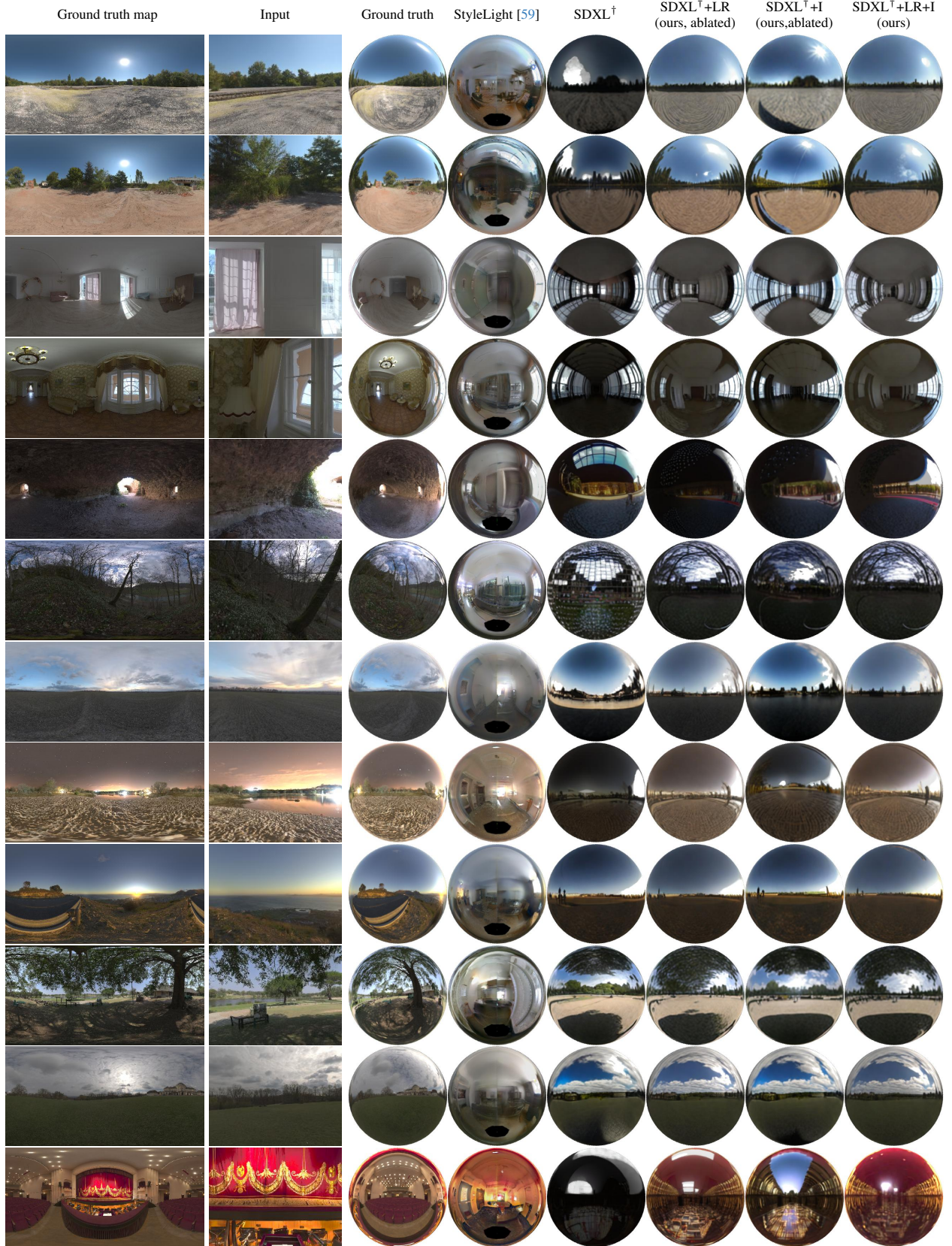


Figure 27. Qualitative results for the Poly Haven dataset using mirror balls.

