EvE: Exploiting Generative Priors for Radiance Field Enrichment

Karim Kassab, Antoine Schnepf, Jean-Yves Franceschi, Laurent Caraffa, Jeremie Mary, Valérie Gouet-Brunet

EvE: Exploiting Generative Priors for Radiance Field Enrichment

Karim Kassab[1,2] Antoine Schnepf[1,3] Jean-Yves Franceschi[1]
Laurent Caraffa[2] Jeremie Mary[1] Valérie Gouet-Brunet[2]

[1] Criteo AI Lab, Paris, France
[2] LASTIG, Université Gustave Eiffel, IGN-ENSG, F-94160 Saint-Mandé
[3] Université Côte d'Azur, CNRS, I3S, France

Abstract

Modeling large-scale scenes from unconstrained image collections in-the-wild has proven to be a major challenge in computer vision. Existing methods tackling in-the-wild neural rendering operate in a closed-world setting, where knowledge is limited to a scene's captured images within a training set. We propose EvE, which is, to the best of our knowledge, the first method leveraging generative priors to improve in-the-wild scene modeling. We employ pre-trained generative networks to enrich K-Planes representations with extrinsic knowledge. To this end, we define an alternating training procedure to conduct optimization guidance of K-Planes trained on the training set. We carry out extensive experiments and verify the merit of our method on synthetic data as well as real tourism photo collections. EvE enhances rendered scenes with richer details and outperforms the state of the art on the task of novel view synthesis in-the-wild. Our project page can be found at https://eve-nvs.github.io.

1. Introduction

To draw novel objects and views, humans often rely on a blend of **cognition** and **intuition**, where the latter is built on a large prior acquired from a long-term continuous exploration of the visual world. Nevertheless, ablating one of these two elements results in catastrophic representations. On the one hand, humans find it particularly difficult to draw a bicycle based solely on this preconceived prior [14]. However, once one photograph is observed, drawing novel views of a bicycle becomes straightforward. On the other hand, drawing monuments and complex objects based solely on observed images, and with no preconceived notions of geometry and physics, is also non-trivial. In computer vision, recent methods tackling neural rendering, object generation, and novel view synthesis are exclusively built on either the former or the latter ablation.

(a) K-Planes (b) EvE (c) GT

Figure 1. **Qualitative Results.** Given images of the Trevi fountain from Phototourism [21], as well as a pre-trained generative model [36], our method leverages the pre-trained model and enriches K-Planes with more details and information that are under-represented when optimizing the same K-Planes on the images alone.

	No 3D supervision	Pre-trained prior	Geometric consistency	In-the-wild scene modeling	Underlying representation	Task
NeRF [29]	✓	✗	✓	✗	NeRF	NVS
K-Planes [12]	✓	✗	✓	✓	K-Planes	NVS
NFD [39]	✗	✗	✓	✗	Tri-Planes	OG
3DGen [17]	✗	✗	✓	✗	Tri-Planes	OG
Latent-NeRF [28]	✓	✓	✓	✗	NeRF	OG
DreamFusion [35]	✓	✓	✓	✗	NeRF	OG
3DiM [44]	✗	✗	✓	✗	—	NVS
NerfDiff [16]	✗	✓	✓	✗	Tri-Planes	NVS
DiffusioNeRF [46]	✓	✗	✓	✗	NeRF	NVS
RealFusion [27]	✓	✓	✓	✗	NeRF	NVS
Zero-1-to-3 [25]	✗	✓	✗	✗	—	NVS
NeRF-W [26]	✓	✗	✓	✓	NeRF	NVS-W
Ha-NeRF [7]	✓	✗	✓	✓	NeRF	NVS-W
CR-NeRF[47]	✓	✗	✓	✓	NeRF	NVS-W
EvE (ours)	✓	✓	✓	✓	K-Planes	NVS-W

Table 1. **Related work overview.** EvE leverages unconstrained tourist photo collections [21] and a pre-trained generative prior [36] to enrich K-Planes, our underlying scene representation, in order to achieve state-of-the-art performance in in-the-wild scene modeling.

The first class of methods approaches novel view synthesis by learning scenes through rigorous **cognition**, as in dense observations of captured images. Although classic methodologies like structure-from-motion [18] and image-based rendering [40] have previously tackled this problem, the field has recently seen substantial advancements thanks to neural rendering techniques. In particular, Neural Radiance Fields [29, NeRF] learn a scene by observing densely captured posed images from said scene and optimizing a neural network to replicate these images thanks to volume rendering. Chan et al. [4] propose a Tri-Plane representation enabling faster learning compared to NeRF. Recent methods extend NeRFs [7, 26, 47] and Tri-Planes [12] to support learning from unconstrained "in-the-wild" photo collections by enabling robustness against illumination variations and transient occluders. NeRFs and Tri-Planes, however, achieve no generalization across scenes as they are trained from scratch. This means that these representations are learned in a closed-world setting, where the information scope is limited to the training set at hand.

The second class of methods tackles novel view synthesis and object generation by learning and leveraging priors over images and scenes, reminiscent of drawing from insights and **intuition**. These methods have also recently witnessed accelerated advancements. Current works either train generative models to infer some implicit representation [17, 39], or leverage large-scale pre-trained models [36, 37] for Object Generation (OG) [28, 35] and Novel View Synthesis (NVS) [25, 27]. Liu et al. [25] achieve novel view synthesis from single images only by simply fine-tuning a pre-trained latent diffusion model [36]. This proves pivotal significance related to large-scale pre-trained vision models, as it shows that, although trained on 2D data, these models learn a rich geometric 3D prior about the visual world. Nevertheless, as these models alone have no explicit multi-view geometric constraints, they tend to suffer from geometric inconsistencies across views [25, 44].

EvE proposal. Our work builds on this discourse and aims to leverage both the high-fidelity geometrically consistent power of closed-world neural implicit representations and the rich prior lying within pre-trained generative models. In this paper, we introduce EvE, a method for *enriching* scene representations by leveraging generative priors. We present an alternating training procedure that iteratively switches between fine-tuning a large-scale pre-trained generative model to generate a K-Planes representation, and correcting this representation by optimizing it on a particular dataset. This guides the optimization of a particular scene, by leveraging not only the training dataset at hand but also the rich information prior lying within the weights of the pre-trained generative model. Note that a particular advantage of our method is its modularity, as the generative model is easily interchangeable with any other image generative model, hence enabling future-proofing.

