NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections

Ricardo Martin-Brualla*, Noha Radwan*, Mehdi S. M. Sajjadi*, Jonathan T. Barron,
Alexey Dosovitskiy, and Daniel Duckworth

Google Research

{rmbrualla, noharadwan, msajjadi, barron, adosovitskiy, duckworthd}@google.com

Abstract

We present a learning-based method for synthesizing novel views of complex outdoor scenes using only unstructured collections of in-the-wild photographs. We build on neural radiance fields (NeRF), which uses the weights of a multilayer perceptron to implicitly model the volumetric density and color of a scene. While NeRF works well on images of static subjects captured under controlled settings, it is incapable of modeling many ubiquitous, real-world phenomena in uncontrolled images, such as variable illumination or transient occluders. In this work, we introduce a series of extensions to NeRF to address these issues, thereby allowing for accurate reconstructions from unstructured image collections taken from the internet. We apply our system, which we dub NeRF-W, to internet photo collections of famous landmarks, thereby producing photorealistic, spatially consistent scene representations despite unknown and confounding factors, resulting in significant improvement over the state of the art.

1. Introduction

Synthesizing novel views of a scene from a sparse set of captured images is a long-standing problem in computer vision, and a prerequisite to many AR and VR applications. Though classic techniques have addressed this problem using structure-from-motion [11] or image-based rendering [27], this field has recently seen significant progress due to *neural rendering* techniques — learning-based modules embedded within a 3D geometric context, and trained to reconstruct observed images. The Neural Radiance Fields (NeRF) approach [21] implicitly models the radiance field and density of a scene within the weights of a neural network. Direct volume rendering is then used to synthesize new views, demonstrating a heretofore unprecedented level of fidelity on a range of challenging scenes. However, NeRF has only been demonstrated to work well in controlled settings: the scene is captured within a short time frame during which lighting effects remain constant, and all content in the scene is static. As we will demonstrate, NeRF's performance degrades significantly when presented with moving objects or variable illumination. This limitation prohibits direct application of NeRF to large-scale in-the-wild scenarios, where input images may be acquired over the course of hours, days, or years, and scenes may contain pedestrians and cars moving throughout them.

(a) Photos (b) Renderings

Figure 1: Given only images from internet photo collections (left), our method is able to render novel views under variable lighting conditions (right). Further results on `https://nerf-w.github.io/`. Photos by Flickr users dbowie78, vasnic64, and punch / CC BY.

The central limitation of NeRF that we address in this work is its assumption that the world is geometrically, materially, and photometrically *static* — that the density and radiance of the world is constant. NeRF therefore requires that any two photographs taken at the same position and orientation must have identical pixel intensities. This assumption is severely violated in many real-world datasets, such as large-scale internet photo collections of well-known

*Denotes equal contribution

tourist landmarks. Two photographers may stand in the same location and photograph the same landmark, but in the time between those two photographs the world can change significantly: cars and pedestrians may move, construction may begin or end, seasons and weather may change, and the sun may move though the sky, etc. Even two photos taken at the same time and location can exhibit considerable variation: exposure, color correction, and tone-mapping may all vary depending on the camera and post-processing procedures employed. We will demonstrate that naively applying NeRF to in-the-wild photo collections results in inaccurate representations that exhibit severe ghosting, oversmoothing, and other artifacts.

To handle these complex scenarios, we present NeRF-W, an extension of NeRF that relaxes the latter's strict consistency assumptions. First, we model per-image appearance variations such as exposure, lighting, weather, and post-processing with a learned low-dimensional latent space. Following the framework of Generative Latent Optimization [3], we optimize an appearance embedding for each input image, thereby granting NeRF-W the flexibility to explain away photometric and environmental variations between images by learning a shared appearance representation for the entire photo collection. The learnt appearance latent space provides control of the appearance of output renderings as illustrated in Figure 1.

Second, we model the scene as the union of shared and image-dependent elements, thereby enabling the unsupervised decomposition of scene content into static and transient components. This decomposition enables the high-fidelity synthesis of novel views of landmarks without the artifacts otherwise induced by dynamic visual content present in the input imagery. Our approach models transient elements as a secondary volumetric radiance field combined with a data-dependent uncertainty field, with the latter capturing variable observation noise and further reducing the effect of transient objects on the static scene representation.

We apply NeRF-W to several challenging in-the-wild photo collections of cultural landmarks and find it capable of producing detailed, high-fidelity renderings from novel viewpoints, surpassing the prior state of the art by a large margin across all considered metrics. We demonstrate smooth appearance interpolation and 3D consistency in rendered videos. In addition, we perform a detailed ablation study of NeRF-W's individual enhancements in a synthetic setting and confirm that each produces its intended effect. We find that NeRF-W significantly improves quality over NeRF in the presence of appearance variation and transient occluders while achieving similar quality in controlled settings.

2. Related Work

The last decade has seen a gradual integration of physics-based multi-view geometry techniques with deep learning-based approaches for the task of 3D scene reconstruction. In this section, we review recent work on the topics of novel view synthesis and neural rendering, and highlight the main differences between existing approaches and our proposed method.

Novel View Synthesis: Constructing novel views of a scene captured by multiple images is a long standing problem in computer vision. Structure-from-Motion (SfM) [11] and bundle adjustment [37] can be used to reconstruct a sparse point cloud representation and recover camera parameters. The seminal work of Photo Tourism [30] showed how to scale such reconstructions to unconstrained photo collections, leading to a revolution of scaling SfM to millions of images [1, 9] and multi-view stereo techniques [10, 25]. Other approaches to novel-view synthesis include light-field photography [17] and image-based rendering [5] but generally require a dense capture of the scene. Recent works explicitly infer the light and reflectance properties of the objects in the scene from a set of unconstrained photo collections [16, 26]. Others utilize semantic knowledge to reconstruct transient objects [23].

Neural Rendering: More recently, neural rendering techniques [34] have been applied to scene reconstruction. Several approaches employ image translation networks [12] to re-render content more realistically using as input traditional reconstruction results [19], learned latent textures [35], point clouds [2], voxels [28], or plane sweep volumes [7, 8]. Most similar in application to our work is Neural Rerendering in the Wild (NRW) [20] which synthesizes realistic novel views of tourist sites from point cloud renders by learning a neural re-rendering network conditioned on a learned latent appearance embedding module. Common drawbacks of the aforementioned approaches, however, are the checkerboard and temporal artifacts visible under camera motion caused by the employed 2D image translation network. Volume rendering approaches [18, 21, 29], on the other hand, lead to more view-consistent reconstructions of the scene. Neural Radiance Fields (NeRF) [21] use a multi-layer perceptron (MLP) to model a radiance field, at an unprecedented level of fidelity, in part thanks to a novel positional encoding layer in the network [33]. Our work focuses on extending NeRF to unconstrained scenarios, like internet photo collections.

