S^{2}D-NeRF: Screen and Space Decoupling for Distractor-Free Novel View Synthesis

Anonymous authors
Paper under double-blind review

Abstract

Neural Radiance Fields (NeRFs) achieve photorealistic novel view synthesis, but assume that the scene is consistently observed across all views. In real-world captures, transient objects and the shadows they cast violate this assumption and lead to floaters and ghosting artifacts. Existing methods often identify distractors using semantic priors or heuristic rules, making the separation sensitive to segmentation quality or hand-designed assignment criteria. Instead, we find that cross-view consistency itself provides a strong cue for separating distractors, and our representation is designed to turn this cue into an effective inductive bias. We therefore propose S^{2}D-NeRF, which models the static scene with a shared 3D field and distractors with per-image 2D fields directly defined in pixel coordinates. This asymmetric representation drives their separation during joint optimization from the first iteration. Experiments on the NeRF On-the-go, RobustNeRF, and Photo Tourism datasets show competitive or superior rendering quality without semantic priors or heuristic rules. Our code will be publicly available.

1 Introduction

Novel view synthesis (NVS) is a long-standing problem in computer vision, aiming to learn a 3D scene representation from a set of posed images and render photorealistic novel views. Neural radiance fields (NeRF) (Mildenhall et al., 2020) and their variants (Barron et al., 2022; 2023) have made remarkable progress in this task. However, they typically assume a static scene consistently observed across all input views. This assumption often fails in real-world captures containing transient distractors, such as moving objects and their cast shadows. Such cross-view inconsistencies can be incorrectly reconstructed, producing artifacts such as floaters and ghosting.

Most existing methods identify distractors by detection or semantic supervision (Chen et al., 2024), leveraging features or masks from external pretrained vision models (Kirillov et al., 2023). However, view-inconsistent effects such as shadows cast by distractors may lack clear semantic cues. As shown in Fig. 1 (right), methods relying on DINOv2 features (Oquab et al., 2024) can miss these effects, leaving them in the static reconstruction (Ren et al., 2024). Other methods mask distractor pixels by thresholding photometric residuals or predicted uncertainty (Sabour et al., 2023; Wang et al., 2024; Duan & Wong, 2026). This hard thresholding is non-differentiable, so pixel assignment follows a prescribed rule rather than being optimized directly by the reconstruction objective.

We find that cross-view consistency provides a cue for separating distractors. An asymmetric representation enables us to exploit this cue. As illustrated in Fig. 1 (left), a 3D field shared across views couples observations from different images: view-consistent content can be learned once and reused across views, whereas fitting view-inconsistent content imposes conflicting constraints on the shared field. Conversely, a per-image 2D field is free from cross-view geometric constraints and can explain image-specific content independently, but recurring content must be represented separately in each image. We refer to this asymmetry as an *Asymmetric Representation Prior (ARP)*, which favors assigning view-consistent content to the shared 3D field and view-inconsistent content to the per-image 2D fields. This prior provides the key bias for static–transient separation without semantic supervision or a threshold on the residuals.

Building on ARP, we represent the static scene using a shared 3D radiance field while modeling distractors with per-image 2D fields. The two branches are alpha-composited and jointly optimized to

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

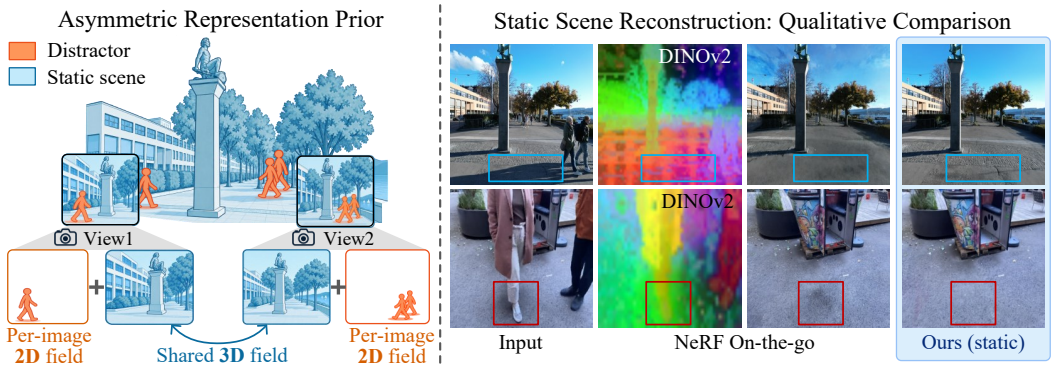


Figure 1: Asymmetric Representation Prior (ARP) and qualitative comparison. **Left:** With the ARP, the shared 3D field efficiently learns content consistent across views, while independent 2D fields capture image-specific distractors. **Right:** Semantic priors struggle to distinguish cast shadows from the static scene, leaving shadow artifacts in the result of NeRF On-the-go (Ren et al., 2024). In contrast, our method produces a cleaner static reconstruction in a scene containing both distractors and their cast shadows.

reconstruct the input images, while novel views are rendered from the static field alone. Benefiting from the ARP, the two branches naturally achieve static–transient separation during joint optimization. To further obtain a cleaner decomposition, we introduce additional regularization terms to resolve the remaining ambiguities between the two fields.

Our contributions are summarized as follows: we introduce S^{2}D-NeRF, a distractor-free NVS framework that represents the static scene with a shared 3D radiance field and distractors with per-image 2D fields. We identify an *asymmetric representation prior* arising from their different sharing structures. We further analyze in detail why this prior leads to successful static–transient separation and show that, together with initialization and regularization, it achieves effective separation without semantic supervision. Experiments on the NeRF On-the-go (Ren et al., 2024), RobustNeRF (Sabour et al., 2023), and Photo Tourism (Snavely et al., 2006) datasets demonstrate competitive or superior rendering quality compared with state-of-the-art methods.

2 Related Work

2.1 Distractor Identification and Masking

One family of methods keeps a single static representation and removes distractors by deciding which observations to trust (Kulhanek et al., 2024; Bao et al., 2026). Sabour et al. (2023) treat distractor pixels as photometric outliers and down-weight them with a robust estimator; Wang et al. (2024) exploit the fact that consistent content is fitted before inconsistent content to filter distractors; Duan & Wong (2026) discard high-uncertainty pixels after training, and Goli et al. (2024) threshold a post-hoc uncertainty field to prune the density of an already-trained NeRF. Other methods derive the decision from pretrained vision models: NeRF On-the-go (Ren et al., 2024) predicts uncertainty from DINOv2 features (Oquab et al., 2024), NeRF-HuGS (Chen et al., 2024) refines SfM and residual cues into masks with SAM (Kirillov et al., 2023), SpotLessSplats (Sabour et al., 2025) clusters Stable Diffusion features (Rombach et al., 2022), Dronesplat (Tang et al., 2025) propagates residual-based masks across frames with SAM2 (Ravi et al., 2025), and RobustSplat (Fu et al., 2025) bootstraps masks from DINOv2 feature similarity while delaying Gaussian densification.

In summary, these methods rely on weights or masks to exclude distractors, sometimes refined with semantic priors for better precision, and are therefore limited by segmentation granularity and regions without clear semantics. Our method needs no such decision: under ARP, the separation between distractors and the static scene emerges from view consistency, without relying on semantic priors or pretrained models.

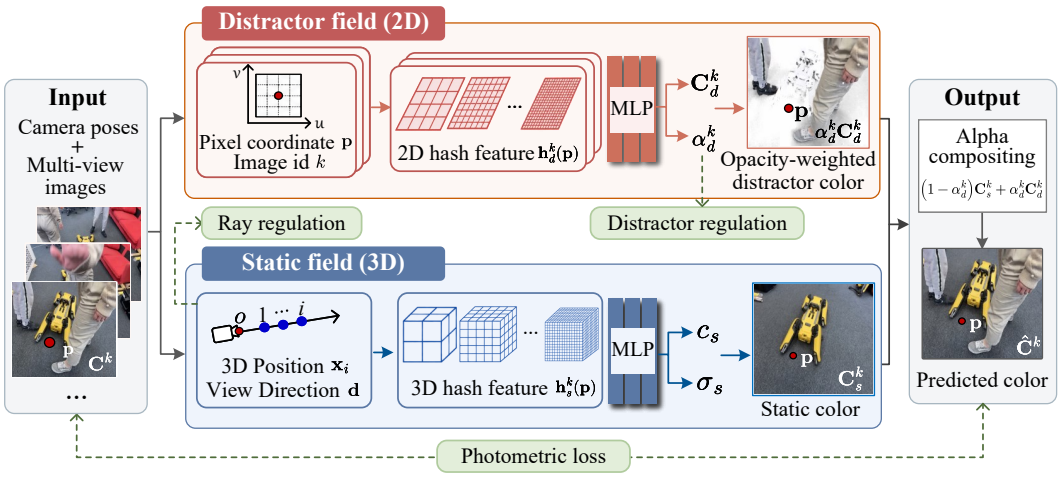


Figure 2: Overview of our method. A per-image 2D distractor field (top) predicts the distractor color $\mathbf{C}_d^k$ and opacity α_d^k at each pixel, while a 3D static field shared across views (bottom) volume-renders the static color $\mathbf{C}_s^k$. The two are alpha-composited into the prediction $\hat{\mathbf{C}}^k$ and jointly optimized with a photometric loss, together with distractor and ray regularization.

2.2 Scene Decomposition

A second family explains view-inconsistent content with a separate representation (Wang et al., 2025a; Lin et al., 2023). NeRF-W (Martin-Brualla et al., 2021) pairs the static field with a transient field conditioned on a per-image embedding to model transient content and estimate image-specific uncertainty. Ha-NeRF (Chen et al., 2022) and GS-W (Zhang et al., 2024) instead use auxiliary branches to predict visibility masks that down-weight distractor-affected pixels during reconstruction. For monocular videos, D^{2}NeRF (Wu et al., 2022) and NeuralDiff (Tschernezki et al., 2021) decompose each frame into a time-invariant static field and time-conditioned dynamic fields, whose rendered outputs are composited to reconstruct the observation. Gaussian-based methods adopt a similar decomposition at the primitive level. Swift4D (Wu et al., 2025) classifies Gaussians as static or dynamic and applies a time-dependent 4D hash-based deformation only to the dynamic primitives. Methods based on 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) adopt a similar decomposition at the primitive level. DeSplat (Wang et al., 2025b) models distractors with per-view 3DGS that are randomly initialized within each camera frustum and whose number is governed by threshold-triggered densification. HybridGS (Lin et al., 2025) models transients with per-image 2D Gaussians and trains the two branches in three stages: a warm-up of the static 3D Gaussians, an iterative stage that alternately optimizes the two branches, and a joint fine-tuning stage. In the second stage, the static 3DGS are supervised only outside a transient mask obtained by thresholding the opacity of the 2D Gaussians.