We carry out extensive experiments and conduct quantitative and qualitative evaluations of EvE on synthetic datasets as well as challenging in-the-wild photo collections of cultural landmarks. For synthetic scenes, we outperform our base representation and highlight the robustness of our method towards the restriction of training data, which proves the effectiveness of the generative model's prior. For in-the-wild datasets, we show superior performance compared to recent methods tackling novel view synthesis in the wild (NVS-W), including the representation we base our method on. Fig. 1 showcases the improvements our method demonstrates on the Trevi fountain scene from Phototourism [21]. To investigate further, we also present an ablation study of our method, which illustrates the value of the generative model's prior, as well as the value of its fine-tuning.

A summary of our contributions can be found below.

- We introduce EvE, which is, to the best of our knowledge, the first method leveraging generative priors for in-the-wild scene modeling.
- By conducting optimization guidance via an alternating training procedure, EvE further proves the value of large-scale pre-trained image generative models, particularly Stable Diffusion, in 3D tasks.
- EvE outperforms the state-of-the-art on the task of novel view synthesis in-the-wild. The code for EvE will be publicly available as open-source.

2. Related Work

EvE achieves geometrically consistent novel view synthesis in-the-wild by leveraging unconstrained image collections from Phototourism [21], and a pre-trained generative prior. Our method is the first to satisfy all of these attributes, as summarized in Tab. 1. In this section, we develop the various preceding works from which our method takes inspiration.

Neural rendering. Neural rendering [41] has seen significant advancements since the introduction of NeRF [29]. At its core, neural rendering blends approaches from classical computer graphics [8, 42, 45] and machine learning [13, 32, 33] to reconstruct scenes from real-world observations. NeRF learns a scene by overfitting the weights of a neural network on posed images of said scene. This subsequently enables the reconstruction of the scene thanks to volume rendering [22]. Chan et al. [4] introduce the Tri-Planes representation as a middle ground between implicit and explicit representations, enabling faster learning of scenes. More recently, Kerbl et al. [23] tackle real-time high-quality radiance field rendering on unbounded scenes. Note that rendering unbounded scenes is dissimilar to our task of in-the-wild rendering, where the latter consists of learning scenes from *unconstrained* internet photo collections of cultural landmarks, that are also unbounded, but additionally naturally plagued by illumination variations as well as transient occluders. This makes learning a scene particularly challenging, as surfaces can exhibit significant visual disparities across views.

Neural rendering in-the-wild. Subsequent to NeRFs, several techniques emerged to extend the NeRF setup to "in-the-wild" unconstrained photo collections plagued by illumination variations and transient occluders. NeRF-W [26] and Ha-NeRF [7] address the challenge of novel view synthesis in the wild (NVS-W) by separately modeling scene lighting through appearance embeddings, and transient occluders through transient embeddings. Fridovich-Keil et al. [12] present K-Planes, which modify and extend Tri-Plane representations [4] to in-the-wild scenes thanks to learnable appearance embeddings (similarly to [26]), as well as dynamic scenes. Contrary to our work, previous in-the-wild methods all learn scenes in a closed-world setup, where knowledge is limited to tourist photo collections [21].

Foundation models as priors. The integration of large-scale pre-trained models as priors for downstream tasks has emerged as a prominent trend, as they enable the effective incorporation of linguistic and semantic knowledge into diverse applications. Notably, denoising diffusion probabilistic models [19, 36] have recently gained particular attention, and have seen applications as plug-and-play priors [15] and utilized in various domains such as super-resolution [43] and more specifically novel view synthesis [25]. Our work takes another step in this direction, and uses these models to *enrich* existing representations learned from multiple images of the same scene.

Generative models in 3D. Generative models have seen many applications in 3D object generation and novel view synthesis. For object generation, Shue et al. [39] learn the distribution of Tri-Plane occupancy fields of 3D objects in ShapeNet [5], and subsequently generate new 3D objects by Tri-Planes diffusion. Poole et al. [35] leverage a pre-trained diffusion model (Imagen [37]), originally intended for 2D image generation, to train a NeRF modeling a novel object. This is done by prompting Imagen with natural language and training a NeRF from scratch for each caption. The NeRF's parameters are then updated by rendering a view, diffusing it, and then reconstructing it with Imagen. For NVS, Watson et al. [44] present a pose-conditional image-to-image diffusion model using a single view. They also introduce stochastic conditioning that improves 3D consistency compared to other generative models. Gu et al. [16] tackle single-image NVS by finetuning a NeRF representation using views synthesized from a jointly-trained diffusion model. Liu et al. [25] fine-tune Stable Diffusion [36], a pre-trained latent diffusion model for 2D images, to learn camera controls over a 3D dataset and thus performing NVS by generalizing to other objects. These results hold paramount value, as they highlight the rich 3D prior learned by Stable Diffusion, even though it has only been trained on 2D images. This however comes with the issue of geometric inconsistencies across views, as generative models alone have no explicit multi-view geometric constraints. As presented, generative models in 3D have been used to generate novel objects as well as novel views. However, the exploitation of pre-trained generative priors for direct in-the-wild scene augmentations remains an unexplored area of research. For an overview of recent methods, we refer the reader to Table 1.

3. Method

We learn a scene through two alternating stages, as illustrated in Fig. 2. *Scene fitting* optimizes our K-Planes representation $\mathbf{P}_\gamma$ to reproduce the images in the training set, as traditionally done in neural rendering techniques. *Scene enriching* finetunes a pre-trained generative prior to this K-Planes representation, and then infers a new one $\mathbf{P}_\varepsilon$, which will subsequently be corrected by scene fitting. The main idea behind this is that we use our 3D implicit model $\mathbf{P}_\gamma$ for optimizing the scene to the available information in the training set and adhere to essential geometric constraints, and a prior for enriching the scene by refining $\mathbf{P}_\gamma$ with inconspicuous details. In this section, we detail each stage and elucidate the intuition behind our method.