3. Background

We frame the problem of *3D scene reconstruction* of a scene from a photo collection as that of learning a representation capable of generating the collection's images in a 3D-consistent way. Such a scene representation should further be capable of synthesizing novel, unseen views. As such, the representation needs to encode the 3D structure of the scene, together with appearance information, to enable consistent view synthesis. In the following we describe

Neural Radiance Fields [21] (NeRF), one such method for 3D scene reconstruction and the system which NeRF-W extends.

NeRF represents a scene with learned, continuous volumetric radiance field F_θ defined over a bounded 3D volume. In NeRF, F_θ is a multilayer perceptron (MLP) that takes as input a 3D position $\mathbf{x} = (x, y, z)$ and unit-norm viewing direction $\mathbf{d} = (d_x, d_y, d_z)$, and produces as output a density σ and color $\mathbf{c} = (r, g, b)$. The weights of the multilayer perceptron that parameterize F_θ are optimized so as to encode the radiance field of the scene.

To compute the color of a single pixel, NeRF approximates the volume rendering integral defined in Equation 1. Let $\mathbf{r}(t) = \mathbf{o} + t\,\mathbf{d}$ be the camera ray emitted from the center of projection of a camera through a given pixel on the image plane. The expected color $\bar{\mathbf{C}}(\mathbf{r})$ of the corresponding pixel is given by:

$$\bar{\mathbf{C}}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(t)\mathbf{c}(t)dt\,, \tag{1}$$

$$\text{where} \quad T(t) = \exp\left(-\int_{t_n}^{t} \sigma(s)ds\right), \tag{2}$$

where $\sigma(t)$ and $\mathbf{c}(t)$ are the density and color at point $\mathbf{r}(t)$, and t_n and t_f are the near and far bounds of integration.

In NeRF, the integrals of Equations (1) and (2) are approximated via numerical quadrature. A stratified sampling approach is used to select random quadrature points $\{t_k\}_{k=1}^{K}$ between t_n and t_f, and the approximate expected color is given by:

$$\hat{\mathbf{C}}(\mathbf{r}) = \sum_{k=1}^{K} \hat{T}(t_k)\,\alpha(\sigma(t_k)\delta_k)\,\mathbf{c}(t_k)\,, \tag{3}$$

$$\text{where} \quad \hat{T}(t_k) = \exp\left(-\sum_{k'=1}^{k-1} \sigma(t_k)\delta_k\right), \tag{4}$$

and $\alpha(x) = 1 - \exp(-x)$ and $\delta_k = t_{k+1} - t_k$ is the distance between two quadrature points.

The volume density $\sigma(t)$ and color $\mathbf{c}(t)$ are represented by a neural network of the following form:

$$[\sigma(t), \mathbf{z}(t)] = \text{MLP}_{\theta_1}(\gamma_{\mathbf{x}}(\mathbf{r}(t)))\,, \tag{5}$$

$$\mathbf{c}(t) = \text{MLP}_{\theta_2}(\mathbf{z}(t), \gamma_{\mathbf{d}}(\mathbf{d}))\,, \tag{6}$$

with parameters $\theta = (\theta_1, \theta_2)$ and fixed positional encoding functions $\gamma_{\mathbf{x}}$ (for position) and $\gamma_{\mathbf{d}}$ (for viewing direction). ReLU and sigmoid nonlinearities are applied to $\sigma(t)$ and $\mathbf{c}(t)$ respectively. We depart from the exposition of [21] and present the neural network as a series of two multilayer perceptrons, with the latter depending on one output of the former, $\mathbf{z}(t)$. Note that volume density $\sigma(t)$ is independent of viewing direction $\mathbf{d}$.

Figure 2: Example in-the-wild photographs from the Phototourism dataset [13] used to train NeRF-W. Due to lighting and post-processing (top row), the same object's color may vary from image to image. In-the-wild photos further exhibit an unrestricted set of potential occluders (bottom row). Photos by Flickr users paradasos, itia4u, jblesa, joshheumann, ojotes, and chyauchentravelworld / CC BY.

To fit parameters θ, NeRF minimizes the sum of squared reconstruction errors with respect to an (RGB) image collection $\{\mathcal{I}_i\}_{i=1}^{N}, \mathcal{I}_i \in [0,1]^{H\times W\times 3}$. We assume each image $\mathcal{I}_i$ is paired with its corresponding intrinsic and extrinsic camera parameters, which can be estimated using structure-from-motion [24] for real images. We construct the set of camera rays $\{\mathbf{r}_{ij}\}_{j=1}^{H\times W\times 3}$ corresponding to each image i and pixel j with each ray passing through the 3D location $\mathbf{o}_i$ with direction $\mathbf{d}_{ij}$ where $\mathbf{r}_{ij}(t) = \mathbf{o}_i + t\,\mathbf{d}_{ij}$.

To improve sampling efficiency, NeRF simultaneously optimizes two volumetric radiance fields, one *coarse* and one *fine*. The volumetric density learned by the *coarse* model is used to bias the sampling of quadrature points for the *fine* model. The parameters of both models are chosen to minimize the following loss function,

$$\sum_{ij} \left\|\mathbf{C}(\mathbf{r}_{ij}) - \hat{\mathbf{C}}_c(\mathbf{r}_{ij})\right\|_2^2 + \left\|\mathbf{C}(\mathbf{r}_{ij}) - \hat{\mathbf{C}}_f(\mathbf{r}_{ij})\right\|_2^2, \tag{7}$$

where $\mathbf{C}(\mathbf{r}_{ij})$ is the observed color of ray j in image $\mathcal{I}_i$.

4. NeRF in the Wild

We now present NeRF-W, a system for reconstructing 3D scenes from in-the-wild photo collections. We build on NeRF [21] and introduce two enhancements explicitly designed to handle the challenges of unconstrained imagery.

Similar to NeRF, we learn a volumetric density representation F_θ from an unstructured photo collection $\{\mathcal{I}_i\}_{i=1}^{N}$ for which camera parameters are available or have been estimated. Intrinsically, NeRF assumes consistency in its input views, *i.e.* a point in 3D space observed from the same position and viewing direction in two different images will have the same intensity. However, internet photos, like the ones