Among these methods, DeSplat (Wang et al., 2025b) and HybridGS (Lin et al., 2025) are the closest to ours, pairing shared 3D Gaussians with per-view 3D and 2D Gaussians, respectively. The key difference lies in the representation. Gaussians are discrete primitives, so both methods rely on prescribed rules outside gradient descent. DeSplat (Wang et al., 2025b) adds and removes distractor Gaussians through densification and pruning, and HybridGS (Lin et al., 2025) relies on staged training with a binarized transient mask. In contrast, our static and distractor fields are neural fields whose capacity covers every point of the scene and every pixel of each image from the start. Both fields are therefore optimized purely by gradient descent under a single differentiable objective.

3 Method

Given a set of posed images captured under unconstrained conditions, our goal is to learn a static radiance field free from transient distractors. Built upon Zip-NeRF (Barron et al., 2023), we represent the static scene with a shared 3D field and distractors with per-image 2D fields, which are alpha-composited and jointly optimized with photometric and regularization losses (Fig. 2). We

briefly review the Zip-NeRF backbone in Sec. 3.1, followed by the static–transient decomposition and regularization in Sec. 3.2 and Sec. 3.3.

3.1 PRELIMINARY

Given a ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ originating from the camera center $\mathbf{o}$ with viewing direction $\mathbf{d}$, Zip-NeRF (Barron et al., 2023) samples n points $\{\mathbf{x}^i = \mathbf{r}(t^i)\}_{i=1}^n$ between the near and far bounds. Each position is encoded using a multi-resolution hash grid (Müller et al., 2022), and the resulting feature $\mathbf{h}(\mathbf{x}^i)$ is decoded by a compact Multi-Layer Perceptron (MLP) F into a view-dependent color $\mathbf{c}^i$ and a volume density σ^i:

$$(\mathbf{c}^i, \sigma^i) = F\big(\mathbf{h}(\mathbf{x}^i), \mathbf{d}; \boldsymbol{\theta}\big), \tag{1}$$

where $\boldsymbol{\theta}$ denotes the learnable parameters of F. Subsequently, the opacity α^i of the i-th sampled point $\mathbf{x}^i$ is computed as:

$$\alpha^i = 1 - \exp(-\sigma^i \delta^i), \tag{2}$$

where $\delta^i = t^{i+1} - t^i > 0$ denotes the distance between adjacent samples. The final pixel color $\hat{\mathbf{C}}(\mathbf{p})$ at pixel $\mathbf{p}$ is obtained through volume rendering:

$$\hat{\mathbf{C}}(\mathbf{p}) = \sum_{i=1}^{n} T^i \, \alpha^i \, \mathbf{c}^i, \quad T^i = \prod_{j<i} (1 - \alpha^j), \tag{3}$$

where T^i the accumulated transmittance. The field is optimized by minimizing a photometric loss between the rendered color $\hat{\mathbf{C}}(\mathbf{p})$ and observed colors $\mathbf{C}(\mathbf{p})$:

$$\mathcal{L}_{\text{rgb}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \rho\big(\hat{\mathbf{C}}(\mathbf{p}) - \mathbf{C}(\mathbf{p})\big), \tag{4}$$

where $\mathcal{P}$ is the set of pixels sampled in a training batch, and $\rho(\mathbf{x}) = \sqrt{\|\mathbf{x}\|_2^2 + \epsilon^2}$ is the Charbonnier loss (Barron et al., 2022).

3.2 STATIC–TRANSIENT DECOMPOSITION

Static scenes and distractors exhibit distinct cross-view properties: static scenes maintain a consistent 3D structure across views, whereas distractors are view-inconsistent. We therefore model them with different representations and fuse their renderings by alpha compositing. For distractors, we construct a 2D hash grid for each training image to represent their image-specific appearance in screen space. Given a pixel coordinate $\mathbf{p} = (u, v)$ in the k-th image, we query the corresponding feature $\mathbf{h}_d^k(\mathbf{p})$ from its 2D hash grid and feed it into a lightweight MLP F_d to predict the pixel-wise distractor color $\mathbf{C}_d^k(\mathbf{p})$ and opacity α_d^k:

$$\big(\mathbf{C}_d^k(\mathbf{p}), \alpha_d^k(\mathbf{p})\big) = F_d\big(\mathbf{h}_d^k(\mathbf{p}); \boldsymbol{\theta}_d\big), \tag{5}$$

where $\boldsymbol{\theta}_d$ denotes the learnable parameters of F_d.

For the static scene, we employ a cross-view shared 3D hash grid to model its multi-view consistent 3D structure. For a pixel $\mathbf{p}$ in the k-th input image, we cast a ray $\mathbf{r}$ through the pixel $\mathbf{p}$ and sample a set of 3D position $\{\mathbf{x}^i = \mathbf{r}(t^i)\}_{i=1}^n$ along it. For each sample $\mathbf{x}^i$, we query feature $\mathbf{h}_s(\mathbf{x}^i)$ from the shared 3D hash grid and feed it, together with the viewing direction $\mathbf{d}$, into an MLP F_s to predict the static scene color $\mathbf{c}_s^i$ and density σ_s^i:

$$\big(\mathbf{c}_s^i, \sigma_s^i\big) = F_s\big(\mathbf{h}_s(\mathbf{x}^i), \mathbf{d}; \boldsymbol{\theta}_s\big), \tag{6}$$

where $\boldsymbol{\theta}_s$ denotes the learnable parameters of F_s. Each density σ_s^i is converted into an opacity α_s^i via Eq. (2), and the static color $\mathbf{C}_s^k(\mathbf{p})$ of pixel $\mathbf{p}$ in the k-th image is then obtained by volume rendering along the ray following Eq. (3).

We treat $\alpha_d^k(\mathbf{p})$ as the extent to which the distractor occludes the underlying static scene and alpha-composite the distractor and static renderings:

$$\hat{\mathbf{C}}^k(\mathbf{p}) = \big(1 - \alpha_d^k(\mathbf{p})\big)\mathbf{C}_s^k(\mathbf{p}) + \alpha_d^k(\mathbf{p})\mathbf{C}_d^k(\mathbf{p}), \tag{7}$$

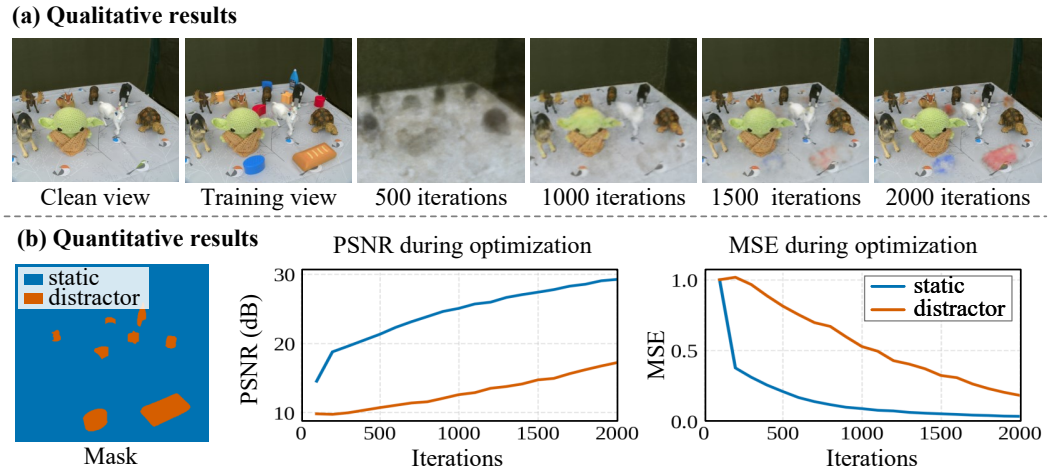


Figure 3: Fitting behavior of the static field on cross-view-consistent and view-inconsistent content. We fit a scene with distractors (*BabyYoda* (Sabour et al., 2023)) using the static field alone. (a) Renderings of the static field during the first 2000 iterations show that static content is recovered first, while the distractors remain semi-transparent at later iterations. (b) Using distractor masks, we measure PSNR and MSE separately on static and distractor regions across all images: the static field converges markedly faster on the former.

where $\hat{\mathbf{C}}^k(\mathbf{p})$ is the final predicted color of pixel $\mathbf{p}$ in the k-th image.

This decomposition is inherently ill-posed: multiple allocations of appearance between the static and distractor fields can produce the same training loss, while exhibiting different performance on novel views. However, we find that no explicit constraint is needed, as the ARP resolves this ambiguity on its own. Specifically, the ARP makes the two branches unequally efficient at the two kinds of content, so joint optimization tends to leave cross-view-consistent content to the static field and view-inconsistent content to the per-image fields. Fig. 3 supports this: optimized alone, the static field fits cross-view-consistent content markedly faster.

Alpha compositing further reinforces this behavior. We initialize α_d close to zero, suppressing the gradient propagated to the distractor color and allowing the static branch to dominate early optimization and first fit cross-view-consistent content. Once a region is sufficiently explained by the static field, its photometric residual approaches zero, reducing the gradient of α_d and preventing the distractor field from taking over. Consequently, the decomposition emerges from the ARP, initialization, and joint optimization rather than a heuristic decomposition rule.

3.3 Regularization

Although the ARP provides a basis for the decomposition, the resulting decomposition may still suffer from ambiguity in three respects: (1) *Ambiguous distractor opacity*. The ARP determines which branch explains a pixel, but nothing forces α_d to commit: it can settle at an intermediate value, leaving the pixel split between the two branches. (2) *Distractor field over-absorption*. The per-image distractor field can also reproduce static appearance with non-zero α_d, leaving the static field under-optimized in these regions. (3) *Incorrect geometric interpretation*. View-inconsistent distractors can be explained by the static field as floaters close to the camera plane.