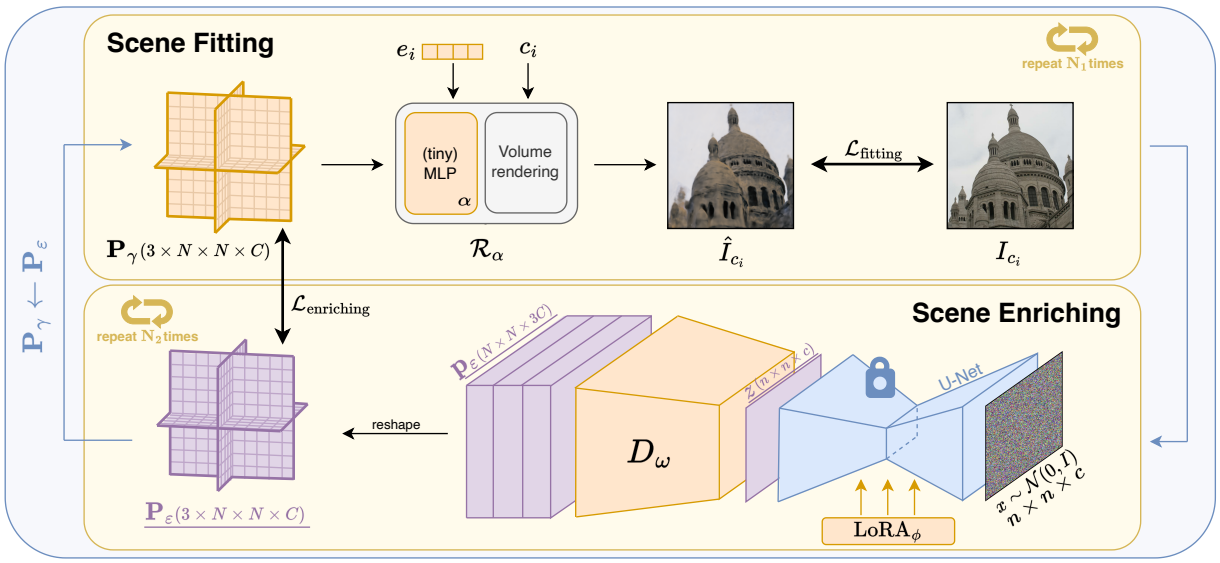


Figure 2. **Scene learning procedure.** The K-Planes $\mathbf{P}_\gamma$, the MLP with trainable parameters α, and the appearance embeddings e_i are learned during scene fitting. The LoRA parameters ϕ as well as the decoder D_w are learned during scene enriching. The pre-trained U-Net is frozen. Assets in violet and underlined are intermediate results. At each iteration, new planes $\mathbf{P}_\varepsilon$ are generated and assigned to $\mathbf{P}_\gamma$, which are then corrected by scene fitting.

3.1. Scene fitting

The goal at this stage is to fit a 3D scene, adhering to predefined geometric constraints, from posed 2D RGB images. To fit the scene, we adopt the K-Planes representation [12] (closely related to Tri-Planes [4] and HexPlanes [3]). As such, this stage corresponds to doing the optimization procedure of K-Planes representations as presented in [12], from which we adapt the code.

K-Planes are compact 3D model representations applicable to static scenes, "in-the-wild" scenes (scenes with varying appearances), and dynamic scenes. These models allow for fast training and rendering, while maintaining low-memory usage. K-Planes model a d-dimensional scene with $k = \binom{d}{2}$ planes, which represent the combinations of every pair of dimensions. This structure makes K-Planes compatible with a multitude of neural network architectures, and more particularly image-specialized network architectures. This enables K-Planes generation by minimally tweaking generative image architectures. For a static 3D scene, $k = 3$ and the planes represent the xy, xz, and yz planes. These planes, each of size $N \times N \times C$, encapsulate features representing the density and view-dependent colors of the scene.

The K-Planes model $\mathbf{P}_\gamma$ is originally randomly initialized. The first goal of scene fitting is then to *correct* this random initialization to fit the training set. Note that the first iteration of scene fitting is especially particular, since it is starting with a randomly initialized scene, as opposed to a *proposed* scene, as we describe in Sec. 3.2.

To render the 3D scene from K-Planes, as done in [12, 29], we cast rays from the desired camera position through the coordinate space of the scene, on which we sample 3D points. We decode the corresponding RGB color for each 3D point $\mathbf{q} = (i, j, k)$ by normalizing it to $[0, N)$ and projecting it onto the $k = 3$ planes, denoted as $\mathbf{P}_\gamma^{(xy)}, \mathbf{P}_\gamma^{(xz)}, \mathbf{P}_\gamma^{(yz)}$:

$$f(\mathbf{q})^{(h)} = \psi(\mathbf{P}_\gamma^{(h)}, \pi^{(h)}(\mathbf{q})) \,, \tag{1}$$

where $h \in \mathbf{H} = \{xy, xz, yz\}$, $\pi^{(h)}(\mathbf{q})$ projects $\mathbf{q}$ onto $\mathbf{P}_\gamma^{(h)}$, and ψ denotes bilinear interpolation on a regular 2D grid.

These features are then aggregated using the Hadamard product to produce a single feature vector of size M:

$$f(\mathbf{q}) = \prod_{h \in \mathbf{H}} f(\mathbf{q})^{(h)} \,. \tag{2}$$

To decode these features, we adopt the hybrid formulation of K-Planes [12]. Two small Multi-Layer Perceptrons (MLPs), g_σ and g_{RGB}, map the aggregated features as follows:

$$\begin{aligned} \sigma(\mathbf{q}), \hat{f}(\mathbf{q}) &= g_\sigma(f(\mathbf{q})) \,, \\ c(\mathbf{q}, \mathbf{d}) &= g_{\text{RGB}}(\hat{f}(\mathbf{q}), \gamma(\mathbf{d})) \,, \end{aligned} \tag{3}$$

where $\gamma(p) = (\sin(2^0 \pi p), \cos(2^0 \pi p), \ldots, \sin(2^{L-1} \pi p), \cos(2^{L-1} \pi p))$ is the positional embedding of p. g_σ maps the K-Planes features into density σ and additional features $\hat{f}$. Subsequently, g_{RGB} maps $\hat{f}$ and the positionally-encoded view directions $\gamma(\mathbf{d})$ into view-dependent RGB colors. This enforces densities to be independent of view directions.

These decoded RGB colors are then used to render the final image thanks to ray marching and integrals from classical volume rendering [22], that are practically estimated using quadrature:

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - \exp(-\sigma_i\delta_i))c_i \ , \\ T_i = \exp\left(-\sum_{j=1}^{i-1}\sigma_j\delta_j\right) \ , \tag{4}$$

where $\hat{C}(\mathbf{r})$ is the expected color, T_i is the accumulated transmittance along the ray, and $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples. Proposal sampling is also employed similarly to [12] to iteratively refine a small instance of K-Planes (using histogram loss [2]) that estimates densities along a ray. These density estimates are used to allocate more points in regions of higher densities.