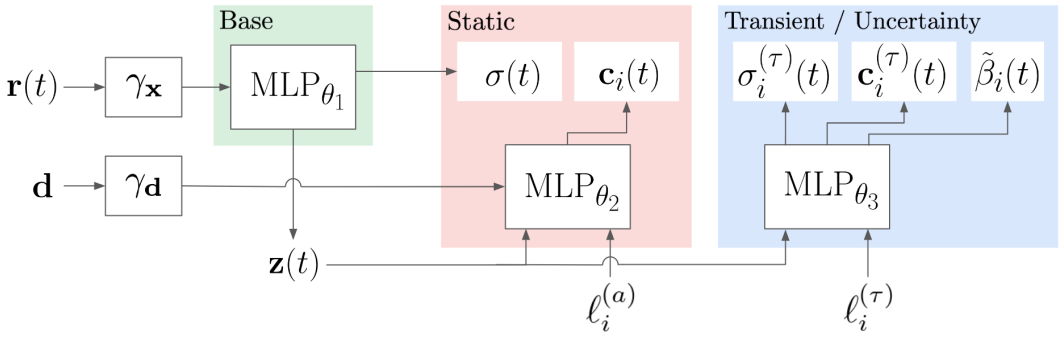


Figure 3: NeRF-W model architecture. Given a 3D position $\mathbf{r}(t)$, viewing direction $\mathbf{d}$, appearance embedding $\boldsymbol{\ell}_i^{(a)}$, and transient embedding $\boldsymbol{\ell}_i^{(\tau)}$, NeRF-W produces differential opacities $\sigma(t), \sigma_i^{(\tau)}(t)$, colors $\mathbf{c}_i(t), \mathbf{c}_i^{(\tau)}(t)$, and uncertainty $\beta_i(t)$. Note that the static opacity $\sigma(t)$ is generated before the model is conditioned on appearance embedding $\boldsymbol{\ell}_i^{(a)}$ to ensure that static geometry is shared across all images.

shown in Figure 2, do not adhere to such strong assumptions. This assumption is violated by two distinct phenomena,

1) Photometric variation: In outdoor photography, time of day and atmospheric conditions directly impact the illumination (and consequently, the emitted radiance) of all objects in the scene. This issue is exacerbated by photographic image pipelines as variation in auto-exposure settings, white balance, and tone-mapping across photographs may result in additional photometric inconsistencies [4]. These issues can also appear in controlled settings, such as when the photographs are taken several minutes apart, causing cast shadows to move, or auto-exposure parameters to change

2) Transient objects: Real-world landmarks are rarely captured in isolation, without moving objects or distractors around them. Internet photo collections are particularly challenging, as they often contain posing subjects and other pedestrians in the photo. Such transient objects occlude the static scene and contaminate reconstructions. In our experiments, this is typically observed as a dark fog appearing above ground-level in locations frequented by transient objects.

We propose two model components to address these issues. In Section 4.1 we extend NeRF to allow for image-dependent appearance and illumination variation such that photometric discrepancies between images can be modeled explicitly. In Section 4.2 we further extend this model by allowing transient objects to be jointly estimated and disentangled from a static representation of the 3D world. Refer to Figure 3 for an illustration of the model architecture.

4.1. Latent Appearance Modeling

To adapt NeRF to variable lighting and photometric post-processing, we introduce a dependency on image index i to the expected color in Equation (1):

$$\bar{\mathbf{C}}_i(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(t)\mathbf{c}_i(t)dt \tag{8}$$

with $T(t)$ defined as before.

We adopt the approach of Generative Latent Optimization [3] (GLO) in which each image $\mathcal{I}_i$ is assigned a corresponding real-valued appearance embedding vector $\boldsymbol{\ell}_i^{(a)}$ of length $n^{(a)}$. As in NeRF, we approximate Equation (8) with numerical quadrature, replacing the image-independent radiance $\mathbf{c}(t)$ with image-dependent radiance,

$$\mathbf{c}_i(t) = \text{MLP}_{\theta_2}\left(\mathbf{z}(t), \gamma_{\mathbf{d}}(\mathbf{d}), \boldsymbol{\ell}_i^{(a)}\right). \tag{9}$$

Embeddings $\{\boldsymbol{\ell}_i^{(a)}\}_{i=1}^N$ are optimized over the course of training alongside NeRF's parameters θ.

Appearance embeddings $\{\boldsymbol{\ell}_i^{(a)}\}$ grant NeRF the freedom to vary the emitted radiance of an image-independent, shared 3D geometry. By setting $n^{(a)}$ to a small value, we encourage optimization to identify a continuous space in which illumination conditions can be embedded and decoded, enabling smooth interpolations between conditions as demonstrated in Figure 8. Although similar in spirit to viewing direction-dependent radiance, appearance embeddings enable the modeling of a wider range of phenomena not easily explained by specularites.

4.2. Transient Objects

We adapt NeRF to transient phenomena in two ways. First, we augment NeRF's volumetric radiance field with an explicit representation for transient objects. This enables NeRF-W to reconstruct images containing occluders without introducing artifacts into the static scene representation. Second, instead of modeling the observed color directly, we

model a probability distribution over its value. In particular, we model each pixel's color as an isotropic normal distribution and generate its mean and variance using the same volume rendering approach applied in NeRF. This enables NeRF-W to express uncertainty when rendering pixels that are likely to contain occluders (*e.g.* pedestrians). These two additions enable NeRF-W to disentangle static and transient phenomena without explicit supervision.

We begin by introducing a variation of the volumetric rendering equation. Building on Equation (8), we augment static density $\sigma(t)$ and radiance $\mathbf{c}_i(t)$ with transient counterparts $\sigma_i^{(\tau)}(t)$ and $\mathbf{c}_i^{(\tau)}(t)$,

$$\bar{\mathbf{C}}_i(\mathbf{r}) = \int_{t_n}^{t_f} T_i(t)\Big(\sigma(t)\,\mathbf{c}_i(t) + \sigma_i^{(\tau)}(t)\,\mathbf{c}_i^{(\tau)}(t)\Big)dt\,, \tag{10}$$

$$\text{where} \quad T_i(t) = \exp\bigg(-\int_{t_n}^{t}\Big(\sigma(s) + \sigma_i^{(\tau)}(s)\Big)ds\bigg)\,. \tag{11}$$

As in Section 4.1, the density of static objects, $\sigma(t)$, is shared among all images while the radiance, $\mathbf{c}_i(t)$, is image-dependent. The expected color of $\mathbf{r}(t)$ is the alpha composite of both static and transient components.

We employ the Bayesian learning framework of Kendall et al. [15] to model uncertainty in observed color. In particular, we assume that observed pixel intensities are inherently noisy (aleatoric) and further that this noise is input-dependent (heteroscedastic). We model observed color $\mathbf{C}_i(\mathbf{r})$ with an isotropic Normal distribution with image- and ray-dependent variance $\beta_i(\mathbf{r})^2$,

$$\mathbf{C}_i(\mathbf{r}) \sim \mathcal{N}\big(\bar{\mathbf{C}}_i(\mathbf{r}), \beta_i(\mathbf{r})^2\mathbb{I}_3\big)\,. \tag{12}$$

Variance $\beta_i(\mathbf{r})$ is 'rendered' analogously to color via alpha-compositing according to transient density $\sigma_i^{(\tau)}(t)$,

$$\beta_i(\mathbf{r}) = \int_{t_n}^{t_f} T_i^{(\tau)}(t)\,\sigma_i^{(\tau)}(t)\,\beta_i(t)dt\,, \tag{13}$$

$$\text{where} \quad T_i^{(\tau)}(t) = \exp\bigg(-\int_{t_n}^{t}\sigma_i^{(\tau)}(s)ds\bigg)\,. \tag{14}$$

Similar to NeRF, we employ numerical quadrature to approximate the above integrals. We provide a detailed description of the approximation in the Appendix A.