Distractor Regularization. To address ambiguous distractor opacity, we introduce a binary entropy loss $\mathcal{L}_{\text{BE}}$ to encourage a clear separation between the two branches. It drives α_d toward the binary states 0 and 1. The binary entropy loss is defined as:

$$\mathcal{L}_{\text{BE}} = -\frac{1}{|\mathcal{K}|} \sum_{(k,\mathbf{p})\in\mathcal{K}} \Big[\alpha_d^k(\mathbf{p}) \log \alpha_d^k(\mathbf{p}) + \big(1-\alpha_d^k(\mathbf{p})\big) \log \big(1-\alpha_d^k(\mathbf{p})\big)\Big], \tag{8}$$

where $\mathcal{K}$ denotes the set of pixels sampled in a training batch, with k indexing the image each pixel belongs to. $\mathcal{L}_{\text{BE}}$ is symmetric with respect to the two binary states and does not favor either one. The

assignment of a pixel to the static or distractor branch is therefore still governed by the structural asymmetry between the two representations in Sec. 3.2.

To address distractor field over-absorption, we propose a duplicate color suppression (DCS) loss that drives α_d toward zero wherever the two fields predict similar colors, so that such regions are explained solely by the static field. Let $\Delta^k(\mathbf{p}) = \frac{1}{3}\sum_{c\in\{\mathrm{R,G,B}\}} \left|C^k_{d,c}(\mathbf{p}) - C^k_{s,c}(\mathbf{p})\right|$ be the mean absolute color difference between the distractor and static colors. The DCS loss is defined as:

$$\mathcal{L}_{\mathrm{DCS}} = \frac{1}{2|\mathcal{K}|} \sum_{(k,\mathbf{p})\in\mathcal{K}} \alpha^k_d(\mathbf{p}) \not\nabla \left[1 - \tanh\left(\frac{\Delta^k(\mathbf{p}) - \tau}{\eta}\right)\right], \tag{9}$$

where $\not\nabla$ is a stop-gradient operator, and τ and η set the center and width of the $\tanh$ transition. Unlike a linear weight $1 - \Delta^k(\mathbf{p})$, this narrow gate spares genuine distractors, whose $\Delta^k(\mathbf{p})$ is typically far above τ. Since the gate is detached, the loss acts only on α_d. Both $\mathcal{L}_{\mathrm{BE}}$ and $\mathcal{L}_{\mathrm{DCS}}$ are enabled only after the first 5000 training steps, once α_d has become relatively stable.

Ray Regularization. Following FreeNeRF (Yang et al., 2023), we employ occlusion regularization $\mathcal{L}_{\mathrm{occ}}$ to suppress static density close to the camera. Let $s_0 < s_1 < \dots$ denote the boundaries of the sampling intervals along ray $\mathbf{r}$, $\bar{s}_i = \frac{1}{2}(s_i + s_{i+1})$ the midpoint of the i-th interval, and $\sigma_s(\mathbf{r}, i)$ the static density of the i-th interval. With the near-camera mask $m_i(\mathbf{r}) = \mathbb{1}\left[\bar{s}_i < \rho_{\mathrm{occ}}\right]$, we define

$$\mathcal{L}_{\mathrm{occ}} = \frac{\sum_{\mathbf{r}\in\mathcal{R}} \sum_i m_i(\mathbf{r})\, \sigma_s(\mathbf{r}, i)}{\sum_{\mathbf{r}\in\mathcal{R}} \sum_i m_i(\mathbf{r})}, \tag{10}$$

where ρ_{occ} is the extent of the penalized region, and $\mathcal{R}$ denotes the set of rays sampled in a batch.

Loss Function. The full objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{rgb}} + \mathcal{L}_{\mathrm{zip}} + \lambda_{\mathrm{BE}}\,\mathcal{L}_{\mathrm{BE}} + \lambda_{\mathrm{DCS}}\,\mathcal{L}_{\mathrm{DCS}} + \lambda_{\mathrm{occ}}\,\mathcal{L}_{\mathrm{occ}}, \tag{11}$$

where $\mathcal{L}_{\mathrm{rgb}}$ is the photometric loss on the composite render, $\mathcal{L}_{\mathrm{zip}}$ collects the regularization terms of the Zip-NeRF backbone (Barron et al., 2023), which we keep unchanged with their default weights, and λ_{BE}, λ_{DCS}, and λ_{occ} weight our three regularizers.

4 Experiment

We evaluate our method on three commonly used public datasets and conduct quantitative and qualitative comparisons with state-of-the-art methods in Sec. 4.2. Ablation study in Sec. 4.3 demonstrates the effectiveness of the proposed components. Additional visualizations and results are provided in the supplementary material.

4.1 Experimental Setup

Dataset. For NeRF On-the-go (Ren et al., 2024) dataset, we evaluate on the six most commonly used scenes (*Mountain*, *Fountain*, *Corner*, *Patio*, *Spot*, and *Patio-High*). For the Robust-NeRF (Sabour et al., 2023) dataset, we use all five scenes, including *Statue*, *Android*, *Crab (1)*, *Crab (2)*, and *BabyYoda*. Moreover, we evaluate on the *Brandenburg Gate*, *Sacre Coeur*, and *Trevi Fountain* scenes from Photo Tourism (Snavely et al., 2006). Following NeRF-W (Martin-Brualla et al., 2021), we assign a 48-dimensional appearance embedding to each image and adopt its left/right evaluation protocol, detailed in Sec. A.1.

Baselines. We compare our method with four NeRF-based methods: Zip-NeRF (Barron et al., 2023), NeRF-HuGS (Chen et al., 2024), NeRF On-the-go (Ren et al., 2024), and Evidential-NeRF (Duan & Wong, 2026), and two 3DGS-based methods: DeSplat (Wang et al., 2025b) and HybridGS (Lin et al., 2025). Among them, Zip-NeRF serves as our baseline. NeRF On-the-go and NeRF-HuGS leverage semantic information for distractor segmentation, while DeSplat and HybridGS decompose the 3D scene representation for distractor-free NVS. We additionally compare with NeRF-W (Martin-Brualla et al., 2021), K-Planes (Fridovich-Keil et al., 2023), and WildGaussians (Kulhanek et al., 2024) on the Photo Tourism (Snavely et al., 2006) dataset, as these methods are specifically designed for in-the-wild reconstruction with varying illumination.

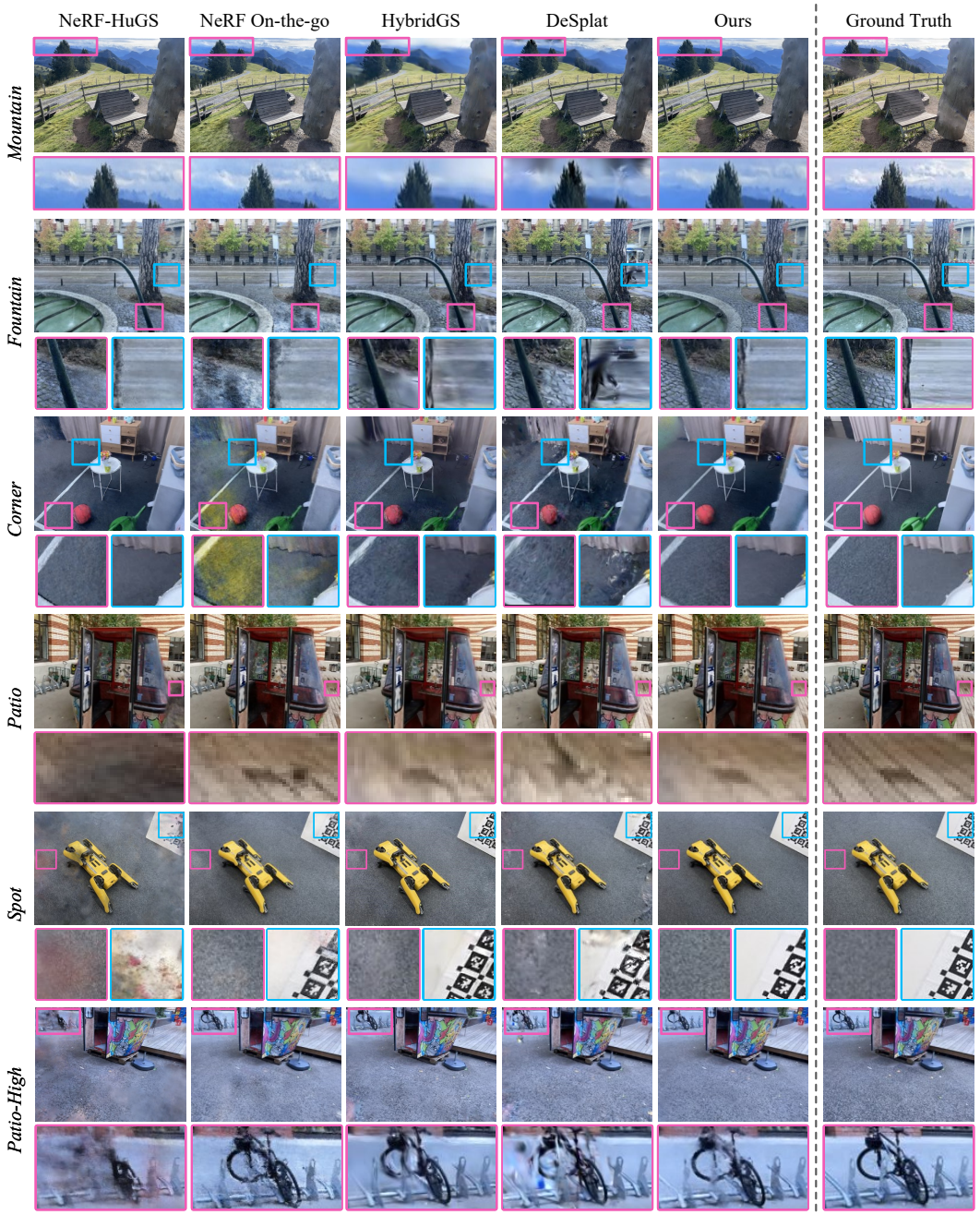


Figure 4: Qualitative results on the NeRF On-the-go dataset.

Metrics. We use PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) to quantitatively evaluate rendering quality.