3.2. Scene enriching

Given a fitted scene representation $\mathbf{P}_\gamma$, scene enriching learns this implicit representation and proposes a new *enriched* representation. To provide scene enriching with a strong prior from which it can complement a scene, we employ a large pre-trained latent diffusion model. To this end, we adopt Stable Diffusion [36, SD] for its proven rich 3D prior, as demonstrated by Liu et al. [25], as well as its proven performances as prior for downstream tasks, as showcased by Wang et al. [43]. Thus, we integrate the U-Net $\mathbf{SD}_\phi$ and the decoder $\mathbf{D}_\omega$ into our pipeline, and treat the K-Planes as $3C$-channel $N \times N$ images. We also replace the last layer of the decoder $\mathbf{D}_\omega$ with a randomly initialized convolutional layer (with no bias), to take into account the shape of the K-Planes. We then fine-tune the pre-trained model using the fitted K-Planes $\mathbf{P}_\gamma$, and generate enriched K-Planes $\mathbf{P}_\varepsilon = \mathbf{D}_w(\mathbf{SD}_\phi(x))$ where $x \sim \mathcal{N}(0, I)$.

To achieve this fine-tuning, a significant challenge presents itself: due to the sheer size of Stable Diffusion, it would be exceptionally costly to fine-tune all of its trainable parameters. Moreover, as we only want to modulate priors embedded into the pre-trained network, we look for an alternative to doing full fine-tuning. To circumvent these constraints, we adopt Low-Rank Adaptation [20, LoRA], a simple yet effective Parameter-Efficient Fine-Tuning (PEFT) method that has proven great transfer capabilities across modalities and tasks [10, 24, 48]. LoRA's relatively minimal design works directly over weight tensors, which means that it can be seamlessly applied to most model architectures. Furthermore, LoRA does not add any additional cost at inference, thanks to its structural re-parameterization (SR) design. To achieve this, Hu et al. [20] inject trainable low-rank decomposition matrices into each layer of a frozen pre-trained model. Let $\mathbf{W}_0$, $\mathbf{b}_0$ be the frozen pre-trained weights and biases, and x be the input. Fine-tuning a frozen linear layer $f(x) = \mathbf{W}_0 x + \mathbf{b}_0$ comes down to learning the low-rank decomposition weights $\Delta\mathbf{W} = \mathbf{BA}$:

$$\begin{aligned} f(x) &= \mathbf{W}_0 x + \Delta\mathbf{W}x + \mathbf{b}_0 \\ &= (\mathbf{W}_0 + \Delta\mathbf{W})x + \mathbf{b}_0 \\ &= \mathbf{W}_{\text{LoRA}}x + \mathbf{b}_0 \ , \end{aligned} \tag{5}$$

where $\mathbf{W}_0, \Delta\mathbf{W} \in \mathbb{R}^{d\times k}$; $\mathbf{B} \in \mathbb{R}^{d\times r}$; $\mathbf{A} \in \mathbb{R}^{r\times k}$; and the rank $r \ll \min(d, k)$.

Thus, to implement scene enriching, we fine-tune the LoRA parameters ϕ modulating the pre-trained U-Net, as well as the decoder's parameters ω, on the fitted scene $\mathbf{P}_\gamma$. Subsequently, we query the U-Net with gaussian noise x, decode its intermediary output latent representation z with $\mathbf{D}_\omega$, and generate $\mathbf{p}_\varepsilon$ that is reshaped into a new *enriched* scene $\mathbf{P}_\varepsilon$. This scene will then be proposed to the *scene fitting* stage and constitutes an improved initialization, which will, in its turn, correct it.

Algorithm 1: Alternating training algorithm.

Input: N_{epochs}, N_1, N_2, N, C, n, c, $\mathcal{I} = \{I_{c_i}, c_i\}$, $\mathcal{R}_\alpha$, $\mathbf{D}_w$, $\mathbf{SD}_\phi$, optimizer

$x \leftarrow$ standard-gaussian(n, n, c)
$\mathbf{P}_\gamma \leftarrow$ standard-gaussian$(N, N, 3C)$
for N_{epochs} *steps* **do**
  for N_1 *steps* **do** // scene fitting
    $\gamma, \alpha \leftarrow$ optimizer.step$(\mathcal{L}_{\text{fitting}}(\mathbf{P}_\gamma, \mathcal{I}))$
  for N_2 *steps* **do** // scene enriching
    $\mathbf{P}_\varepsilon \leftarrow \mathbf{D}_w(\mathbf{SD}_\phi(x))$
    $\omega, \phi \leftarrow$ optimizer.step$(\mathcal{L}_{\text{enriching}}(\mathbf{P}_\varepsilon, \mathbf{P}_\gamma))$
  $\mathbf{P}_\gamma \leftarrow \mathbf{P}_\varepsilon$

3.3. Training

We define an alternating training procedure rotating between scene fitting and scene enriching, as described above, and as illustrated in Fig. 2.

For *scene fitting*, we train the K-Planes model as proposed by Fridovich-Keil et al. [12]. We use spatial total variation regularization to encourage smooth gradients. This is applied over all the spatial dimensions of each plane in the representation:

$$\mathcal{L}_{\text{TV}}(\mathbf{P}) = \frac{1}{|C|N^2}\sum_{c,i,j}(\|\mathbf{P}_c^{i,j} - \mathbf{P}_c^{i-1,j}\|_2^2 \\ + \|\mathbf{P}_c^{i,j} - \mathbf{P}_c^{i,j-1}\|_2^2) \ . \tag{6}$$

For scenes with varying lighting conditions (*e.g. in-the-wild* scenes as in the Phototourism dataset [21]), an M-dimensional appearance vector e_i is additionally optimized for each image. This vector is then passed as input to the

Figure 3. **Qualitative results.** Results on three scenes from Phototourism [21]. EvE enriches K-Planes with richer and finer details, notably in less frequently captured areas (as tourists mostly tend to capture main facades).