To model the transient component of the scene, we assign each image $\mathcal{I}_i$ a second image-dependent embedding $\boldsymbol{\ell}_i^{(\tau)} \in \mathbb{R}^{n^{(\tau)}}$. We augment F_θ by introducing a second MLP *head*,

$$\Big[\sigma_i^{(\tau)}(t), \mathbf{c}_i^{(\tau)}(t), \tilde{\beta}_i(t)\Big] = \mathrm{MLP}_{\theta_3}\Big(\mathbf{z}(t), \boldsymbol{\ell}_i^{(\tau)}\Big)\,, \tag{15}$$

$$\beta_i(t) = \beta_{\min} + \log\Big(1 + \exp\Big(\tilde{\beta}_i(t)\Big)\Big)\,, \tag{16}$$

where $\beta_{\min} > 0$ is a hyperparameter ensuring a minimum importance is assigned to each ray. Similar to their static counterparts, ReLU and sigmoid nonlinearities are applied to $\sigma_i^{(\tau)}(t)$ and $\mathbf{c}_i^{(\tau)}(t)$. See Figure 3 for an illustration of NeRF-W's model architecture.

The loss for ray $\mathbf{r}(t)$ in image i with ground-truth color $\mathbf{C}_i(\mathbf{r})$ is given by,

$$L_i(\mathbf{r}) = \frac{1}{2\beta_i(\mathbf{r})^2}\Big\|\mathbf{C}_i(\mathbf{r}) - \hat{\mathbf{C}}_i(\mathbf{r})\Big\|_2^2 + \frac{1}{2}\log\beta_i(\mathbf{r})^2 + \frac{\lambda_u}{K}\sum_{k=1}^{K}\sigma_i^{(\tau)}(t_k)\,. \tag{17}$$

The first two terms represent the negative log likelihood of $\mathbf{C}_i(\mathbf{r})$ according to a normal distribution with mean $\hat{\mathbf{C}}_i(\mathbf{r})$ and variance $\beta_i(\mathbf{r})^2$. Intuitively, larger values of $\beta_i(\mathbf{r})^2$ attenuate the importance assigned to a pixel — typically transient or other phenomena not explicitly modeled. The first term is balanced by the second, which corresponds to the log-partition function of the normal distribution and excludes the trivial minimum achieved at $\beta_i(\mathbf{r}) = \infty$. The third term is an explicit L_1 regularizer for (non-negative) transient density $\sigma_i^{(\tau)}(t)$ with hyperparameter λ_u, encouraging its sparsity.

At test time, we omit the transient volumetric radiance field and uncertainty, rendering only with $\sigma(t)$ and $\mathbf{c}(t)$. See Figure 4 for an illustration of static, transient, and uncertainty components.

4.3. NeRF-W

We now introduce NeRF-W, integrating the previously-described enhancements into the learning framework employed in NeRF. As in NeRF, we apply hierarchical volume sampling by simultaneously optimizing two copies of F_θ. The *fine* model uses the extensions described above while the *coarse* model follows the same architecture as that employed in NeRF. We find that this configuration consistently produces higher-accuracy models than using the same losses for both.

In addition to parameters θ, we optimize per-image appearance embeddings $\{\boldsymbol{\ell}_i^{(a)}\}_{i=1}^N$ and transient embeddings $\{\boldsymbol{\ell}_i^{(\tau)}\}_{i=1}^N$. We apply no regularization to these additional parameters. NeRF-W's loss function is then,

$$\sum_{ij} L_i(\mathbf{r}_{ij}) + \frac{1}{2}\Big\|\mathbf{C}(\mathbf{r}_{ij}) - \hat{\mathbf{C}}_c(\mathbf{r}_{ij})\Big\|_2^2\,, \tag{18}$$

where $L_i(\mathbf{r})$ is defined in Equation (17) and applied solely to the *fine* model. λ_u, $\beta_{\min}$, and embedding dimensionalities $n^{(a)}$ and $n^{(\tau)}$ form the set of additional hyperparameters for NeRF-W.

As described in Section 4.2, we omit transient volume density $\sigma^{(\tau)}(t)$, transient radiance $\mathbf{c}^{(\tau)}(t)$, and uncertainty $\beta(t)$ at test time. As training only optimizes appearance

(a) Static (b) Transient (c) Combined (d) Uncertainty (e) Reference

Figure 4: NeRF-W's rendering process. NeRF-W separately models the static (a) and transient (b) elements of the scene, rendering them simultaneously to produce a composite image (c). At training time, NeRF-W may also choose to ignore parts of the image (d) that cannot be easily explained at training time. The combined image is compared to a ground-truth reference (e) via a weighted reconstruction loss. Photo by Flickr user vasnic64 / CC BY.

embeddings $\{\boldsymbol{\ell}_i^{(a)}\}$ for images in the training set, we are free to choose their value at test time. For visualizations, we choose $\boldsymbol{\ell}^{(a)}$ to match a target image (*e.g*. Figure 8) or set it to an arbitrary value.

5. Experiments

Here we provide an evaluation of NeRF-W on in-the-wild (Table 1) and synthetic (Table 2) photo collections. We urge the reader to refer to the project website for additional results and videos: `https://nerf-w.github.io/`.

Baselines: We evaluate our proposed method against Neural Rerendering in the Wild (NRW) [20], NeRF [21], and two ablations of NeRF-W: NeRF-A (appearance), wherein the 'transient' head is eliminated; and NeRF-U (uncertainty), wherein appearance embedding $\boldsymbol{\ell}_i^{(a)}$ is eliminated. Note that NeRF-W is a composition of both NeRF-A and NeRF-U.

Datasets: We present experiments in two domains: unconstrained (*e.g*. "in-the-wild"), internet photo collections of cultural landmarks and rendered images of a synthetic scene. For the former, we select three landmarks from the Phototourism dataset [13]. Inspired by prior work [20], we reconstruct the *Trevi Fountain* and *Sacre Coeur* as well as a novel scene, the *Brandenburg Gate*. For each dataset, we run COLMAP [24] with two radial and two tangential distortion parameters enabled. Results are presented in Section 5.1. For a controlled ablation study, we construct variations of the Lego dataset [21] inspired by effects we expect to find in-the-wild. Results are presented in Section 5.2.

Training: Building off of NeRF[1], we implement all experiments in Tensorflow 2 using Keras. As in NeRF, we train a model per scene from scratch.