4.2 Evaluation

Comparison on NeRF On-the-go Dataset. Table 1 and Fig. 4 report the results. NeRF-HuGS depends on the segmentation granularity of SAM (Kirillov et al., 2023), which splits pedestrians into parts and leaves artifacts in Spot and Patio. NeRF On-the-go derives uncertainty from DINOv2 features (Oquab et al., 2024), so shadows without distinctive semantics remain static content and produce uneven ground illumination in Patio-High. DeSplat places per-view distractor Gaussians by densification heuristics, which leaves residual distractors in Fountain and absorbs view-dependent

Table 1: Quantitative comparison on the NeRF On-the-go dataset. The best and second-best results in each column are highlighted with red and orange cell backgrounds, respectively.

Method	Low Occlusion						Medium Occlusion						High Occlusion					
	Mountain			*Fountain*			*Corner*			*Patio*			*Spot*			*Patio-High*		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
DeSplat (Wang et al., 2025b)	20.29	0.68	0.17	20.67	0.68	0.17	24.74	0.81	0.13	20.89	0.79	0.12	24.55	0.78	0.16	21.35	0.73	0.19
HybridGS (Lin et al., 2025)	21.73	0.69	0.28	21.11	0.67	0.25	25.03	0.84	0.15	21.98	0.81	0.17	24.33	0.79	0.19	21.77	0.74	0.21
Zip-NeRF (Barron et al., 2023)	18.82	0.62	0.28	19.95	0.66	0.23	20.85	0.66	0.30	14.30	0.57	0.34	15.38	0.35	0.70	14.30	0.32	0.60
NeRF-HuGS (Chen et al., 2024)	19.26	0.58	0.24	19.61	0.59	0.24	23.81	0.77	0.16	18.83	0.70	0.19	20.11	0.62	0.33	20.08	0.60	0.30
NeRF On-the-go (Ren et al., 2024)	20.15	0.64	0.26	20.11	0.61	0.31	24.22	0.81	0.19	20.87	0.75	0.22	23.33	0.79	0.19	21.41	0.72	0.24
EvidentialNeRF (Duan & Wong, 2026)	17.99	0.61	0.30	18.70	0.60	0.30	22.02	0.74	0.24	20.77	0.77	0.16	15.73	0.43	0.75	16.18	0.44	0.50
Ours	21.50	0.74	0.16	21.59	0.72	0.16	25.13	0.84	0.11	22.07	0.81	0.13	24.10	0.80	0.12	22.12	0.76	0.17

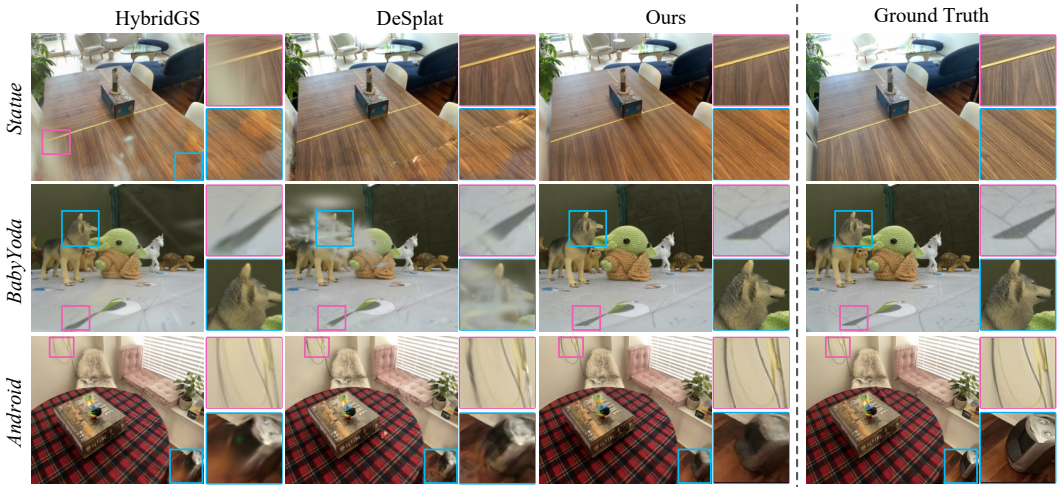


Figure 5: Qualitative results on the RobustNeRF dataset.

static content such as the sky in Mountain. HybridGS leaves ghosting in Spot and less static detail in Fountain. EvidentialNeRF thresholds its predicted uncertainty after training, and its quality drops on the high-occlusion scenes. Our per-image field has capacity at every pixel and a continuous opacity optimized by the objective itself, so distractors and semi-transparent shadows are captured without such heuristics. We obtain the best or second-best result in 17 of the 18 entries.

Comparison on RobustNeRF Dataset. Table 2 and Fig. 5 present the results on the RobustNeRF dataset. Our method achieves the best LPIPS on all five scenes and the best SSIM on four of them. We attribute the PSNR gap to the uneven illumination and strong reflections in the ground-truth images: such view-inconsistent variations are partly absorbed by the per-image distractor field and thus removed from the static rendering. Nevertheless, as shown in Fig. 5, our method removes distractors more cleanly than the baselines.

Comparison on Photo Tourism Dataset. As shown in Table 3, our method achieves the best SSIM on two of the three scenes and the best PSNR on *Sacre Coeur*, while ranking second in LPIPS on *Brandenburg Gate* and *Sacre Coeur*. This indicates that our distractor field can separate occluders from the static scene under large appearance and exposure variations, while the appearance embedding represents the per-image illumination.

4.3 Ablation Study

Table 4 ablates each component on the NeRF On-the-go dataset. Adding the per-image 2D distractor field alone (row 0) brings the largest gain over the Zip-NeRF baseline, confirming that the ARP already drives most of the decomposition (Sec. 3.2). Each regularizer improves PSNR over row 0, by 0.13 to 0.91 dB. Their individual effects on the other two metrics are much smaller, and $\mathcal{L}_{\mathrm{BE}}$ and $\mathcal{L}_{\mathrm{DCS}}$ are marginally below row 0 on SSIM. Note that $\mathcal{L}_{\mathrm{occ}}$ is not required in general and is enabled only on the few scenes with near-camera floaters (*Spot* here, and *Crab (1)* on RobustNeRF); its weight is set to zero on the remaining scenes, and the averages in Table 4 are computed accordingly.

Table 2: Quantitative comparison on the RobustNeRF dataset. The best and second-best results in each column are highlighted with red and orange cell backgrounds, respectively.

Method	*Statue*			*Android*			*BabyYoda*			*Crab (1)*			*Crab (2)*		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
DeSplat (Wang et al., 2025b)	23.43	0.87	0.10	24.80	0.85	0.08	35.20	0.95	0.06	36.66	0.97	0.04	35.29	0.95	0.07
HybridGS (Lin et al., 2025)	22.93	0.87	0.10	25.15	0.85	0.07	35.32	0.96	0.07	36.31	0.97	0.05	35.17	0.96	0.08
Zip-NeRF (Barron et al., 2023)	19.47	0.76	0.24	23.92	0.81	0.10	33.77	0.96	0.07	23.94	0.93	0.10	32.82	0.96	0.05
NeRF-HuGS (Chen et al., 2024)	21.00	0.77	0.14	23.32	0.76	0.12	30.70	0.83	0.12	34.16	0.96	0.03	–	–	–
NeRF On-the-go (Ren et al., 2024)	21.58	0.77	0.24	23.54	0.79	0.22	31.16	0.93	0.24	29.92	0.93	0.16	30.15	0.92	0.19
EvidentialNeRF (Duan & Wong, 2026)	20.24	0.78	0.20	23.21	0.79	0.14	31.38	0.93	0.07	23.88	0.91	0.10	31.61	0.92	0.07
Ours	22.98	0.88	0.09	24.23	0.84	0.07	35.24	0.97	0.03	35.04	0.98	0.02	34.36	0.97	0.03

Table 3: Quantitative comparison on the Photo Tourism dataset. The best and second-best results in each column are highlighted with red and orange cell backgrounds, respectively.

Method	*Brandenburg Gate*			*Sacre Coeur*			*Trevi Fountain*		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
DeSplat (Wang et al., 2025b)	25.04	0.920	0.142	20.14	0.868	0.178	23.31	0.775	0.226
WildGaussians (Kulhanek et al., 2024)	27.76	0.927	0.133	22.56	0.859	0.177	23.63	0.766	0.228
NeRF-W (Martin-Brualla et al., 2021)	24.22	0.884	0.213	19.56	0.795	0.260	21.48	0.693	0.331
K-Planes (Fridovich-Keil et al., 2023)	23.65	0.859	0.253	19.96	0.762	0.299	19.70	0.662	0.388
NeRF-HuGS (Chen et al., 2024)	27.17	0.929	0.083	22.23	0.862	0.124	23.41	0.788	0.165
EvidentialNeRF (Duan & Wong, 2026)	22.82	0.828	0.331	19.86	0.743	0.382	20.41	0.711	0.404
Ours	27.42	0.934	0.114	23.26	0.876	0.152	23.38	0.764	0.245

Table 4: Ablation study of each component on the NeRF On-the-go dataset. The best and second-best results are highlighted with red and orange cell backgrounds, respectively.

	Distractor field	$\mathcal{L}_{\text{BE}}$	$\mathcal{L}_{\text{DCS}}$	$\mathcal{L}_{\text{occ}}$	PSNR↑	SSIM↑	LPIPS↓
Baseline	✗	✗	✗	✗	17.267	0.5304	0.4085
0	✓	✗	✗	✗	21.657	0.7739	0.1481
1	✓	✓	✗	✗	22.566	0.7727	0.1471
2	✓	✗	✓	✗	21.908	0.7735	0.1482
3	✓	✗	✗	✓	21.785	0.7751	0.1466
Full	✓	✓	✓	✓	22.748	0.7769	0.1439

Combining all three regularizers achieves the best results on all metrics. More details and ablation results are provided in the appendix.

5 Conclusions

We presented S^{2}D-NeRF, a decomposition-based method for distractor-free NVS that represents the static scene with a shared 3D field and distractors with per-image 2D fields in pixel space. We further analyzed why this decomposition succeeds and attributed its success to the ARP. Driven by ARP, S^{2}D-NeRF separates diverse transient content, including shadows, based on view consistency, without relying on pretrained models or heuristic strategies. Additional regularization further improves the decomposition. Experiments on three datasets show competitive or superior rendering quality compared with state-of-the-art methods.

Limitations. Our decomposition relies on multi-view consistency to distinguish static content from distractors, which makes sparse-view settings challenging when cross-view evidence is insufficient. We leave its extension to sparse-view scenarios for future work.