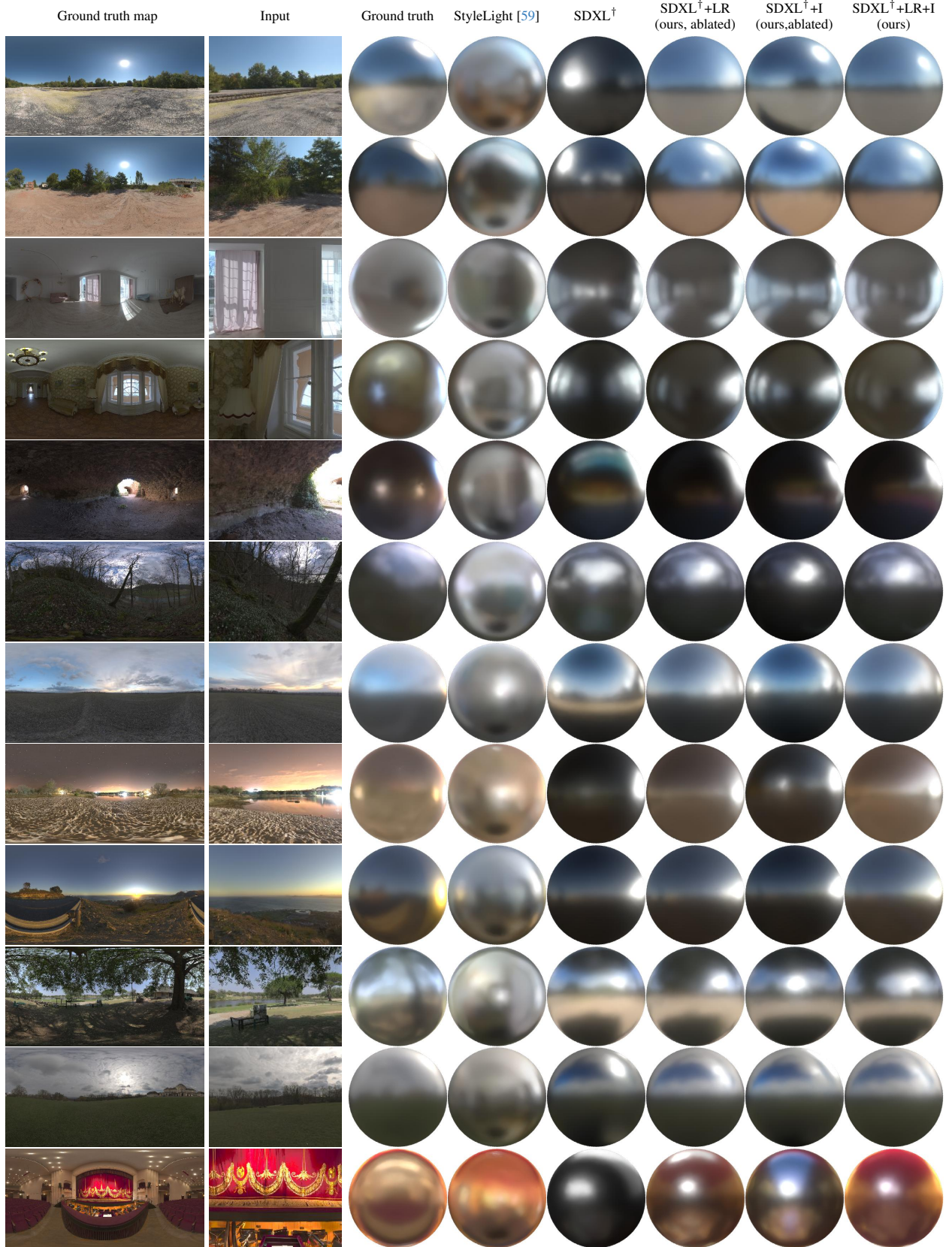
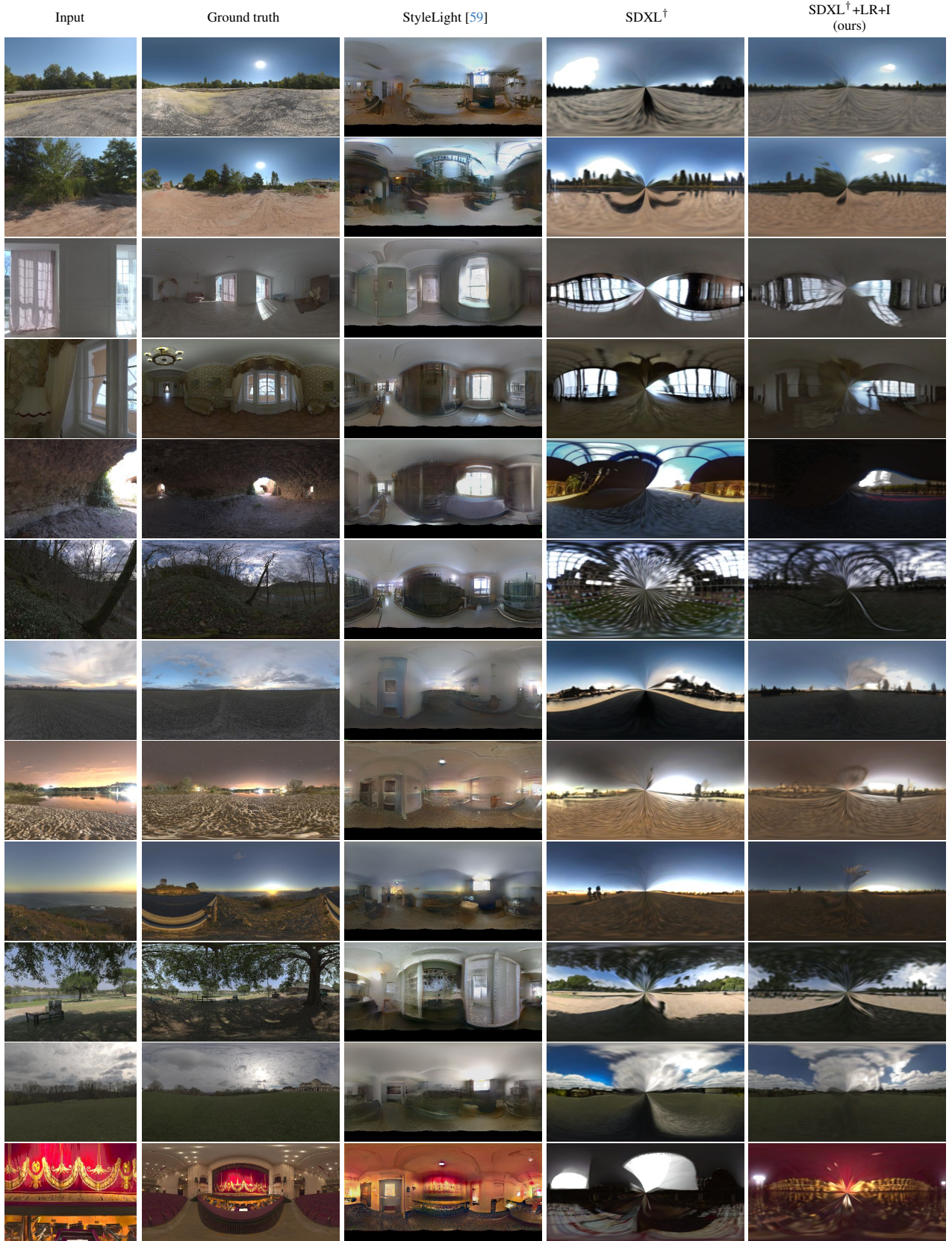
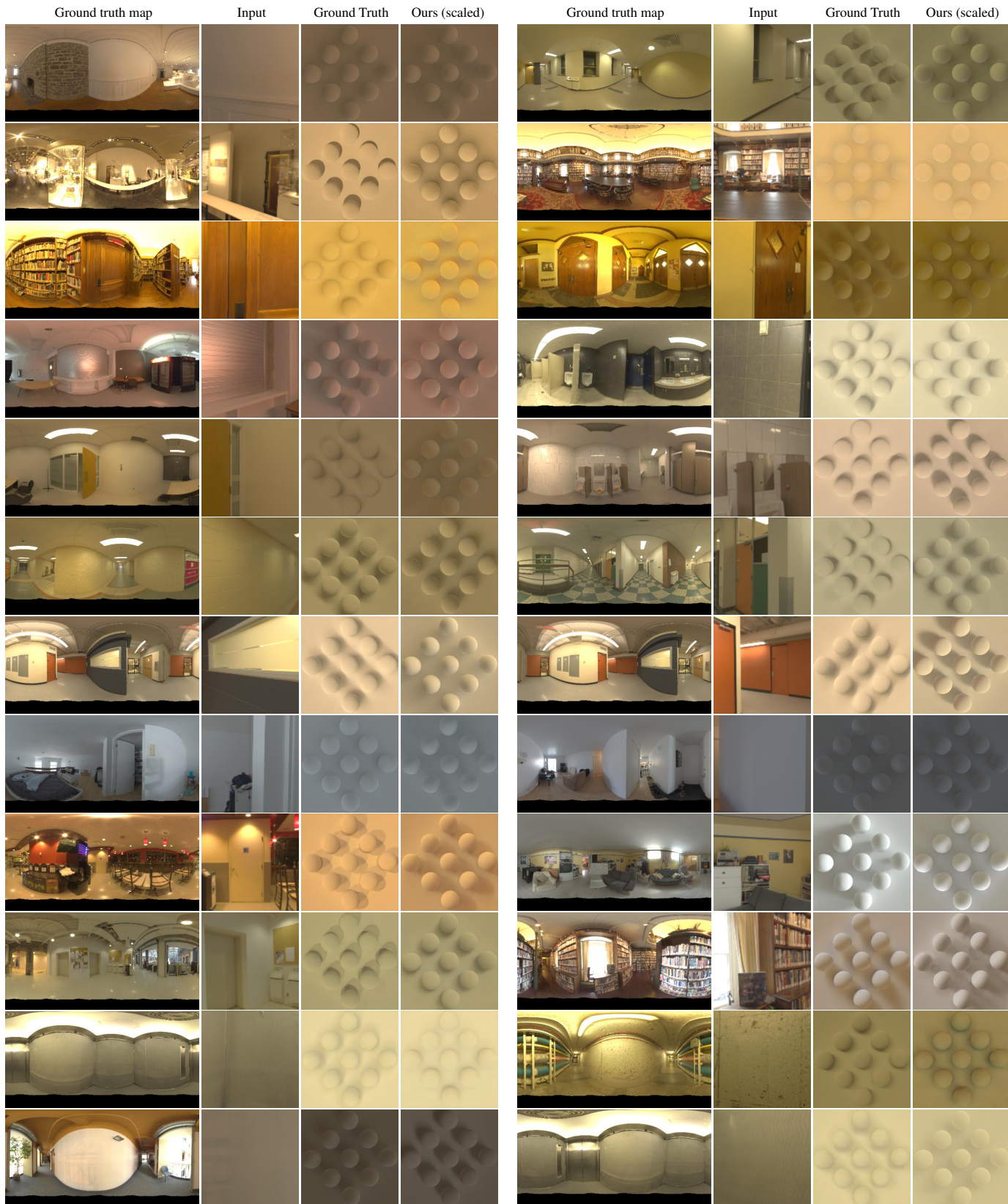