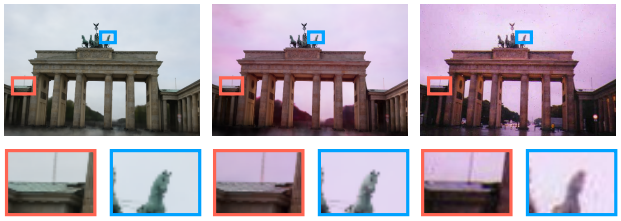
For the Phototourism datasets, we optimize all NeRF variants for 300,000 steps on 8 GPUs with the Adam optimizer [6]. Training takes approximately 2 days. Hyperparameters shared by all NeRF variants are chosen to maximize PSNR on the Brandenburg Gate dataset and are reused in Sacre Coeur and the Trevi Fountain. Additional hyperparameters for variants of NeRF-W are chosen via grid search to maximize PSNR on a held-out validation set on each scene. See Appendix B for additional details on the hyperparameter choices.

For the Lego datasets, we optimize for 125,000 steps on 4 GPUs, taking approximately 8 hours. We use the same NeRF hyperparameters reported by Mildenhall et al. [21] for all NeRF variants. Similar to the Phototourism datasets, additional NeRF-W hyperparameters are optimized per-photo collection. We detail our choice of hyperparameters in Appendix B.

Evaluation: We evaluate on the task of novel view synthesis: given a heldout image with accompanying camera parameters, we render an image from the same pose and compare it to the ground truth. As measuring perceptual image similarity is challenging [22, 36, 38, 41], we present rendered images for visual inspection and report quantitative results based on the PSNR, MS-SSIM [39], LPIPS [41] and the Census Transform (CT) [40]. CT encodes only the relative intensity of each pixel with respect to its neighbors, and is therefore invariant to low frequency scaling or shifts of pixel intensity (such as the global appearance changes we observe in real scenes) and focuses on the structure of the image. See Appendix B for additional details.

Recall that NeRF-W is conditioned upon an appearance embedding $\boldsymbol{\ell}^{(a)}$. Appearance embeddings are optimized for photos in the training set and are an otherwise free variable when rendering novel views. A natural choice is the mean appearance embedding over the training set. However, we find that global photometric effects obscure differences in model quality and aggravates the challenges of quantitative evaluation. To better align global photometric qualities with the ground truth, we optimize an appearance embedding $\boldsymbol{\ell}^{(a)}$ on the *left half* of each validation image and report metrics on the *right half*. Note that $\boldsymbol{\ell}^{(a)}$ does not impact the volumetric density of static scene content by design; see Figure 3 and Figure 5. We discuss implications of this approach and report metrics without optimization in the supplement (see Appendix B). We encourage readers to com-

[1] `https://github.com/bmild/nerf`



(a) NeRF-W w/o opt. (b) NeRF-W (c) Reference

Figure 5: To better match the appearance of NeRF-W renders (a, b) to that of the reference images (c), we optimize appearance embeddings $\boldsymbol{\ell}^{(a)}$ using only the *left half* of each image. Note that the scene geometry is not affected by this process due to the intentional constraints in the architecture shown in Figure 3. All metrics presented are computed on the *right half* of test images to avoid information leakage. Photo by Flickr user eadaoinflynn / CC BY.

(a) Reference (b) NeRF (c) NeRF-W

Figure 6: Rendered depth maps. NeRF-W is better able to model the 3D and 2D phenomena found in in-the-wild photo collections and thereby produces cleaner, crisper 3D reconstructions than its baseline. Photos by Flickr users burkeandhare, photogreuhphies / CC BY.

pare model quality in rendered videos on the project website, `https://nerf-w.github.io/`.

5.1. Phototourism Dataset

In this section, we evaluate all models on three photo collections from the Phototourism dataset [13]. We elaborate on dataset pre-processing steps for filtering the image collections in Appendix C.

Figure 7 shows qualitative results for all models and baselines. NRW produces renders with typical checkerboard artifacts across all images common to 2D re-rendering methods [14]. NRW is also sensitive to upstream errors in 3D geometry, such as incomplete point clouds, as shown in the Brandenburg Gate. NeRF produces a consistent 3D geometry, but large parts of the scene have unpleasing ghosting artifacts and occlusions, which are particularly noticeable on the Sacre Coeur image in this case. Furthermore, renderings from NeRF tend to exhibit strong global color shifts when compared to the ground truth. These artifacts are the direct consequence of NeRF's static-world assumption — photometric variation and transient occlusion must therefore be directly integrated into NeRF's scene representation. Such artifacts are also encountered in the rendered depth map from NeRF as seen in Figure 6, where the 3D geometry of the scene is much more affected by the presence of transient objects and color variations resulting in inaccuracies and blurriness. On the other hand, NeRF-W is able to produce higher quality crisp reconstructions.

NeRF-A improves upon NeRF, and as shown in Figure 7, its renderings are largely free of fog. We attribute this to NeRF-A's ability to capture photometric variation in $\boldsymbol{\ell}^{(a)}$. However, NeRF-A must expend model capacity to capture transient phenomena and thus fails to high-frequency detail such as the tower in Sacre Coeur. NeRF-U, on the other hand, is able to capture fine detail but is unable to model photometric effects. NeRF-W improves upon both and produces sharper, photometrically matching renders; note, for example, the inscription shown on the Trevi Fountain (red box).

Quantitative results are summarized in Table 1. Training NeRF on in-the-wild photo collections leads to particularly poor results that are unable to compete with NRW. In contrast, NeRF-W outperforms the baselines on all metrics across all datasets. In particular, NeRF-W improves over the previous state of the art NRW by an average margin of 5.7dB in PSNR, and with up to 40% improvements in MS-SSIM and CT.

Controllable Appearance: A consequence of modeling appearance with a latent embedding space $\boldsymbol{\ell}^{(a)} \in \mathbb{R}^{n^{(a)}}$ is that it enables one to modify the lighting and appearance of a render without altering the underlying 3D geometry. In Figure 1 (right), we see slices of four rendered images produced by NeRF-W using appearance embeddings associated with four photos from the training set. While appearance significantly varies between each slice, the underlying 3D structure remains unchanged.

In addition to the embeddings associated with photos in the training set, one may also apply NeRF-W to arbitrary vectors in the same space. In Figure 8, we present five images rendered from a fixed camera position, where we interpolate between the appearance embeddings associated with the left and right training photos. Note that the appearance of the rendered images smoothly transitions between the two end points without introducing artifacts to the 3D

Figure 7: Qualitative results from experiments on Phototourism dataset. NeRF-W is simultaneously able to model appearance variation, remove transient occluders, and capture a consistent 3D scene geometry. Photos by Flickr users jingjing, firewave, yatani / CC BY.