ADDITIONAL STATEMENTS

AI use statement. We used a generative AI assistant (Claude, accessed through Claude Code) during this project. The assistant was used to help configure the environment, download and run official baseline methods, implement and edit parts of the codebase, organize dataset files, write automated experiment scripts, prepare scripts for rendering videos, identify related literature, and improve the readability of the manuscript. It was also used to draft and polish portions of the supplementary material and related work section. The static scene and the two multi-view images in the left panel of Figure 1 were generated with an AI image generation tool; we present this panel as a schematic illustration for ease of understanding, and these images are not used in any experiment. We did not use generative AI tools to generate synthetic datasets, formulate the research question, design the proposed representation or regularizers, or produce mathematical claims. All core ideas, experimental decisions, and final interpretations were made by the authors. We have reviewed all AI-assisted work. Every number reported in this paper was produced by our own training and evaluation code; the assistant was given read access to the resulting metric files but performed no measurement itself, and each table was checked against those files by the authors. All AI-drafted text and all references introduced with AI assistance were read and verified by the authors against the original publications. We take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

Ethics statement. This work involves no human subjects, no user study and no newly collected data. All experiments use publicly released benchmarks — NeRF On-the-go, RobustNeRF and Photo Tourism — under their original licenses and their official training/test splits; we redistribute none of the images. The authors declare no conflicts of interest.

Reproducibility statement. The method is specified in Sec. 3: Eq. (7) defines the compositing of the two fields, and Eq. (8), Eq. (9) and Eq. (10) define the three regularizers with their weights given in Sec. A.2. Table 5 lists, for every scene we report, the training/test split, the resolution we train at, and the calibration resolution the provided intrinsics refer to; Sec. A.1 explains the intrinsics rescaling this implies on NeRF On-the-go and the near plane and test-time appearance protocol we use on Photo Tourism. Table 6 gives the complete configuration in a single table, including optimizer, schedule, loss weights and both field architectures, and the paragraph that follows it lists the three settings that deviate per scene, so that no hyperparameter is left implicit. Sec. A.3 reports the parameter counts of both branches. Our implementation extends the public Zip-NeRF codebase, and we will release our code together with the configuration file of every run reported in the paper.

REFERENCES

Yanqi Bao, Jing Liao, Jing Huo, and Yang Gao. Distractor-free generalizable 3d gaussian splatting. In *Proc. Int. Conf. Learn. Represent. (ICLR)*, volume 2026, pp. 33344–33365, 2026.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 5470–5479, 2022.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 19640–19648, 2023.

Jiahao Chen, Yipeng Qin, Lingjie Liu, Jiangbo Lu, and Guanbin Li. NeRF-HuGS: Improved neural radiance fields in non-static scenes using heuristics-guided segmentation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 19436–19446, 2024.

Xingyu Chen, Qi Zhang, Xiaoyu Li, Yue Chen, Ying Feng, Xuan Wang, and Jue Wang. Hallucinated neural radiance fields in the wild. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 12943–12952, 2022.

Ruxiao Duan and Alex Wong. Evidential neural radiance fields. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 28632–28641, 2026.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-Planes: Explicit radiance fields in space, time, and appearance. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 12479–12488, 2023.

Chuanyu Fu, Yuqi Zhang, Kunbin Yao, Guanying Chen, Yuan Xiong, Chuan Huang, Shuguang Cui, and Xiaochun Cao. RobustSplat: Decoupling densification and dynamics for transient-free 3DGS. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2025.

Lily Goli, Cody Reading, Silvia Sellán, Alec Jacobson, and Andrea Tagliasacchi. Bayes' Rays: Uncertainty quantification for neural radiance fields. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 20061–20070, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 4015–4026, 2023.

Jonas Kulhanek and Torsten Sattler. NerfBaselines: Consistent and reproducible evaluation of novel view synthesis methods. In *Proc. of Adv. Neural Inf. Process. Syst. (NeurIPS)*, volume 38, pp. 127378–127403, 2025.

Jonas Kulhanek, Songyou Peng, Zuzana Kukelova, Marc Pollefeys, and Torsten Sattler. WildGaussians: 3D gaussian splatting in the wild. In *Proc. of Adv. Neural Inf. Process. Syst. (NeurIPS)*, volume 37, pp. 21271–21288, 2024.

Geng Lin, Chen Gao, Jia-Bin Huang, Changil Kim, Yipeng Wang, Matthias Zwicker, and Ayush Saraf. OmnimatteRF: Robust omnimatte with 3D background modeling. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 23471–23480, 2023.

Jingyu Lin, Jiaqi Gu, Lubin Fan, Bojian Wu, Yujing Lou, Renjie Chen, Ligang Liu, and Jieping Ye. HybridGS: Decoupling transients and statics with 2d and 3d gaussian splatting. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 788–797, 2025.

Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 7210–7219, 2021.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proc. of Eur. Conf. Comput. Vis. (ECCV)*, pp. 405–421, 2020.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):1–15, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russ Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael G. Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Trans. Mach. Learn. Res.*, 2024.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *Proc. Int. Conf. Learn. Represent. (ICLR)*, volume 2025, pp. 28085–28128, 2025.

Weining Ren, Zihan Zhu, Boyang Sun, Jiaqi Chen, Marc Pollefeys, and Songyou Peng. NeRF On-the-go: Exploiting uncertainty for distractor-free NeRFs in the wild. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 8931–8940, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 10674–10685, 2022.

Sara Sabour, Suhani Vora, Daniel Duckworth, Ivan Krasin, David J Fleet, and Andrea Tagliasacchi. RobustNeRF: Ignoring distractors with robust losses. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 20626–20636, 2023.

Sara Sabour, Lily Goli, George Kopanas, Mark Matthews, Dmitry Lagun, Leonidas Guibas, Alec Jacobson, David Fleet, and Andrea Tagliasacchi. SpotLessSplats: Ignoring distractors in 3D gaussian splatting. *ACM Trans. Graph.*, 44(2):1–11, 2025.

Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3D. *ACM Trans. Graph.*, 25(3):835–846, 2006.

Jiadong Tang, Yu Gao, Dianyi Yang, Liqi Yan, Yufeng Yue, and Yi Yang. Dronesplat: 3D gaussian splatting for robust 3d reconstruction from in-the-wild drone imagery. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 833–843, 2025.

Vadim Tschernezki, Diane Larlus, and Andrea Vedaldi. NeuralDiff: Segmenting 3D objects that move in egocentric videos. In *Proc. Int. Conf. 3D Vis. (3DV)*, pp. 910–919, 2021.

Rui Wang, Quentin Lohmeyer, Mirko Meboldt, and Siyu Tang. DeGauss: Dynamic-static decomposition with gaussian splatting for distractor-free 3D reconstruction. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2025a.

Yihao Wang, Marcus Klasson, Matias Turkulainen, Shuzhe Wang, Juho Kannala, and Arno Solin. DeSplat: Decomposed gaussian splatting for distractor-free rendering. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 722–732, 2025b.

Yukun Wang, Kunhong Li, Minglin Chen, Longguang Wang, Shunbo Zhou, Kaiwen Xue, and Yulan Guo. Distractor-free novel view synthesis via exploiting memorization effect in optimization. In *Proc. of Eur. Conf. Comput. Vis. (ECCV)*, pp. 477–493, 2024.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004.

Jiahao Wu, Rui Peng, Zhiyan Wang, Lu Xiao, Luyang Tang, Jinbo Yan, Kaiqiang Xiong, and Ronggang Wang. Swift4d: Adaptive divide-and-conquer gaussian splatting for compact and efficient reconstruction of dynamic scene. In *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025.

Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. D$\hat{2}$ nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. In *Proc. of Adv. Neural Inf. Process. Syst. (NeurIPS)*, volume 35, pp. 32653–32666, 2022.

Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 8254–8263, 2023.

Dongbin Zhang, Chuming Wang, Weitao Wang, Peihao Li, Minghan Qin, and Haoqian Wang. Gaussian in the wild: 3D gaussian splatting for unconstrained image collections. In *Proc. of Eur. Conf. Comput. Vis. (ECCV)*, 2024.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 586–595, 2018.

Supplementary Material

A Implementation Details

A.1 Datasets

Table 5 lists the scenes we use, the official training/test splits released with each dataset, and the resolution at which we train and evaluate.

NeRF On-the-go. We use the six scenes reported in Table 1; results on the remaining six scenes are shown in Sec. C.1. The released images are *not* an exact integer downsampling of the resolution at which the provided COLMAP model was calibrated: for *Spot*, for instance, the calibration resolution is 3964×2972 while the released images are 504×378, a ratio of 7.864 rather than 8. We scale the COLMAP intrinsics by this measured ratio, rather than by the nominal factor listed in Table 5, for every NeRF On-the-go scene, which keeps our training and evaluation resolution the same as that of the methods we compare against.

RobustNeRF On the RobustNeRF dataset, the released images are an exact eighth of the calibration resolution ($4032/504 = 3456/432 = 8$), so the nominal factor is used.

Photo Tourism. We use the three scenes and the training/test splits distributed by NerfBaselines (Kulhanek & Sattler, 2025) at their released resolution, *i.e.*, the original images downscaled by a factor of two, following the convention introduced by NeRF-W (Martin-Brualla et al., 2021) and used by the published results we compare against. Since the photographs are crowd-sourced, their resolutions and intrinsics vary across images; we therefore use each image at its native resolution with its corresponding intrinsics rather than assuming a common calibration. The default near plane of 0.2 clips a substantial portion of the reconstructed geometry, so we use 0.02 to avoid excessive clipping.

Appearance handling and test-time protocol on Photo Tourism. Images in Photo Tourism were captured over years under uncontrolled illumination, so a static radiance field cannot reproduce a given test image without knowing its appearance. Following NeRF-W (Martin-Brualla et al., 2021), we attach a 48-dimensional appearance embedding to each training image and concatenate it with the bottleneck vector of the static color MLP.

For testing views, we adopt the left/right protocol of NeRF-W (Martin-Brualla et al., 2021) as implemented by K-Planes (Fridovich-Keil et al., 2023). A new embedding is initialized at the mean of the training embeddings and optimized on the left half with all other parameters frozen; the reported metrics are computed on the unseen right half. Optimization uses Adam at a learning rate of 0.1 on shuffled left-half pixels in batches of 8192 rays for 60 epochs (one epoch is a single pass over the left-half pixels, 11–16 iterations), with the learning rate decayed by 0.1 every 18 epochs, and we keep the iterate with the lowest left-half loss. Rendering always uses the static branch only: no distractor field is instantiated for test images.

