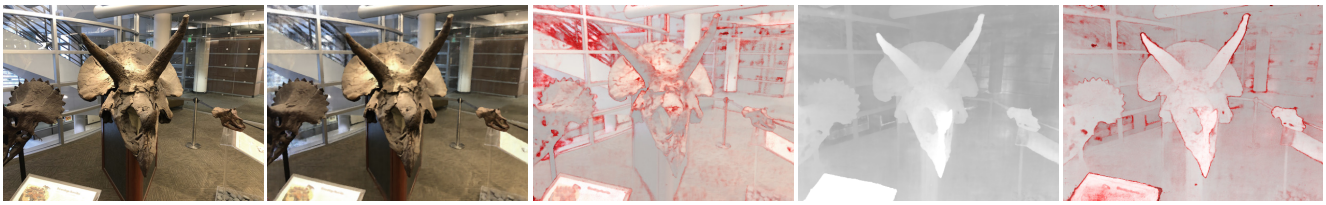
Stochastic Neural Radiance Fields: Quantifying Uncertainty in Implicit 3D Representations

Jianxiong Shen Adria Ruiz Antonio Agudo Francesc Moreno-Noguer
Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain
{jshen,aruiz,aagudo,fmoreno}@iri.upc.edu



Ground-Truth Rendered novel view RGB-Uncertainty Depth Depth-Uncertainty

Figure 1. **Illustration of the results obtained by Stochastic Neural Radiance Fields (S-NeRF).** Our method is a probabilistic generalization of the original NeRF, which is able to not only address tasks such as novel-view generation (Rendered novel view) or depth-map estimation (Depth), but also quantify the uncertainty (red color) associated with the model outputs. This is specially important in domains such as robotics, where this information is necessary to evaluate the risk associated with decisions based on the model estimations.

Abstract

Neural Radiance Fields (NeRF) has become a popular framework for learning implicit 3D representations and addressing different tasks such as novel-view synthesis or depth-map estimation. However, in downstream applications where decisions need to be made based on automatic predictions, it is critical to leverage the confidence associated with the model estimations. Whereas uncertainty quantification is a long-standing problem in Machine Learning, it has been largely overlooked in the recent NeRF literature. In this context, we propose Stochastic Neural Radiance Fields (S-NeRF), a generalization of standard NeRF that learns a probability distribution over all the possible radiance fields modeling the scene. This distribution allows to quantify the uncertainty associated with the scene information provided by the model. S-NeRF optimization is posed as a Bayesian learning problem that is efficiently addressed using the Variational Inference framework. Exhaustive experiments over benchmark datasets demonstrate that S-NeRF is able to provide more reliable predictions and confidence values than generic approaches previously proposed for uncertainty estimation in other domains.

1. Introduction

Recent learning based methods have shown impressive results in 3D modeling. In particular, implicit neural volume rendering [27, 31, 22, 28] has become a popular framework to learn compact 3D scene representations from a sparse set of images. Among these methods, Neural Radiance Fields (NeRF) [28] has received a lot of attention given its ability to render photo-realistic novel views of the scene. Additionally, several works have shown that the 3D representations learned by NeRF can be used for different downstream tasks, such as camera-pose recovery [38], 3D semantic segmentation [16] or depth estimation [28]. Even though all these tasks have relevant applications in fields such as robotics or augmented reality, existing NeRF-based approaches are limited in these scenarios, for being unable to provide information about the confidence associated with the model outputs. For instance, consider a robot using Neural Radiance Fields to reason about its environment. In order to plan the optimal actions and reduce potential risks, the robot must take into account not only the outputs produced by NeRF, but also their associated uncertainty.

In this context, we propose Stochastic-NeRF, a generalization of the original NeRF framework able to quantify the uncertainty associated with the implicit 3D representation. Unlike standard NeRF which only estimates deterministic radiance-density values for all the spatial-locations in the scene, S-NeRF models these pairs as stochastic variables following a distribution whose parameters are optimized during learning. In this manner, our method implicitly encodes a distribution over all the possible radiance fields modeling the scene. The introduction of this stochasticity enables S-NeRF to quantify the uncertainty associated with the the resulting outputs in different tasks such

as novel-view rendering or depth-map estimation (see Figure 1). During learning, we follow a Bayesian approach to estimate the posterior distribution of all the possible radiance fields given training data. To make this optimization problem tractable, we devise a learning procedure for S-NeRF based on Variational Inference [5]. Conducting exhaustive experiments over benchmark datasets, we show that S-NeRF is able to provide more reliable uncertainty estimates than generic approaches previously proposed for uncertainty estimation in other domains. In particular, we evaluate the ability of S-NERF to quantify the uncertainty in novel-view synthesis and depth-map estimation.

2. Related Work

Neural Radiance Fields. Similar to other neural volumetric approaches such as Scene Representation Networks [35] or Neural Volumes [22], NeRF uses a collection of sparse 2D views to learn a neural network encoding an implicit 3D representation of the scene. NeRF employs a simple yet effective approach where the network predicts the volume density and emitted radiance for any given view-direction and spatial coordinate. These outputs are then combined with volume rendering techniques [26] to synthesize novel views or estimate the implicit 3D geometry of the scene.

Since it was firstly introduced, many works have extended the original NeRF framework to address some of its limitations. For instance, [21, 20, 29, 23] explored several techniques to accelerate the time-consuming training and rendering process. Other works [17, 39] introduced the notion of "scene priors", allowing a single NeRF model to encode the information of different scenes and generalize to novel ones. Similarly, [25] proposed to account for illumination changes and transient occluders in order to leverage in-the-wild training views. Other recent works [33, 8, 34, 19, 37, 32] have extended NeRF to cases with dynamic objects in the scene.