	Brandenburg Gate				Sacre Coeur				Trevi Fountain			
	↑ PSNR	↑ SSIM	↓ CT	↓ LPIPS	↑ PSNR	↑ SSIM	↓ CT	↓ LPIPS	↑ PSNR	↑ SSIM	↓ CT	↓ LPIPS
NRW [20]	23.85	0.914	0.091	0.141	19.39	0.797	0.233	0.229	20.56	0.811	0.225	0.242
NeRF	21.05	0.895	0.072	0.208	17.12	0.781	0.185	0.278	17.46	0.778	0.186	0.334
NeRF-A	27.96	0.941	0.063	0.145	24.43	0.923	0.159	0.174	26.24	0.924	0.154	0.211
NeRF-U	19.49	0.921	0.067	0.174	15.99	0.826	0.170	0.223	15.03	0.795	0.199	0.277
NeRF-W	**29.08**	**0.962**	**0.055**	**0.110**	**25.34**	**0.939**	**0.150**	**0.151**	**26.58**	**0.934**	**0.148**	**0.189**

Table 1: Quantitative results on the Phototourism dataset [13] for NRW [20], NeRF [21], and variations of the proposed model. Note that we report multiscale structural similarity for SSIM. Best results **highlighted**. NeRF-W outperforms the previous state of the art across all datasets and metrics by a significant margin.

geometry. Figure 9 shows reconstruction for a single image of NeRF and NeRF-W with the camera panning along a straight path. Unlike NeRF, reconstructions from NeRF-W are more view-consistent. We further notice less artifacts in

Figure 8: Interpolating between appearance embeddings $\boldsymbol{\ell}_i^{(a)}$ learned from two photos. Appearance embeddings $\boldsymbol{\ell}_i^{(a)}$ are taken from training photos on far left and right sides. All other images are renders from NeRF-W. Notice that people (left) and lights (right) do not appear in the renderings. Appearance embeddings modify a rendering's color and lighting without impacting 3D geometry. Photos by Flickr users mightyohm, blatez / CC BY.

(a) Reference (b) NeRF (c) NeRF-W

Figure 9: Epipolar plane images synthesized from reconstructions, where the camera is panning along a straight path. (a) reference rendering with epipole marked in red, (b) and (c) epipolar plane images for NeRF and NeRF-W. NeRF reconstructions contain severe ghosting artifacts in front of the landmark, whereas the NeRF-W reconstruction is view-consistent.

the image caused by transient objects in front of the landmark. We encourage readers to visit the project website to better appreciate the naturalness of such interpolations, `https://nerf-w.github.io/`.

5.2. Lego Dataset

To better understand the impact of each of NeRF-W's features, we apply NeRF-W to variations of the Lego dataset [21]. We introduce two classes of perturbations, tinting and occluders, to simulate the challenges we expect in the wild: photometric variation and transient objects (see Appendix D for additional information). Our setup is otherwise identical to Mildenhall et al [21]. We train on 100 images and evaluate on an additional 200, using the same NeRF model hyperparameters presented in the original work. Additional NeRF-W hyperparameters are tuned for each dataset variation via grid search.

Baseline: We begin by applying all methods to the original, unperturbed Lego dataset. Quantitatively, we find that all model variations perform similarly (Table 2). While NeRF achieves slightly higher PSNR than all NeRF-W variants, all other metrics suggest indistinguishable model quality. We find that our implementation of NeRF performs slightly better than that originally reported.

Color Perturbations To simulate appearance variation, we apply random tinting to each image in the training set. We find that this change alone decreases NeRF's PSNR by approximately 10dB on average (Table 2). As illustrated in Figure 10, NeRF is unable to isolate image-dependent photometric effects from its shared scene representation and thus entangles color variation with viewing direction. NeRF-A and NeRF-W, on the other hand, isolate tinting using the appearance embedding $\boldsymbol{\ell}^{(a)}$. Novel views rendered with a fixed appearance embedding demonstrate consistent color from all camera angles. Quantitatively, we find both methods maintain almost identical metrics to those achieved on the original dataset.

Random Occluders To simulate the effect of transient objects, we composite randomly-colored striped squares over each image in the training set. As shown in Table 2, this variation reduces NeRF's PSNR by 14dB on average. To reduce training error, NeRF and NeRF-A represent occluders as colored fog in 3D space, thereby causing the Lego figure to be obscured (Figure 10). While latent appearance embeddings were not designed to capture transient objects, we find that they enable NeRF-A to reduce error by learning a radiance field that imitates the color of the underlying 3D geometry. NeRF-A and NeRF-W are better able to isolate transient occluders from the static scene than their counterparts.

When both color and occluder perturbations are simultaneously enabled, we observe a decrease in performance across all methods, with NeRF-W ultimately outperforming its baselines. We further observe significant variance in model accuracy for both NeRF and NeRF-U across five random seeds. Both methods are poorly equipped to cope with photometric effects and occasionally fail to model the scene at all.

Figure 10: Example dataset perturbations and renderings from NeRF, NeRF-A, NeRF-U and NeRF-W. The leftmost column illustrates perturbations applied to the training dataset on a test image. All other columns show renderings from models trained on datasets with corresponding perturbations. NeRF-A and NeRF-U are largely able to disentangle color and occluder perturbations in isolation while NeRF-W is able to do so simultaneously. Render by Blender Swap user Heinzelnisse / CC BY.

6. Conclusion

We present NeRF-W, a novel approach for 3D scene reconstruction of complex outdoor environments from unstructured internet photo collections. Our method builds upon NeRF and learns a per-image latent embedding capturing photometric appearance variations often present in in-the-wild data. Further, we decompose the scene into shared and image-dependent components, thereby enabling

	ORIGINAL				COLOR PERTURBATIONS			
	↑ PSNR	↑ MSSSIM	↓ CT	↓ LPIPS	↑ PSNR	↑ MSSSIM	↓ CT	↓ LPIPS
NERF	**33.35**±0.05	**0.989**±0.000	**0.033**±0.000	**0.019**±0.000	23.38±0.05	0.964±0.001	0.039±0.000	0.076±0.001
NERF-A	33.04±0.06	**0.989**±0.000	**0.033**±0.000	0.020±0.000	30.66±1.38	0.983±0.007	0.038±0.006	0.031±0.015
NERF-U	33.07±0.27	**0.989**±0.001	**0.033**±0.000	**0.019**±0.001	24.87±0.52	0.968±0.000	0.039±0.001	0.063±0.007
NERF-W	32.89±0.14	**0.989**±0.000	**0.033**±0.000	0.020±0.001	**31.51**±0.28	**0.987**±0.001	**0.034**±0.000	**0.022**±0.001
	OCCLUDERS				COLORS PERTURBATIONS & OCCLUDERS			
	↑ PSNR	↑ MSSSIM	↓ CT	↓ LPIPS	↑ PSNR	↑ MSSSIM	↓ CT	↓ LPIPS
NERF	19.35±0.11	0.891±0.001	0.057±0.000	0.112±0.001	15.73±3.13	0.804±0.109	0.061±0.003	0.217±0.100
NERF-A	22.71±0.63	0.922±0.005	0.051±0.001	0.086±0.003	21.08±0.41	0.903±0.007	0.057±0.004	0.116±0.016
NERF-U	23.47±0.50	0.944±0.004	**0.045**±0.001	**0.059**±0.004	17.65±4.10	0.846±0.130	0.053±0.007	0.183±0.117
NERF-W	**25.03**±1.00	**0.946**±0.009	0.046±0.002	0.063±0.009	**22.19**±0.30	**0.927**±0.003	**0.050**±0.001	**0.087**±0.004

Table 2: Quantitative evaluation of NeRF and our proposed extensions on the synthetic Lego dataset. We report mean ± standard deviation across 5 independent runs with different random initializations. **Best** and second best results are highlighted. On the ORIGINAL dataset, all models perform near identically. NeRF fails to varying degrees on the perturbed datasets because it has no mechanism to account for those perturbations. As expected, NeRF-U fails on COLORS, but improves over NeRF on OCCLUDERS. Likewise, NeRF-A performs well on COLORS but fails on OCCLUDERS. NeRF-W is the only model that handles both types of perturbations.