A.2 Hyperparameters

Table 6 lists the full configuration. A single setting is used for all scenes of all three datasets, with the exceptions noted below the table. We train for 25k iterations with 8192 rays per batch sampled uniformly from all training images, which takes about 40 minutes per NeRF On-the-go or RobustNeRF scene on one NVIDIA RTX 5090. The Photo Tourism scenes are an order of magnitude larger — up to 1689 training images at roughly four times the pixel count — and are trained for 100k iterations, about four hours per scene.

Two choices require further explanation. First, we enable both $\mathcal{L}_{\text{BE}}$ and $\mathcal{L}_{\text{DCS}}$ at iteration 5000 rather than from the start. Both act on α_d, which carries little information about distractors before the static field has formed; constraining it at that stage can lock in the arbitrary assignment formed during the first few hundred iterations. Second, batches are sampled across all training images rather than from a single image. Since the per-image distractor fields are independent, a single-image batch provides no cross-view signal to the static field, weakening the ARP that drives the decomposition (Sec. 3.2).

Table 5: Scenes, splits and resolutions. "Calibration" is the image size the provided COLMAP intrinsics refer to, and "Factor" is the nominal downsampling factor relating it to the resolution we render at. On NeRF On-the-go the true quotient deviates slightly from this integer, and we scale the intrinsics by the measured value rather than by the nominal factor (Sec. A.1). Photo Tourism photographs differ from one another in size and in intrinsics, and the factor relates the released resolution to the original photographs.

Dataset	Scene	#Train	#Test	Resolution	Calibration	Factor
NeRF On-the-go	*Mountain*	119	12	504×378	3944×2958	8
	Fountain	168	17	504×378	3953×2964	8
	Corner	101	20	504×378	3966×2974	8
	Patio	99	26	480×270	1908×1072	4
	Spot	168	10	504×378	3964×2972	8
	Patio-High	222	45	504×378	3967×2975	8
RobustNeRF	*Statue*	254	19	504×378	4032×3024	8
	Android	122	19	504×378	4032×3024	8
	BabyYoda	109	202	432×432	3456×3456	8
	Crab (1)	72	71	432×432	3456×3456	8
	Crab (2)	109	194	432×432	3456×3456	8
Photo Tourism	*Brandenburg Gate*	763	10	per image	per image	2
	Sacre Coeur	830	21			
	Trevi Fountain	1689	19			

Per-scene deviations. One setting departs from Table 6: $\mathcal{L}_{\text{occ}}$ is enabled only on *Spot* and *Crab (1)*. In both captures the distractors sit almost on the camera plane, and the capture trajectory leaves those near-camera positions observed by few other views, so the static field receives little cross-view supervision there and represents the distractors as floaters directly in front of the camera; $\mathcal{L}_{\text{occ}}$ suppresses these floaters. The term acts only within a thin band in front of the camera and leaves the main structure of the scene untouched, but inside that band it suppresses genuine static content as readily as floaters, so we enable it only where the capture makes it necessary. This situation is uncommon: among the twenty scenes we evaluate, only these two exhibit it.

A.3 Architecture

Static branch. The static field follows Zip-NeRF (Barron et al., 2023) without modification. Two proposal MLPs, each operating on a hash grid with a finest resolution of 512 and 2048, respectively, draw 64 samples per ray and are supervised by the anti-aliased interlevel loss. The final level draws 32 samples and queries a 10-level hash grid with 4 features per level, from resolution 16 to 8192 with a 2^{21}-entry table. Samples are multisampled conical frustums, distances are parameterized by the power transformation with $\lambda = -1.5$, and the background is opaque. Together, these components account for 77.6M shared parameters.

Distractor branch. The per-image fields are stored as a single tensor of shape $K \times E \times F$, where K is the number of training images, E is the total number of hash entries summed over the 16 levels, and $F = 2$. Image k therefore owns a private slice of $E \times F = 108{,}976$ parameters, and the whole branch adds 109K parameters per training image — 18.3M for *Spot* with its 168 training images, against the 77.6M of the static field. A pixel $\mathbf{p} = (u, v)$ is mapped to $[0, 1]^2$ and looked up bilinearly at all 16 resolutions. The concatenated 32-dimensional feature is decoded by a two-layer, 64-wide MLP shared across all images, whose four outputs are passed through a sigmoid and give the distractor color and the distractor opacity α_d in Eq. (7). At initialization α_d takes the same value $\alpha_0 = 0.018$ at every pixel. Only the hash tables are per-image; the decoder is shared across all images. A training batch mixes rays from many images, so the whole batch is served by one gather over (k, entry) pairs.

Inference. The distractor branch exists only for training images. We optimize the composite $\hat{\mathbf{C}}$ during training, while at test time we discard the distractor branch and render the static branch $\mathbf{C}_s$ alone. All reported results are therefore computed from $\mathbf{C}_s$.

Table 6: Hyperparameters. The same values are used for every scene of every dataset except where noted.

Optimization	
Iterations	25,000 (Photo Tourism: 100,000)
Rays per batch	8192, sampled from all training images
Optimizer	Adam, $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 10^{-15}$
Learning rate	$10^{-2} \rightarrow 10^{-3}$, log-linear
Learning-rate warm-up	5000 steps, starting at 10^{-8} of the rate
Hash-grid weight decay	0.1
Near / far plane	0.2 / 10^6 (Photo Tourism: 0.02 / 10^6)
Loss weights	
Photometric	Charbonnier, padding 10^{-3}, weight 1.0
Anti-aliased interlevel	0.01
Distortion	0.005
$\mathcal{L}_{\text{BE}}$	$\lambda_{\text{BE}} = 0.01$, from iteration 5000
$\mathcal{L}_{\text{DCS}}$	$\lambda_{\text{DCS}} = 0.1$, $\tau = 0.04$, $\eta = 0.003$, from iteration 5000
$\mathcal{L}_{\text{occ}}$	$\lambda_{\text{occ}} = 0.1$, $\rho_{\text{occ}} = 0.15$ (*Spot* and *Crab (1)* only; 0 elsewhere)
Distractor field (per training image)	
Hash levels L / features F	16 / 2
Resolution	$17 \rightarrow 7008$, per-level scale 1.5
Table size $\log_2 T$	12 (108,976 parameters per image)
Table initialization	$\mathcal{N}(0, 10^{-4})$
Decoder F_d	2×64 MLP with ReLU, 4 outputs
Initial opacity α_0	0.018 at every pixel
Static field (Zip-NeRF)	
Sampling levels	2 proposal (64 samples each) +1 NeRF (32 samples)
Proposal grids	finest resolution 512 and 2048, 1 feature per level
NeRF grid	10 levels, 4 features per level, $16 \rightarrow 8192$, $\log_2 T = 21$
Density MLP	1×64, bottleneck 256
Color MLP	2×256, viewing direction by positional encoding (degree 4)
Ray parameterization	power transformation, $\lambda = -1.5$, opaque background
Appearance embedding	none (Photo Tourism: 48-d, concatenated into the color MLP)

B Additional Ablation Results

This section first presents qualitative ablation results that verify the effectiveness of each component of our method (Sec. B.1). We then examine two hyperparameters of the distractor branch that are fixed in the main paper without further discussion: the opacity at which the per-image fields are initialized (Sec. B.2) and the size of their hash tables (Sec. B.3). Both are ablated on NeRF On-the-go with every other setting held at the values of Table 6. Finally, we report an experiment in which the distractor Gaussians of DeSplat (Wang et al., 2025b) are given the same per-pixel setting as our per-image field (Sec. B.4).

B.1 Qualitative Ablation of Component Effectiveness

Figures 6 and 7 demonstrate the effectiveness of each component of our method. As shown in Fig. 6, after introducing the distractor field, the model can more clearly separate dynamic distractors from the static scene, thereby reducing their impact on static scene reconstruction. On this basis, adding the regularization term $\mathcal{L}_{\text{BE}}$ prevents the model from excessively assigning illumination changes, local details, and reflective regions to the distractor branch. It also leads to higher distractor opacity, indicating a stronger occlusion effect on the static scene. This effect is more evident in the visualization of α_d in Fig. 7: with $\mathcal{L}_{\text{BE}}$, the distractor mask is more concentrated on true distractor regions and does not cover excessive static scene content. We further compute the PSNR of the distractor-free region marked by the red box in the lower-left corner of Fig. 7 and report it on the



Figure 6: Qualitative ablation results of different components in our method.

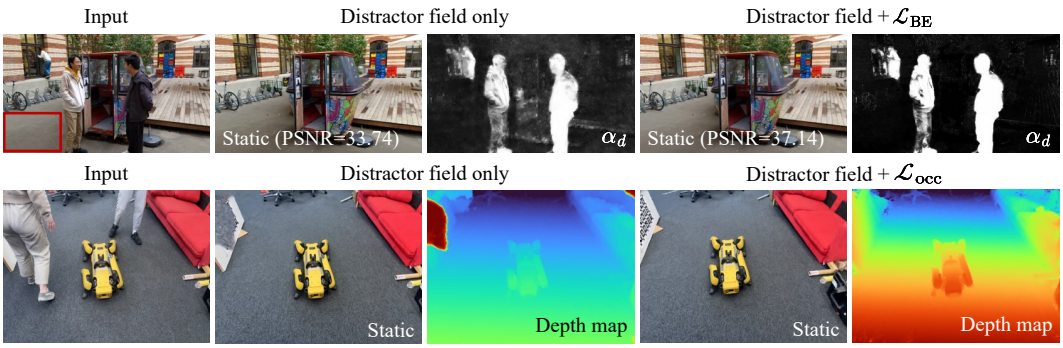


Figure 7: Visualization of distractor opacity α_d and depth maps under different ablation settings.

corresponding rendered images. The results show that adding the regularization term $\mathcal{L}$BE produces rendered colors that are closer to the ground truth.

In addition, $\mathcal{L}_{\text{DCS}}$ is used to penalize cases where the distractor color is too similar to the background color, thereby reducing regions that are explained only by the distractor branch. As shown in Fig. 6, after adding this term, the reconstructed static background becomes clearer, and the scene structure is more stable. Finally, $\mathcal{L}_{\text{occ}}$ is used to penalize floater artifacts close to the camera. The depth maps in Fig. 7 further show that after adding $\mathcal{L}_{\text{occ}}$, foreground floaters are significantly reduced and the depth estimation becomes more accurate.