	Brandenburg Gate		Sacré Coeur		Trevi Fountain	
	PSNR (↑)	SSIM (↑)	PSNR (↑)	SSIM (↑)	PSNR (↑)	SSIM (↑)
NeRF [29]	18.90	0.8159	15.60	0.7155	16.14	0.6007
NeRF-W [26]	24.17	<u>0.8905</u>	19.20	0.8076	18.97	0.6984
Ha-NeRF [7]	24.04	0.8773	20.02	0.8012	20.18	0.6908
CR-NeRF [47]	<u>26.53</u>	**0.9003**	<u>22.07</u>	**0.8233**	21.48	0.7117
K-Planes [12]	25.49	0.8785	20.61	0.7735	<u>22.67</u>	0.7139
K-Planes-SS [12]	24.48	0.8629	19.86	0.7419	21.30	0.6627
EvE-F (ours)	25.39	0.8834	21.41	0.8059	22.54	<u>0.7324</u>
EvE-P (ours)	25.42	0.8822	21.17	0.7978	22.16	0.7251
EvE (ours)	**26.64**	0.8869	**22.26**	<u>0.8176</u>	**23.42**	**0.7379**

Table 2. **Quantitative results.** Results on three real-world datasets from Phototourism [21]. The **bold** and <u>underlined</u> entries respectively indicate the best and second-best results.

MLP color decoder g_{RGB} at the rendering step $\mathcal{R}_\alpha$. Hence, the training objective for scene fitting is written as:

$$\min_{\alpha,\gamma} \mathcal{L}_{\text{fitting}} \triangleq \|\mathcal{R}_\alpha(\mathbf{P}_\gamma, \mathbf{C}) - I_{\mathbf{C}}\|_2^2 + \lambda_{\text{TV}} \mathcal{L}_{\text{TV}}(\mathbf{P}_\gamma)\,, \quad (7)$$

where $\mathcal{R}_\alpha$ represents the K-Planes rendering procedure (*i.e.* ray marching, feature decoding via a small MLP with trainable parameters α, and volume rendering), $\mathbf{P}_\gamma$ are the K-Planes with trainable parameters γ and $I_{\mathbf{C}}$ is a ground truth RGB image with camera position $\mathbf{C}$.

As for the *scene-enriching* phase, we optimize the decoder parameters w as well as the LoRA parameters ϕ, modulating the frozen U-Net weights, on the fitted scene $\mathbf{P}_\gamma$. These parameters are optimized to generate new K-Planes that will be corrected by scene fitting in the next phase. This allows the model to be both creative and geometrically self-consistent when modeling the scene. Thus, the fine-tuning objective for *scene enriching* is written as:

$$\min_{w,\phi} \mathcal{L}_{\text{enriching}} \triangleq \|\mathbf{D}_w(\mathbf{SD}_\phi(x)) - \mathbf{P}_\gamma\|_2^2\,, \quad (8)$$

where $x \sim \mathcal{N}(0, I)$, $\mathbf{SD}_\phi$ is the frozen Stable Diffusion model modulated by LoRA with trainable parameters ϕ, and $\mathbf{D}_w$ is the latent K-Planes decoder. Note that, thanks to the alternating nature of our training and the absence of bias in the decoder's convolutional layers, this optimization does not reach full convergence. This is key because over-fitting the generative model to generate exactly $\mathbf{P}_\gamma$ leads to resuming scene fitting from exactly the same point.

At the end of the alternating training procedure, we save the enriched and corrected representation $\mathbf{P}_\gamma$ for rendering and testing. Algorithm 1 provides an overview of our training procedure.

4. Experiments

We evaluate EvE via experiments on synthetic datasets as well as real Phototourism [21] scenes. Quantitative results can be found in Tabs. 2 and 3, where we report for each experiment the Peak Signal-to-Noise Ratio (PSNR) for pixel-level similarity and the Structural Similarity Index Measure (SSIM) for structural-level similarity, computed on entire images. Figs. 1 and 3 illustrate the qualitative improvements of

	PSNR (↑)								
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Mean
NeRF [29]	33.00	25.01	30.13	36.18	32.54	29.62	32.91	28.65	31.00
TensoRF [6]	<u>35.76</u>	<u>26.01</u>	**33.99**	**37.41**	<u>36.46</u>	**30.12**	34.61	30.77	<u>33.14</u>
Plenoxels [11]	33.98	25.35	31.83	36.43	34.10	29.14	33.26	29.62	31.71
INGP [31]	35.00	**26.02**	<u>33.51</u>	<u>37.40</u>	36.39	<u>29.78</u>	**36.22**	<u>31.10</u>	**33.18**
K-Planes [12]	34.98	25.68	31.44	36.75	35.81	29.48	34.10	30.76	32.37
K-Planes-SS [12]	33.61	25.27	30.92	35.88	35.09	28.83	33.01	30.04	31.58
EvE (ours)	**35.77**	25.94	32.45	37.08	**36.47**	29.39	<u>34.77</u>	**31.41**	32.91

	SSIM (↑)								
	Chair	Drums	Ficus	Hotdog	Lego	Materials	Mic	Ship	Mean
NeRF [29]	0.967	0.925	0.964	0.974	0.961	0.949	0.980	0.856	0.947
TensoRF [6]	**0.985**	<u>0.937</u>	**0.982**	**0.982**	0.983	**0.952**	<u>0.988</u>	0.895	**0.963**
Plenoxels [11]	0.977	0.933	0.976	0.980	0.975	0.949	0.985	0.890	0.958
INGP [31]	—	—	—	—	—	—	—	—	—
K-Planes [12]	<u>0.983</u>	**0.938**	0.975	**0.982**	<u>0.982</u>	<u>0.950</u>	<u>0.988</u>	<u>0.897</u>	<u>0.962</u>
K-Planes-SS [12]	0.974	0.932	0.971	0.977	0.978	0.943	0.983	0.887	0.956
EvE (ours)	**0.985**	<u>0.937</u>	<u>0.980</u>	<u>0.981</u>	**0.984**	0.945	**0.989**	**0.903**	**0.963**

Table 3. **Quantitative results.** Results on static synthetic scenes [29]. The **bold** and <u>underlined</u> entries respectively indicate the best and second-best results. Dashes denote values that were not reported in prior work.

our method. For a further look, experimental details including hyperparameters and more dataset details can be found in Appendix A. Additional qualitative results are available in Appendix B. Please refer to our project page for more information: https://eve-nvs.github.io .

4.1. Datasets

For synthetic evaluations, as done in prior work [12, 29], we evaluate EvE on the NeRF synthetic dataset consisting of eight scenes containing various synthetic objects. As for real-world datasets, we evaluate EvE similarly to prior work [7, 12, 26] in novel view synthesis in-the-wild by adopting three scenes of cultural monuments from Phototourism [21]: *Brandenburg Gate*, *Sacré Coeur*, and *Trevi Fountain*.