Figure 11: Limitations exhibited by NeRF-W on the Phototourism dataset. Rarely seen parts of the scene (ground area, left) as well as failed camera registration (lamp post, right) can result in blurry regions.

our approach to isolate transient elements from the static scene. Extensive experimental evaluation on both synthetic and real-world data demonstrate significant qualitative and quantitative improvement over previous state-of-the-art approaches.

Nonetheless, the problem of outdoor scene reconstruction from image data remains far from being fully solved. While NeRF-W is able to produce photorealistic and temporally consistent renderings from unstructured photographs, the quality of the renderings degrade for areas of the scene that are very rarely covered in the captured images, as shown in Figure 11. Similar to NeRF, NeRF-W is also sensitive to camera calibration errors, which can lead to overly blurry reconstructions on certain parts of the scene. Overall, we believe that this work accomplishes significant strides towards generating novel views in outdoor unconstrained environments.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Richard Szeliski. Building rome in a day. *Commun. ACM*, 2011.

[2] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. *arXiv preprint arXiv:1906.08240*, 2019.

[3] Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the latent space of generative networks. *ICML*, 2018.

[4] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned raw denoising. *CVPR*, 2019.

[5] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. *SIGGRAPH*, 2001.

[6] Jimmy Ba Diederik P. Kingma. Adam: A method for stochastic optimization. *ICLR*, 2015.

[7] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. *CVPR*, 2019.

[8] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. *CVPR*, 2016.

[9] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, and Marc Pollefeys. Building rome on a cloudless day. *ECCV*, 2010.

[10] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 2009.

[11] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[13] 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. *arXiv preprint arXiv:2003.01587*, 2020.

[14] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[15] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *NeurIPS*, 2017.

[16] Pierre-Yves Laffont, Adrien Bousseau, Sylvain Paris, Frédo Durand, and George Drettakis. Coherent intrinsic images from photo collections. *SIGGRAPH Asia*, 2012.

[17] Marc Levoy and Pat Hanrahan. Light field rendering. *SIGGRAPH*, 1996.

[18] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *SIGGRAPH*, 2019.

[19] Ricardo Martin-Brualla, Rohit Pandey, Shuoran Yang, Pavel Pidlypenskyi, Jonathan Taylor, Julien Valentin, Sameh Khamis, Philip Davidson, Anastasia Tkach, Peter Lincoln, Adarsh Kowdle, Christoph Rhemann, Dan B Goldman, Cem Keskin, Steve Seitz, Shahram Izadi, and Sean Fanello. Lookingood: Enhancing performance capture with real-time neural re-rendering. *SIGGRAPH Asia*, 2018.

[20] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. *CVPR*, 2019.

[21] 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. *ECCV*, 2020.

[22] Thrasyvoulos N Pappas, Robert J Safranek, and Junqing Chen. Perceptual criteria for image quality evaluation. *Handbook of image and video processing*, 110, 2000.

[23] True Price, Johannes L Schönberger, Zhen Wei, Marc Pollefeys, and Jan-Michael Frahm. Augmenting crowd-sourced 3d reconstructions using semantic detections. *CVPR*, 2018.

[24] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. *CVPR*, 2016.

[25] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. *CVPR*, 2006.

[26] Qi Shan, Riley Adams, Brian Curless, Yasutaka Furukawa, and Steven M Seitz. The visual turing test for scene reconstruction. *3DV*, 2013.

[27] Heung-Yeung Shum, Shing-Chow Chan, and Sing Bing Kang. *Image-based rendering*. Springer Science & Business Media, 2008.

[28] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. *CVPR*, 2019.

[29] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *NeurIPS*, 2019.

[30] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo Tourism: Exploring photo collections in 3D. *SIGGRAPH*, 2006.

[31] Fridtjof Stein. Efficient computation of optical flow using the census transform. *Joint Pattern Recognition Symposium*, 2004.

[32] Hossein Talebi and Peyman Milanfar. NIMA: Neural image assessment. *IEEE TIP*, 2018.

[33] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains, 2020.

[34] Ayush Tewari, Christian Theobalt, Dan B Goldman, Eli Shechtman, Gordon Wetzstein, Jason Saragih, Jun-Yan Zhu, Justus Thies, Kalyan Sunkavalli, Maneesh Agrawala, Matthias Niessner, Michael Zollhöfer, Ohad Fried, Ricardo Martin Brualla, Rohit Kumar Pandey, Sean Fanello, Stephen Lombardi, Tomas Simon, and Vincent Sitzmann. State of the art on neural rendering. *Computer Graphics Forum*, 2020.

[35] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *SIGGRAPH*, 2019.

[36] Kim-Han Thung and Paramesran Raveendran. A survey of image quality measures. *TECHPOS*, 2009.

[37] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. *International Workshop on Vision Algorithms*, 1999.

[38] Zhou Wang and Alan C Bovik. A universal image quality index. *IEEE Signal Processing Letters*, 2002.

[39] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. *Asilomar Conference on Signals, Systems & Computers*, 2003.

[40] Ramin Zabih and John Woodfill. Non-parametric local transforms for computing visual correspondence. *ECCV*, 1994.

[41] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *CVPR*, 2018.

Appendices

A. Approximations to Volumetric Rendering Equations for NeRF-W

We present here numerical quadrature approximations to the integrals presented in Section 4.2. We begin by approximating the expected color of a camera ray $\mathbf{r}$ as described in Equations (10) and (11). Let $\{t_k\}_{k=1}^K$ be quadrature points sampled between t_n and t_f and $\delta_k = t_{k+1} - t_k$ We approximate expected color $\bar{\mathbf{C}}_i(\mathbf{r})$ with:

$$\hat{\mathbf{C}}_i(\mathbf{r}) = \sum_{k=1}^{K} \hat{T}_i(t_k) \left(\alpha(\sigma(t_k)\delta_k)\, \mathbf{c}_i(t_k) + \alpha\left(\sigma_i^{(\tau)}(t_k)\delta_k\right) \mathbf{c}_i^{(\tau)}(t_k) \right), \tag{19}$$

$$\text{where} \quad \hat{T}_i(t_k) = \exp\left(- \sum_{k'=1}^{k-1} \left(\sigma(t_k) + \sigma_i^{(\tau)}(t_k)\right)\delta_k \right), \tag{20}$$

and $\alpha(x) = 1 - \exp(-x)$.