B.2 Initial Distractor Opacity

We define α_0 as the initial opacity of all pixels before optimization. Table 7 compares our default $\alpha_0 = 0.018$ (nearly transparent) against $\alpha_0 = 0.5$ and $\alpha_0 = 0.993$ (nearly opaque). It can be observed that our default value is the only safe setting across all six scenes. Starting at $\alpha_0 = 0.5$ matches it on five of them — it is even 0.12 dB ahead on *Corner* — but loses 5.37 dB on *Patio*. Starting at $\alpha_0 = 0.993$ loses more than 3 dB on four of the six, and the two worst cases, *Corner* and *Spot*, fall to 5.41 and 8.74 dB.

Figure 8 illustrates this failure mode. When the per-image field is already opaque at initialization, it can reproduce the composite color $\hat{\mathbf{C}}$ by itself, leaving almost no photometric residual for the static branch. As a result, on the *Spot* scene, the static field fails to form, while the opacity map α_d remains saturated over the entire image. Since the decomposition is largely determined within the first few thousand iterations, an opaque initialization can bias the model toward an incorrect solution. In contrast, a nearly transparent initialization leaves the residual to the static field, which can reduce it consistently across multiple views (Sec. 3.2). Whether a scene can recover from an opaque initialization depends on how much of its content is sufficiently cross-view consistent for the static field to reclaim, explaining why the effect varies across scenes. Setting $\alpha_0 = 0.018$ avoids this issue.

Table 7: Initial distractor opacity α_0 setting. Per-scene values are PSNR; the last three columns average all six scenes. Every other setting follows Table 6. The best value in each column is highlighted.

α_0	PSNR↑ per scene						Mean over six scenes		
	Patio	*Fountain*	*Mountain*	*Corner*	*Spot*	*Patio-High*	PSNR↑	SSIM↑	LPIPS↓
0.018	22.06	21.59	21.50	25.13	24.10	22.11	22.75	0.777	0.144
0.500	16.69	21.49	21.42	25.25	24.04	21.96	21.81	0.757	0.163
0.993	21.96	17.67	15.52	5.41	8.74	21.20	15.08	0.541	0.486

B.3 Capacity of the Per-Image Hash Grid

We first examine the fitting capacity of a single per-image field on its own. We optimize one field with the default configuration ($T = 12$) on one training image of *Spot*, using only the photometric loss and no static field. As shown in Fig. 9, it reaches 35.1 dB PSNR after 1000 iterations and 38.3 dB after 20,000, reproducing both the pedestrians and the fine texture of the ground. The per-image field therefore converges quickly and is expressive enough to capture the distractors of an image in full detail.

Each per-image field stores 16 levels of two-dimensional grids at increasing resolutions, with each level stored in a hash table capped at 2^T entries. This capacity controls how much image-specific detail can be represented by the private field. Our default setting, $T = 12$, uses 108,976 parameters per training image (Sec. A); in comparison, $T = 10$ and $T = 13$ use 30,528 and 204,352 parameters, respectively. Table 8 reports a sweep over T on three scenes, one selected from each occlusion level in Table 1.

Between 2^{10} and 2^{12} the three-scene means move by 0.12 dB in PSNR, 0.002 in SSIM and 0.005 in LPIPS, so the setting is not critical over that range and we keep the middle value. The largest table is the worst setting on all three means, and its LPIPS is the worst of the four on each of the three scenes. This is the behavior the ARP (Sec. 3.2) predicts: the per-image fields are held back from cross-view-consistent content by how expensive it is for them to relearn it in every image, and enlarging them makes that content cheaper for the wrong branch to absorb. Fig. 10 shows this directly. The static branch is visually unchanged across the whole sweep, which is what the flat numbers say, but the distractor field is not: as the table grows it reproduces more of the static scene, and at 2^{13} on *Corner* it carries the easel, the warning sign and the shelf edges, none of which move between views. Capacity therefore does not need to be tuned upward, and doing so is mildly harmful.

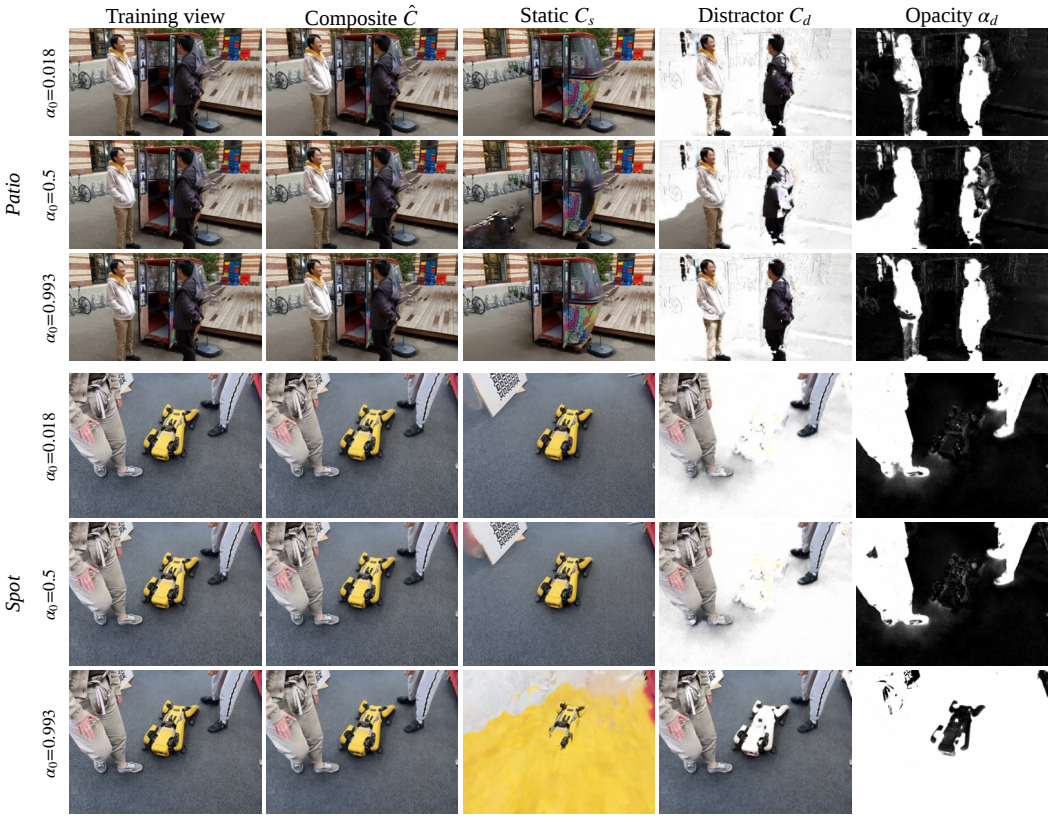


Figure 8: Effect of the initial distractor opacity α_0 on the decomposition, shown on one training view of *Patio* and *Spot*.

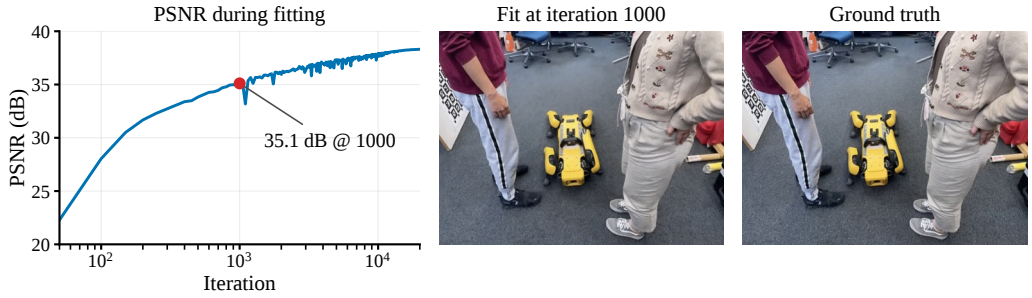


Figure 9: Fitting capacity of a single per-image field. One field with the default configuration ($T = 12$) is optimized alone on one training image of *Spot*. **Left:** PSNR against iteration. **Middle:** the fit after 1000 iterations (35.1 dB). **Right:** the training image.

B.4 Per-Pixel Gaussian Distractors

DeSplat (Wang et al., 2025b) assigns each training view its own set of distractor Gaussians. By default, it initializes 1000 of them per view at random in front of the camera, each with an opacity of 0.1. Their number is then controlled by threshold-based rules: a distractor Gaussian is cloned or split when its average screen-space positional gradient exceeds a threshold, and pruned when its opacity falls below 0.1. Our per-image field is instead defined at every pixel. To test whether DeSplat benefits from the same dense coverage, we replace its random initialization on *Spot* with one Gaussian per pixel, back-projected through the camera intrinsics to the same depth and scaled to the footprint of one pixel. This gives 182,780 Gaussians per view. We keep the initial opacity at 0.1 instead of a value close to zero as in our field ($\alpha_0 = 0.018$). Because the pruning threshold is also 0.1, any lower initial opacity would delete every distractor Gaussian at the first pruning step,

Table 8: Size of the per-image hash tables. T is the per-level entry cap 2^T and "Params" the resulting number of parameters per training image. $\mathcal{L}_{\text{occ}}$ is enabled on *Spot* only, as in the main results.

T	Params	*Mountain*			*Corner*			*Spot*			Mean		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
10	30,528	21.90	0.744	0.143	25.07	0.836	0.108	23.92	0.797	0.121	23.63	0.792	0.124
11	57,856	21.86	0.742	0.148	25.10	0.839	0.107	24.12	0.800	0.120	23.69	0.794	0.125
12	108,976	21.50	0.738	0.157	25.13	0.841	0.106	24.10	0.802	0.123	23.57	0.794	0.129
13	204,352	21.66	0.731	0.174	24.94	0.842	0.115	23.72	0.797	0.133	23.44	0.790	0.141

and the distractor branch would vanish. We evaluate three variants: with the default densification and pruning, without them, and without them while freezing the positions, scales and rotations so that only the color and opacity of each pixel are learned. The frozen variant is the closest Gaussian counterpart of our per-pixel field. All variants otherwise follow the official configuration.