4.2. Implementation details

For a fair comparison, we take similar experimental settings in scene fitting to [12]. However, due to the nature of our scene enriching pipeline, we limit the implementation in our case to a single-scale K-Planes of 512×512 resolution, in contrast to the multi-scale approach taken in [12] where $N \in \{64, 128, 256, 512\}$. The number of channels in each plane remains the same ($C = 32$). Also note that throughout all the experiments, we consider the hybrid implementation of K-Planes, where plane features are decoded into colors and densities by a small MLP. As for the scene enriching pipeline, we apply no modification to the U-Net in Stable Diffusion. Yet, we replace the last layer of the decoder D_ω with a new convolutional layer (without bias) to account for the shape of the K-Planes. Hence, the dimensions we work with in scene enriching (Fig. 2) are: $N = 512$, $C = 32$, $n = 64$, and $c = 4$. For an in-depth look at our hyperparameter settings, we refer the reader to Appendix A.

4.3. Evaluations

Baselines. For synthetic scenes, we evaluate our method against NeRF [29], TensoRF [6], Plenoxels [11], INGP [31], and K-Planes [12]. For in-the-wild scenes, we compare our results against NeRF-W [26] (results from the public implementation [1]), Ha-NeRF [7], CR-NeRF [47] and K-Planes [12]. Note that in both cases, for a fair assessment of the added value of scene enriching, our main point of comparison would be a single-scale ablation of K-Planes (dubbed K-Planes-SS). Nevertheless, our method almost always outperforms both K-Planes-SS and K-Planes.

Comparisons. For synthetic scenes, EvE consistently demonstrates better performances than K-Planes-SS and almost always outperforms K-Planes (Tab. 3), proving the effectiveness of the scene enriching procedure. As for in-the-wild scenes, EvE outperforms K-Planes as well as previous work tackling NVS in-the-wild (Tab. 2). This particularly highlights the value of the prior in scene enriching, as it has been trained on large-scale image collections, including real world images. Figs. 1 and 3 show qualitative comparisons of EvE with K-Planes, showing the visual improvements brought by scene enriching. EvE brings finer details to monuments, notably in less-frequently captured views (as

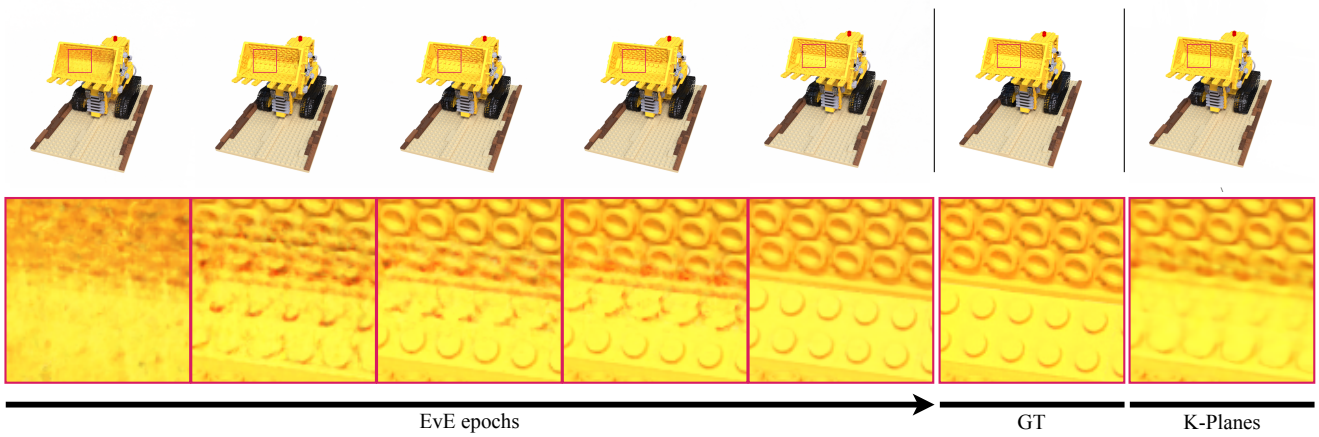


Figure 4. **Constrained optimization results.** Qualitative results showcasing the optimization progression on EvE, and a comparison with the ground truth and K-Planes. The training set is constrained to 50% of its initial size for both EvE and K-Planes.

denoted by the left side facade of the Sacré Coeur (Fig. 3)). Further qualitative results can be found in Appendix B.

4.4. Ablations

To justify our choices and explore further, we compare our in-the-wild results (Tab. 2) to two main ablations of our method. **EvE-F** is a variation of our method without LoRA fine-tuning. Here, we consider the same exact pipeline (frozen U-Net, same decoder configuration), except that we don't modulate the weights of the frozen U-Net with LoRA. This means that the prior is kept intact and no fine-tuning is done. **EvE-P** ablates the prior of Stable Diffusion by randomly re-initializing all U-Net weights while leaving all other elements of the scene-enriching pipeline intact. Finally, ablating the entire scene-enriching pipeline leads back to the **K-Planes-SS** setting. As illustrated in Tab. 2, we consistently obtain worse results during the ablation study as compared to our full model, thus demonstrating the value of the pre-trained prior, as well as the value of LoRA finetuning. This proves the importance of extending scene learning beyond closed-world settings, as a reasonably sized training set cannot capture all information in a scene, especially in-the-wild.

4.5. Further experiments

To further evaluate our method, and more particularly the robustness of the scene-enriching pipeline, we train K-Planes [12] and EvE on the synthetic Lego scene in a constrained setting: we restrict the training set to half of its original size. In the case of K-Planes, we qualitatively find that the reconstruction of the interior of the shovel is particularly challenging. We speculate that this is because the interior of the shovel is fully visible only in a few views in the constrained training set. Here, we observe an increase in the Mean Squared Error (MSE) from 3.37×10^{-4} to 4.36×10^{-4}. As for EvE, we observe that it gradually manages to reconstruct this interior throughout the alternating epochs. The MSE increases from 2.6×10^{-4} to 3.46×10^{-4}. Fig. 4 qualitatively showcases these findings.

	Training Time (↓)
NeRF-W [26]	400 hrs
Ha-NeRF [7]	452 hrs
CR-NeRF[47]	420 hrs
K-Planes [12]	50 mins
EvE (ours)	150 hrs

Table 4. **Training times.** Comparison of training time among methods tackling novel view synthesis in-the-wild. Timings are based on a single NVIDIA V100 GPU.