Figure 29. Qualitative results for the Poly Haven dataset using diffuse balls.





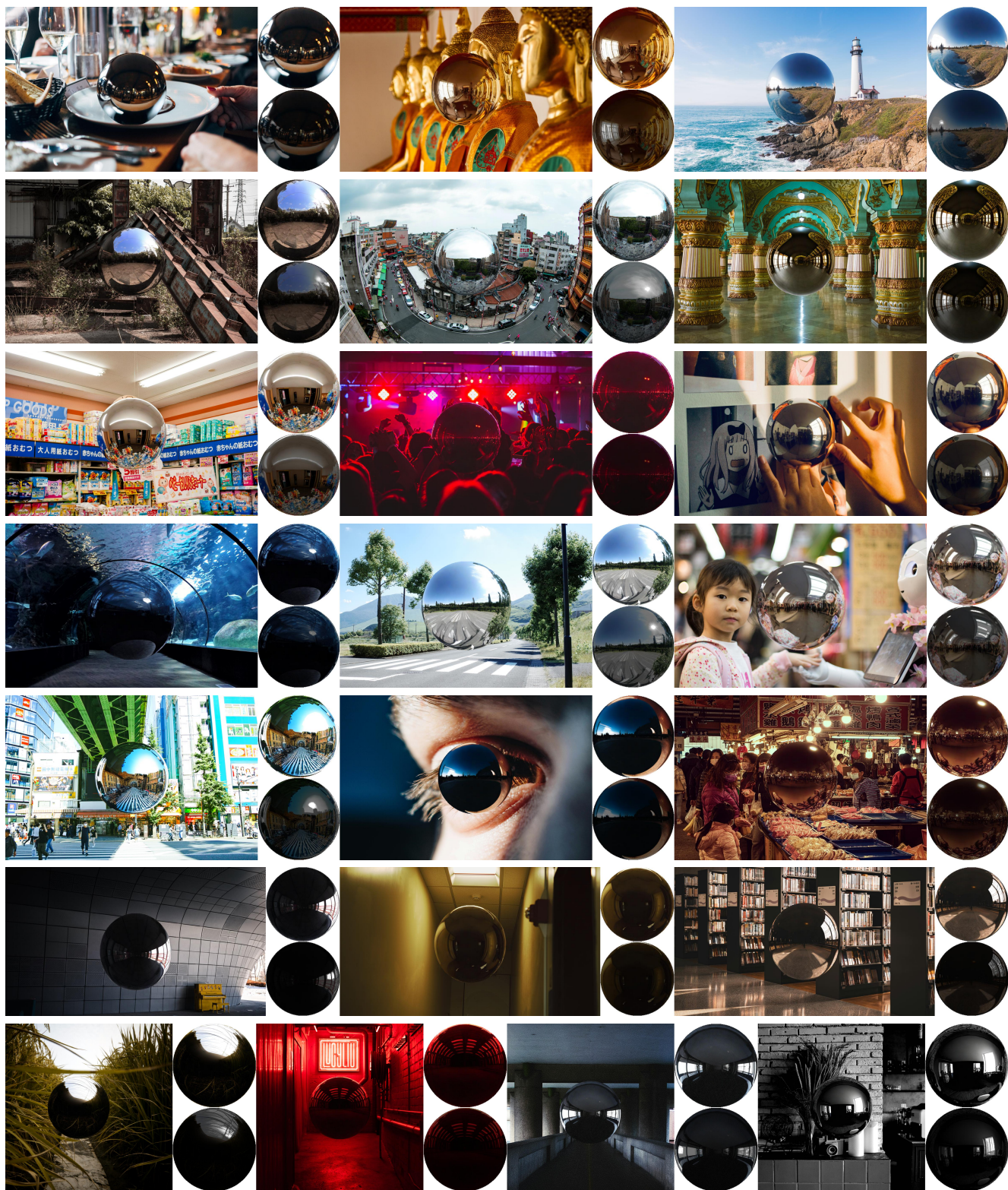


Figure 32. Additional qualitative results for in-the-wild scenes. For each input, we show a chrome ball generated from our pipeline and its underexposed version.