Table 9: Per-pixel initialization of the DeSplat distractor Gaussians on *Spot*. All DeSplat rows are trained and evaluated with the official DeSplat code (LPIPS with AlexNet). "Ours" is our result in Table 1, for which "Transparent" is computed from α_d. "Init" and "Final" are the numbers of distractor Gaussians per view before and after training (mean over views). "Transparent" is the fraction of training pixels whose accumulated distractor opacity is below 0.05 after training.

Initialization	Densification	Init	Final	PSNR↑	SSIM↑	LPIPS↓	Transparent
Random (default)	✓	1000	1397	24.73	0.786	0.147	41.4%
Random	✓	10,000	3170	24.69	0.789	0.142	40.6%
Per-pixel	✓	182,780	28,171	22.40	0.712	0.304	25.9%
Per-pixel		182,780	182,780	20.58	0.663	0.380	3.2%
Per-pixel, frozen geometry		182,780	182,780	20.72	0.693	0.268	0.0%

Table 9 shows that all three per-pixel variants fall behind the default by 2.3 to 4.2 dB. With the per-pixel initialization, about $28{,}000$ Gaussians per view survive density control and all $182{,}780$ remain without it, compared with about 1400 in the default setting. The fraction of transparent pixels drops from 41% to 26%, 3% and 0%, so the distractor layer stays partially opaque over most of the image and hides part of the static scene in every view. Making the distractor Gaussians of DeSplat per-pixel therefore does not improve the results but degrades them.

C Additional Visualization Results

This section collects the qualitative results referred to in Sec. 4: the decomposition produced by our method on the six NeRF On-the-go scenes that are not reported in the main paper (Sec. C.1), and a comparison of the recovered static scene against WildGaussians (Kulhanek et al., 2024) on Photo Tourism (Sec. C.2).

C.1 Remaining NeRF On-the-go Scenes

The NeRF On-the-go dataset contains twelve sequences, among which the six reported in Table 1 are those commonly used in prior work. For the remaining six sequences, namely *Arc de Triomphe*, *Drone*, *Statue*, *Train*, *Train Station*, and *Tree*, we directly visualize the learned decomposition in Fig. 11. For one training view from each scene, Fig. 11 shows the input image, our composite prediction $\hat{\mathbf{C}}$, the static field $\mathbf{C}_s$, the opacity-weighted distractor color $\alpha_d \mathbf{C}_d$, and the distractor opacity α_d.

The occluders differ in kind across the six sequences: a parked car in *Arc de Triomphe*, pedestrians in *Statue*, *Drone*, *Train* and *Tree*, and a tram in *Train Station*. In all cases, the composite prediction reproduces the training image, while the static field renders the scene without the occluder and the distractor field captures the occluding content, with α_d concentrated on the occluder and close to zero on the surrounding static regions. The static field does more than simply inpaint the occluded area. For example, in *Arc de Triomphe*, it reconstructs the building behind the car, and in *Train Station*, it recovers the building behind the tram. These structures are not visible in the corresponding

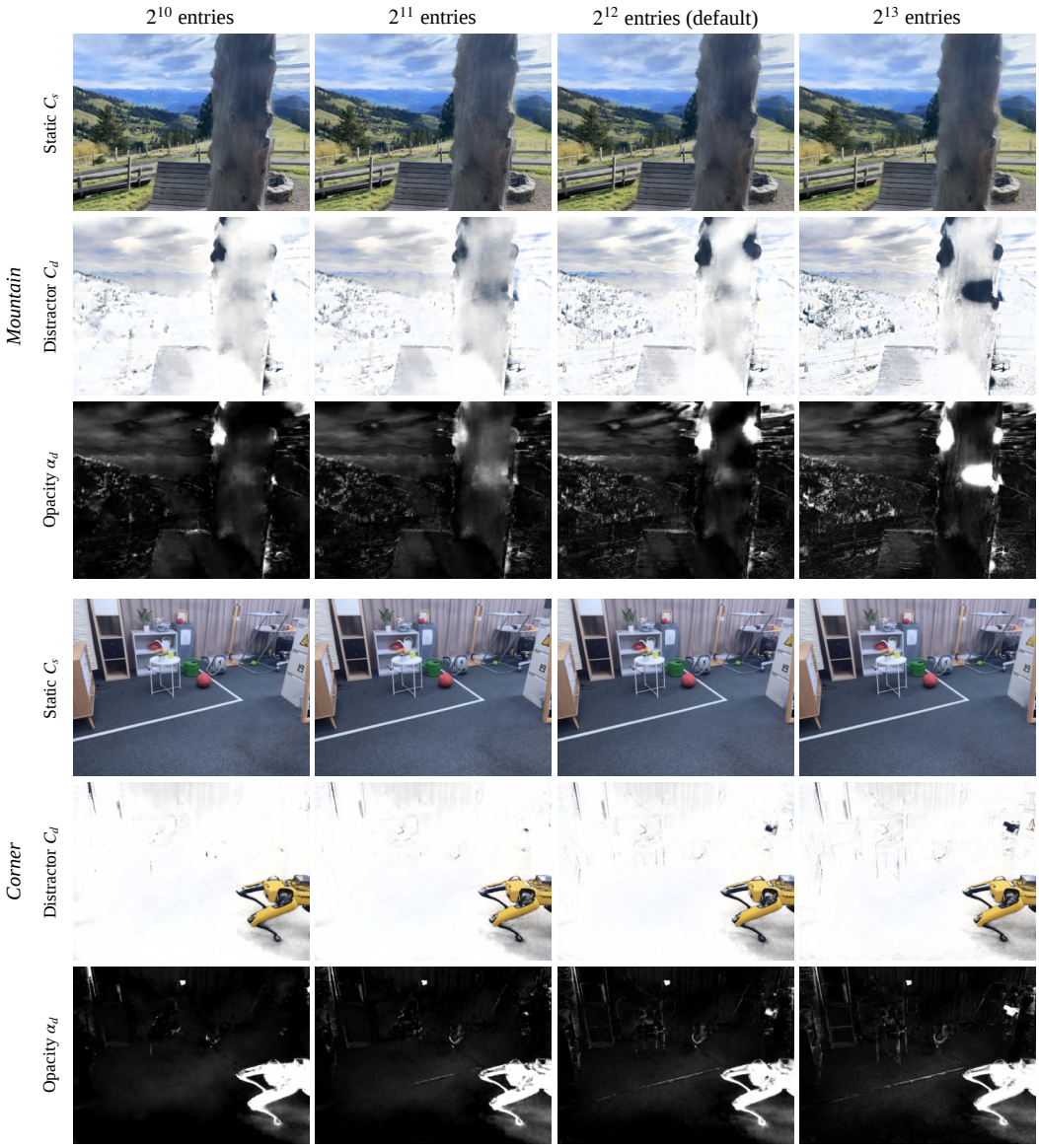


Figure 10: Effect of the per-image table size on the decomposition, shown on one training view of *Mountain* and *Corner*. Columns sweep the per-level entry cap 2^T; rows are the static branch $\mathbf{C}_s$, the distractor color $\mathbf{C}_d$ and the distractor opacity α_d. The static branch is visually unchanged across the sweep, while the distractor field takes on progressively more static structure: at 2^{13} it reproduces the easel, the warning sign and the shelf edges of *Corner*, and the fence and bench of *Mountain*.

training images and can only be inferred from other views. This further clarifies the basis of the separation: the two branches are distinguished by cross-view consistency. Therefore, any content that cannot be explained consistently across views, including global appearance changes such as haze or moving cloud cover, is also routed to the per-image branch.

C.2 Comparison on Photo Tourism

Fig. 12 compares the static scene recovered by our method with that of WildGaussians (Kulhanek et al., 2024), using the best available renderings directly downloaded from NerfBaselines (Kulhanek & Sattler, 2025). As shown in the figure, in the *Brandenburg Gate* and *Sacre Coeur* scenes, our method removes occluders more effectively, recovers the original scene colors, and preserves high-



Figure 11: Qualitative results on the remaining six NeRF On-the-go scenes. Each row shows one training view of one scene. From left to right: the training image, our composite rendering $\hat{\mathbf{C}}$, the static field $\mathbf{C}_s$, the opacity-weighted distractor color $\alpha_d \, \mathbf{C}_d$, and the distractor opacity α_d.

frequency structures. For the *Trevi Fountain* scene, WildGaussians produces blurred details, whereas our method better removes black artifacts and recovers more details.

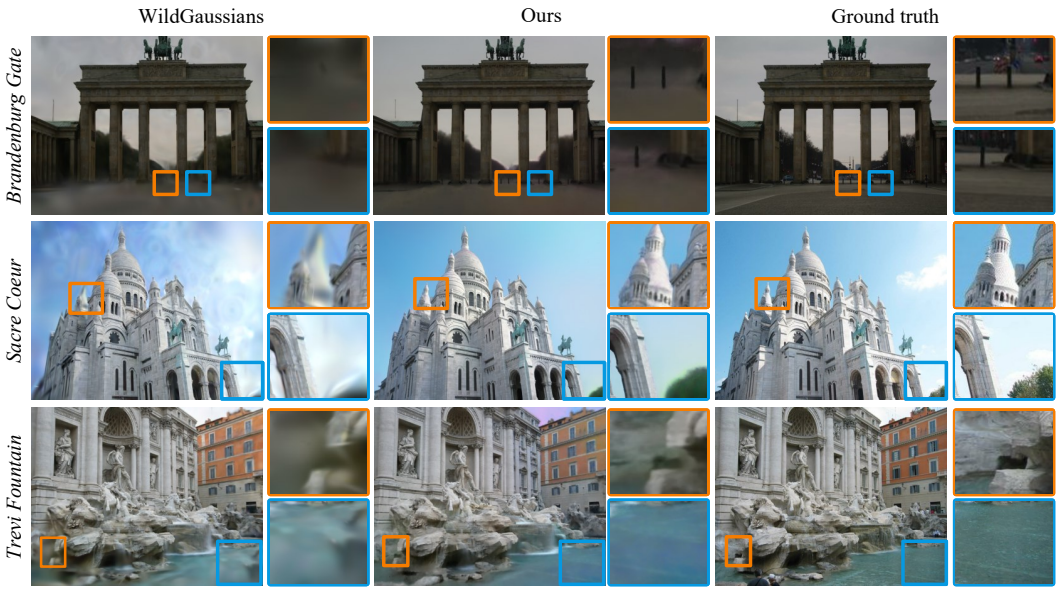


Figure 12: Qualitative comparison with WildGaussians (Kulhanek et al., 2024) on the Photo Tourism dataset.