5. Discussions

As presented, EvE utilizes an alternating training procedure and a large-scale pre-trained generative prior to enrich scenes with fine details. While EvE demonstrates interesting results, it exhibits a limitation lying in training time, where our alternating training procedure leads to the repeated fine-tuning of both scene fitting and scene enriching pipelines. In addition, in order for this training to converge well, we apply a relatively small learning rate (10^{-4}) to the LoRA parameters, which leads to a slow optimization during the scene enriching phase. This means that further optimizing the LoRA finetuning would greatly accelerate our method. Yet, as our underlying representation is K-Planes, our total training time is still considerably lower than most in-the-wild methods (Tab. 4). We leave the exploration of training time optimizations for future work.

6. Conclusions

In this paper, we tackle the closed-world constraint of NVS in-the-wild. We introduce EvE, a method that incorporates a pre-trained generative prior to enrich scene representations thanks to an alternating training procedure. Extensive experiments show that EvE exhibits notable improvements, not only in real monument details, but also in constrained synthetic setups. In concluding this study, several avenues of future work emerge as we consider this work to be a first stepping-stone in eliminating closed-world constraints in scene learning using generative models. This includes the exploration of approaches to achieve scene enriching other than optimization guidance, the application of scene enriching on multi-scale representations, and the extension of appearance modeling beyond learnable embeddings.

Acknowledgements

This work was granted access to the HPC resources of IDRIS under the allocation 2023-AD011014261 made by GENCI. We thank Loic Landrieu, Vicky Kalogeiton and Thibaut Issenhuth for inspiring discussions and valuable feedback.

References

[1] Al Aoi. nerf_pl. https://github.com/kwea123/nerf_pl/tree/nerfw, 2022. Accessed: 2023-10-25. 7, 12

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, 2022. 5

[3] Ang Cao and Justin Johnson. HexPlane: A Fast Representation for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 130–141, 2023. 4

[4] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient Geometry-Aware 3D Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16123–16133, 2022. 2, 3, 4

[5] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 3

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. TensoRF: Tensorial Radiance Fields. In *European Conference on Computer Vision (ECCV)*, 2022. 7

[7] Xingyu Chen, Qi Zhang, Xiaoyu Li, Yue Chen, Ying Feng, Xuan Wang, and Jue Wang. Hallucinated Neural Radiance Fields in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12943–12952, 2022. 2, 3, 6, 7, 8

[8] Michael F. Cohen and Richard Szeliski. *Lumigraph*, pages 462–467. Springer US, Boston, MA, 2014. 3

[9] Hugging Face. Diffusers library. https://huggingface.co/docs/diffusers/v0.9.0/en/index, 2022. Accessed: 2023-10-25. 12

[10] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *arXiv preprint arXiv:2305.16381*, 2023. 5

[11] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance Fields Without Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5501–5510, 2022. 7

[12] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-Planes: Explicit Radiance Fields in Space, Time, and Appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12479–12488, 2023. 2, 3, 4, 5, 6, 7, 8, 15

[13] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local Deep Implicit Functions for 3D Shape. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[14] Gianluca Gimini. Velocipedia. https://www.gianlucagimini.it/portfolio-item/velocipedia/, 2016. Accessed: 2023-10-25. 1

[15] Alexandros Graikos, Nikolay Malkin, Nebojsa Jojic, and Dimitris Samaras. Diffusion Models as Plug-and-Play Priors. In *Advances in Neural Information Processing Systems*, pages 14715–14728. Curran Associates, Inc., 2022. 3

[16] Jiatao Gu, Alex Trevithick, Kai-En Lin, Josh Susskind, Christian Theobalt, Lingjie Liu, and Ravi Ramamoorthi. NerfDiff: Single-image View Synthesis with NeRF-guided Distillation from 3D-aware Diffusion. In *International Conference on Machine Learning*, 2023. 2, 3

[17] Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023. 2

[18] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004. 2

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, pages 6840–6851. Curran Associates, Inc., 2020. 3

[20] 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. In *International Conference on Learning Representations*, 2022. 5

[21] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image Matching Across Wide Baselines: From Paper to Practice. *International Journal of Computer Vision*, 129(2):517–547, 2020. 1, 2, 3, 5, 6, 7, 12, 13

[22] James T. Kajiya and Brian Von Herzen. Ray tracing volume densities. *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, 1984. 3, 5

[23] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4), 2023. 3

[24] Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*, 2023. 5

[25] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot One Image to 3D Object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9298–9309, 2023. 2, 3, 5

[26] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7210–7219, 2021. 2, 3, 6, 7, 8

[27] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. RealFusion: 360deg Reconstruction of Any Object From a Single Image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8446–8455, 2023. 2

[28] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-NeRF for Shape-Guided Generation of 3D Shapes and Textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12663–12673, 2023. 2

[29] 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 *ECCV*, 2020. 2, 3, 4, 6, 7, 12

[30] Thomas Müller. tiny-cuda-nn, 2021. 12

[31] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 7

[32] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable Volumetric Rendering: Learning Implicit 3D Representations Without 3D Supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[33] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 12

[35] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D Diffusion. In *The Eleventh International Conference on Learning Representations*, 2023. 2, 3

[36] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, 2022. 1, 2, 3, 5

[37] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. In *Advances in Neural Information Processing Systems*, 2022. 2, 3

[38] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 12

[39] J. Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3D Neural Field Generation Using Triplane Diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20875–20886, 2023. 2, 3

[40] H.Y. Shum, S.C. Chan, and S.B. Kang. *Image-Based Rendering*. Springer US, 2008. 2

[41] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, Rohit Pandey, Sean Fanello, Gordon Wetzstein, Jun-Yan Zhu, Christian Theobalt, Maneesh Agrawala, Eli Shechtman, Dan B. Goldman, and Michael Zollhöfer. State of the Art on Neural Rendering. *Computer Graphics Forum (Eurographics '20): State of the Art Reports*, 39(2):701 – 727, 2020. 3

[42] Michael Waechter, Nils Moehrle, and Michael Goesele. Let There Be Color! Large-Scale Texturing of 3D Reconstructions. In *Computer Vision – ECCV 2014*, pages 836–850, Cham, 2014. Springer International Publishing. 3

[43] Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin CK Chan, and Chen Change Loy. Exploiting Diffusion Prior for Real-World Image Super-Resolution. In *arXiv preprint arXiv:2305.07015*, 2023. 3, 5