We approximate the variance of a camera ray $\mathbf{r}$ as described in Equations (13) and (14) with,

$$\hat{\beta}_i(\mathbf{r}) = \sum_{k=1}^{K} \hat{T}_i^{(\tau)}(t_k)\, \alpha\left(\sigma_i^{(\tau)}(t_k)\delta_k\right) \beta_i(t_k), \tag{21}$$

$$\text{where} \quad \hat{T}_i^{(\tau)}(t_k) = \exp\left(- \sum_{k'=1}^{k-1} \sigma_i^{(\tau)}(t_k)\delta_k \right). \tag{22}$$

B. Model Parameters & Evaluation

Hyperparameters: We document the selected hyperparameters and procedure for their selection in an upcoming version of this article.

Evaluation: For evaluation, we use the AlexNet variant of LPIPS as implemented in `https://github.com/richzhang/PerceptualSimilarity`. We further use a variant of CT [31] with a tolerance parameter $\epsilon = 0.05$ such that the transform is less sensitive to noise, and we report the mean average deviation between the census transformation of both images over all pixels and color channels.

For the models trained on in-the-wild photo collections, we evaluate NeRF-A and NeRF-W by optimizing appearance embeddings $\boldsymbol{\ell}^{(a)}$ on the *left half* of each ground-truth image and compute metrics on the *right half*. While this may raise the concern of overfitting, we argue it to be a necessary step to accurately measure rendered image quality in our setting. Unlike controlled settings, image generation from a fixed camera viewpoint in-the-wild is fundamentally an ill-posed problem [20]. Changes to time-of-day, lighting, and post-processing all result in valid variations of the same photo. An ideal metric for measuring generated image quality would be invariant to such effects. We fashion an approximation to such a metric by choosing an appearance embedding according to the same process used for images in the training set. To prevent information leakage, we further ensure that the portion of the image used for identifying the embedding is distinct from that used for evaluation. In our experiments, appearance embeddings are 48-dimensional and are unable to alter the 3D geometry of the scene, see Figures 7 and 8.

A valid concern when comparing NRW to NeRF-W and its variants is that the former employs an appearance encoding network while the former uses an optimization process. While a network is less powerful than direct optimization, it limits NRW's ability to encode the full image in a latent code. Unlike NeRF-W, it is not possible to directly design an image-to-image translation model in such a way as to enforce independence of 3D geometry and appearance, and thus directly optimizing a latent code with NRW may result in "hallucinating" objects not appearing in its input point cloud representation. We design NeRF-W such that appearance embeddings cannot impact the static scene geometry and experimentally do not observe divergent behavior between the left and right halves of rendered images.

Figure 12: Sample of the filtered images during the preprocessing step. Images where transient objects occupy more than 80% of the image or where NIMA score is below a certain threshold are discarded from the scene data. Photos by Flickr users alcanthus, headnut, uwehiksch, and stevebaty / CC BY

C. Phototourism Dataset

As a coarse pre-filtering step, we remove low quality images largely consisting of transient objects by dropping the lowest portion of images as ranked automatically by NIMA [32], using a minimum score threshold of 3.0. In addition to this, we filter out images where the transient objects occupy more than 80% of the image. Figure 12 depicts some examples of the filtered images from the Brandenburg Gate

	Train		Validation	
Dataset	Images	Pixels	Images	Pixels
Brandenburg Gate	763	564 M	38	12 M
Sacre Coeur	830	605 M	40	14 M
Trevi Fountain	1,689	1,249 M	39	14 M

Table 3: Number of images and pixels per Phototourism scene. Pixel counts measuring in millions.

scene.

For quantitative evaluation, we form a test set by hand-selecting photos representative of the qualities we intend to replicate: well-focused and without occluders. While a naive random selection of images may seem appropriate, image comparison metrics such as PSNR, MS-SSIM, LPIPS, and CT are unable to ignore transient objects. Indeed, NeRF-W is designed to generate images *without* such occluders, and will thus be at an unfair disadvantage when the reference contains them. We thus explicitly select photos without transient phenomena or extreme photometric effects. Photos constituting the test set were chosen during the preliminary experiments stage and held-out until the final evaluation shown in Table 1. In particular, the chosen photos were not used to guide model design or hyperparameter search. We provide the names of all photos selected for training and test on the project website, `https://nerf-w.github.io/`. See Table 3 for scene-specific statistics on this dataset.

D. Lego Dataset

Color perturbations To simulate variable lighting and exposure, we apply a random affine transformation to the RGB values of each image. In particular, we replace each training image $\mathcal{I}_i \in [0,1]^{800\times800\times3}$ with $\tilde{\mathcal{I}}_i = \min(1, \max(0, \mathbf{s}_i\mathcal{I}_i + \mathbf{b}_i))$ where scale $\mathbf{s}_{ij} \sim \mathcal{U}(0.8, 1.2)$ and offset $\mathbf{b}_{ij} \sim \mathcal{U}(-0.2, 0.2)$ are sampled uniformly at random for each image i and RGB color channel $j \in \{1,2,3\}$. Qualitatively, this results in variable tint and brightness. We apply perturbations to all training images except the first, whose appearance embedding is used to render novel views. The top row of Figure 13 shows the effect of applying random color perturbations to the same image.

Occlusions We simulate transient occluders by drawing randomly-positioned and randomly-colored squares on each training image. Each square consists of ten vertical, colored stripes with colors chosen at random. Like transient occluders in the real world, these squares do not have a consistent 3D location from image to image. We again leave the first training image untouched for reference. Figure 13 shows the effect of adding occlusions randomly to the same view.

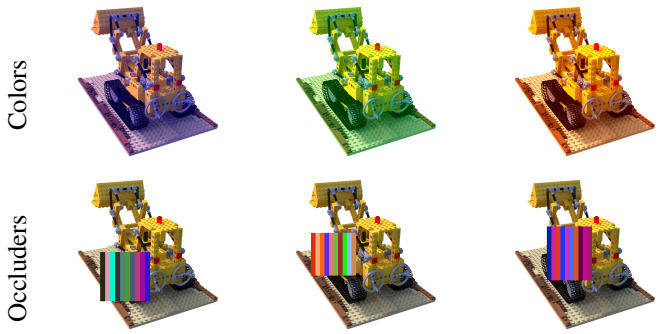


Figure 13: Example of perturbations applied to the Lego dataset. The top row shows various color perturbations applied to the same view, whereas the bottom shows the effect of randomly adding occluders to the same view.