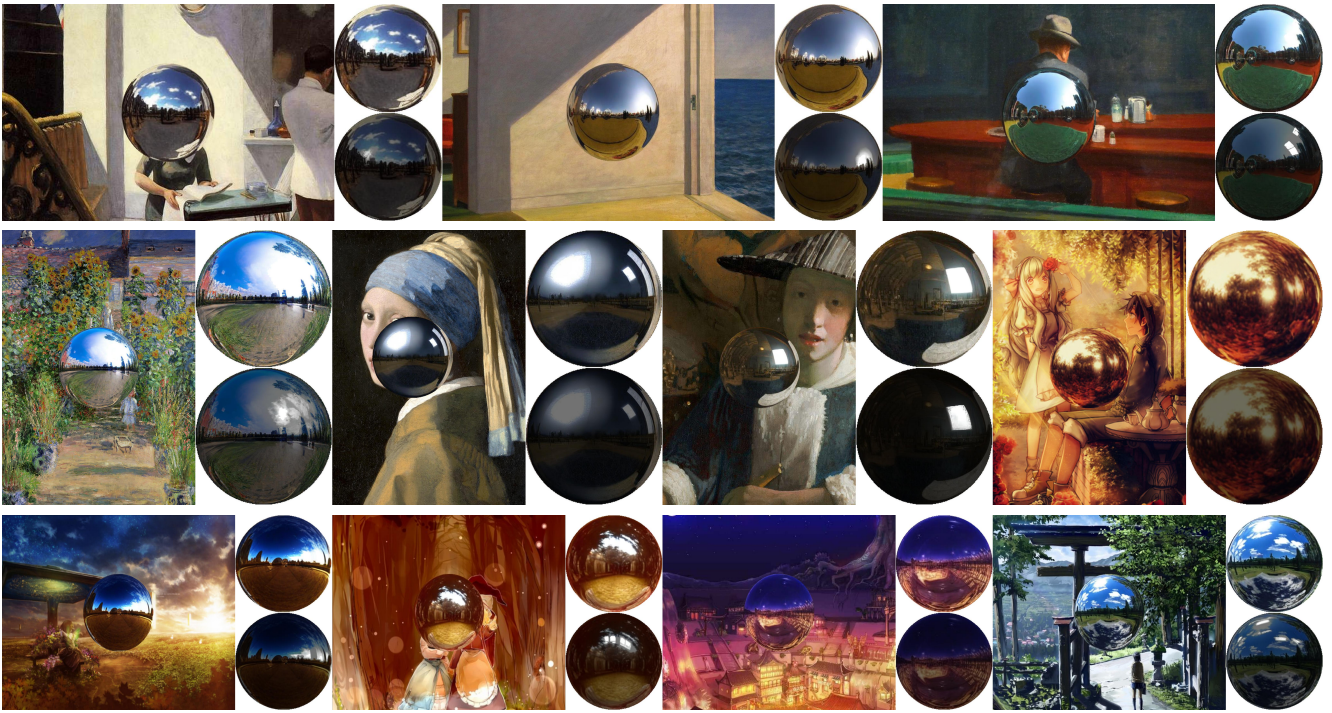


Figure 33. Qualitative results for artificial images such as paintings and painting-like Japanese animation-style images. For each input, we show a chrome ball generated from our pipeline and its underexposed version. Our proposed method can still perform reasonably well, albeit with some performance degradation, by leveraging the strong generative prior of SDXL [45].