[44] Daniel Watson, William Chan, Ricardo Martin Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel View Synthesis with Diffusion Models. In *The Eleventh International Conference on Learning Representations*, 2023. 2, 3

[45] Dan Wood, Daniel Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David Salesin, and Werner Stuetzle. Surface Light Fields for 3D Photography. *SIGGRAPH 2000, Computer Graphics Proceedings*, 2002. 3

[46] Jamie Wynn and Daniyar Turmukhambetov. DiffusioNeRF: Regularizing Neural Radiance Fields With Denoising Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4180–4189, 2023. 2

[47] Yifan Yang, Shuhai Zhang, Zixiong Huang, Yubing Zhang, and Mingkui Tan. Cross-Ray Neural Radiance Fields for Novel-View Synthesis from Unconstrained Image Collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15901–15911, 2023. 2, 6, 7, 8

[48] Yuchen Zeng and Kangwook Lee. The expressive power of low-rank adaptation. *arXiv preprint arXiv:2310.17513*, 2023. 5

EvE: Exploiting Generative Priors for Radiance Field Enrichment

Supplementary Material

A. Experimental Details

A.1. Datasets

Synthetic dataset. For synthetic renderings, we adopt the *Real Synthetic 360°* dataset from NeRF [29]. This dataset consists of eight path-traced scenes containing objects exhibiting complicated geometry and realistic non-Lambertian materials. Each image is coupled with its corresponding camera parameters. Consistently with prior work, 100 images are used for training each scene and 200 images are used for testing. All images are at 800×800 pixels.

In-the-wild dataset. For in-the-wild renderings, we adopt the Phototourism dataset [21] which is commonly used for in-the-wild tasks. This dataset consists of multitudes of images of touristic landmarks gathered from the internet. Thus, these images are naturally plagued by visual discrepancies, notably illumination variation and transient occluders. Camera parameters are estimated using COLMAP [38]. All images are normalized to [0,1]. We adopt three scenes from Phototurism: *Brandenburg Gate* (1363 images), *Sacré Coeur* (1179 images), and *Trevi Fountain* (3191 images). Testing is done on a standard set that is free of transient occluders.

A.2. Frameworks

We make use of multiple frameworks to implement our method. Our Python source code (tested on version 3.7.16), based on PyTorch [34] (tested on version 1.13.1) and CUDA (tested on version 11.6) will be publicly available as open source. We also utilize Diffusers [9] and Stable Diffusion (tested on version 1-5, main revision). K-Planes also adopt the *tinycudann* framework [30]. We run all experiments on a single NVIDIA V100 GPU.

A.3. Hyperparameters

A summary of our hyperparameters for synthetic as well as in-the-wild scenes can be found in Tab. A. Please refer to the configuration files of our code for more information.

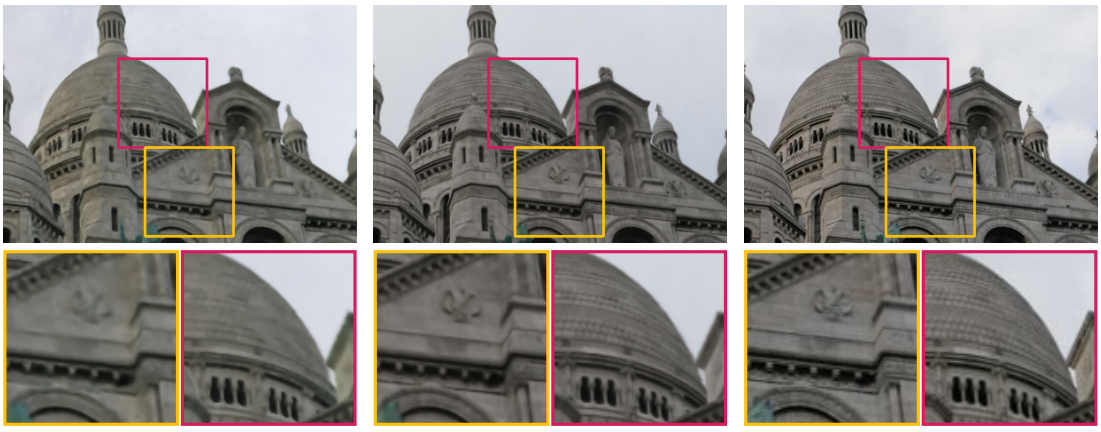
B. Results

Quantitative results. We reproduce the results on K-Planes and K-Planes-SS. In-the-wild results on NeRF-W come from a public implementation of the paper [1] and are extracted from the K-Planes paper. Ha-NeRF and CR-NeRF results are extracted from the recent CR-NeRF paper. Remaining results in Tabs. 2 and 3 are extracted from their corresponding papers.

Qualitative results. We present below additional qualitative results for in-the-wild scenes (Figs. A to C) and synthetic renderings Figs. D and E. Please refer to our project page for animations of our renderings.

(a) K-Planes (b) EvE (c) GT

Figure A. **Qualitative results.** Results on the *Brandenburg Gate* scene from Phototourism [21].



(a) K-Planes (b) EvE (c) GT

Figure B. **Qualitative results.** Results on the *Sacré Coeur* scene from Phototourism [21].

(a) K-Planes (b) EvE (c) GT

Figure C. **Qualitative results.** Results on the *Trevi Fountain* scene from Phototourism [21].

Figure D. **Qualitative results.** EvE results on the NeRF Synthetic scenes.

Figure E. **Ground Truth Renderings.** Ground truth images from the NeRF Synthetic dataset.

Hyperparameter	Value
Epochs (N_{epochs})	200 (synthetic) 20 (*Sacré Coeur*) 20 (*Brandenburg Gate*) 10 (*Trevi Fountain*)
Epochs K-Planes (N_1)	1
Epochs LoRA (N_2)	3000
Batch size	4096
Optimizer	Adam
Scheduler	Warmup Cosine
K-Planes Learning Rate	0.01
LoRA Learning rate	0.0001
SD latent resolution	64
SD channel dimension	4
SD prompt	" "
Number of planes	3
K-Planes resolution	512
K-Planes channel dimension	32
Epochs Appearance Optimization	10
Appearance embeddings dimension	32
Appearance learning rate	0.1 (*Sacré Coeur*) 0.1 (*Trevi Fountain*) 0.001 (Brandenburg Gate)
Appearance batch size	512

Table A. **Hyperparameters.** A summary of the hyperparameters used to train our model. Appearance optimizations only apply in in-the-wild training. Remaining K-Planes parameters are taken identically to [12].