Different from the aforementioned methods, the proposed S-NeRF explicitly addresses the problem of estimating the uncertainty associated with the learned implicit 3D representation. Our framework is a probabilistic generalization of original NeRF and thus, our formulation can be easily combined with most of previous works in order to improve different aspects of the model.

NeRF Applications. Implicit representations learned by NeRF can be used to infer useful scene information for domains such as Robotics or Augmented Reality (AR). For instance, the ability to render novel views, estimate 3D meshes [28, 33] or recover camera-poses[38] can be used to allow robots to reason about the environment and plan navigation paths or object manipulations. Additionally, [30] proposed to incorporate a compositional 3D scene representation into a generative model to achieve controllable novel view synthesis. This capability is specially interesting in AR scenarios. Despite these potential applications, previous NeRF approaches are still limited in the aforementioned domains. The reason is that they are not able to quantify the underlying uncertainty of the model and thus, it is not possible to evaluate the risk associated with downstream decisions based on the output estimations.

Recently, NeRF-in-the-Wild (NeRF-W) [25] considered to identify the uncertainty produced by transient objects in the scene such as pedestrians or vehicles. In particular, the authors proposed to estimate a value indicating the variance for each rendered pixel in a novel synthetic view. This variance is computed by treating it as an additional value to be rendered analogously to pixel RGB intensities. However, this approach has two critical limitations. Firstly, pixel-colors are produced by a specific physical process that is not related to the model uncertainty. As a consequence, estimating the latter with volume rendering techniques is not theoretically-founded and can lead to sub-optimal results. On the other hand, while NeRF-W is able to predict the variance associated with the rendered pixels, it does not explicitly model the uncertainty of the radiance field representing the scene. Hence, it can not quantify confidence estimates about the underlying 3D geometry.

To the best of our knowledge, S-NeRF is the first approach to explicitly model the uncertainty of the implicit representation learned by NeRF. In contrast to NeRF-W, our method allows us to quantify the uncertainty associated not only with rendered views, but also with estimations related with the 3D geometry (see Fig. 1 again).

Uncertainty Estimation is a long-standing problem in Deep Learning [1, 10, 13, 9, 18]. To address it, a popular approach imposes the Bayesian Learning framework [4] to estimate the posterior distribution over the model given observed data. This posterior distribution can be used during inference to quantify the uncertainty of the model outputs. In the context of deep learning, Bayesian Neural Networks [36, 7, 13, 24] use different strategies to learn the posterior distribution of the network parameters given the training set. However, these approaches are typically computationally expensive and require significant modifications over network architectures and training procedures.

To address this limitation, other approaches have explored strategies to implicitly learn the parameter distribution. For instance, dropout-based methods [9, 3, 12, 6] introduce stochasticity over the intermediate neurons of the network in order to efficiently encode different possible solutions in the parameter space. By evaluating the model with different dropout configurations over the same input, the uncertainty can be quantified by computing the variance over the set of obtained outputs. A similar strategy consists on using deep ensembles [18, 14], where a finite set of independent networks are trained and evaluated in order

to measure the output variance. Whereas these solutions are more simple and efficient than Bayesian Neural Networks, they still require multiple model evaluations. This limits their application in NeRF, where the rendering process is already computationally expensive for a single model.

Different from the previous approaches learning a posterior distribution over the model parameters, the proposed S-NeRF learns a single network encoding the distribution over all the possible radiance fields modelling the scene. As we will discuss in the following sections, this allows to efficiently obtain uncertainty estimates without the need of evaluating multiple model instances.

3. Stochastic Neural Radiance Fields

3.1. From Standard to Stochastic-NeRF

Standard NeRF [28] models a 3D volumetric scene as a radiance field F defining a set:

$$F = f(\mathbf{r}(\mathbf{x}; \mathbf{d}), \sigma(\mathbf{x})) : \mathbf{x} \in \mathbb{R}^3, \mathbf{d} \in \mathbb{R}^2 g \tag{1}$$

where $\sigma(\mathbf{x}) \in \mathbb{R}^+$ is the volume density in a specific 3D spatial-location $\mathbf{x}$ and $\mathbf{r}(\mathbf{x}; \mathbf{d}) \in \mathbb{R}^3$ is the emitted RGB radiance which is also dependent on the view direction $\mathbf{d}$.

To model the radiance field F, NeRF uses a parametric function $f(\mathbf{x}; \mathbf{d}) : \mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}^4$ which encodes the radiance $\mathbf{r}$ and density σ for every possible location-view pair $(\mathbf{x}; \mathbf{d})$ in the scene. Concretely, this function is implemented by a deep neural network with parameters θ.

NeRF Optimization: For a given scene, NeRF optimizes the network f_θ by leveraging a training set $T = \{(\mathbf{c}^1; \mathbf{x}_o^1; \mathbf{d}^1); \ldots; (\mathbf{c}^N; \mathbf{x}_o^N; \mathbf{d}^N)\}$ formed by N triplets, where $\mathbf{c}^n$ is a RGB pixel-color captured by a camera located in a 3D position $\mathbf{x}_o^n$ in the scene. Additionally, $\mathbf{d}^n$ is the normalized direction from the camera origin to the pixel in world-coordinates. This training set can be obtained by capturing a collection of views from the scene using different cameras with known poses.

By assuming that training samples $(\mathbf{c}^n; \mathbf{x}_o^n; \mathbf{d}^n) \in T$ are independent observations, the network parameters θ are optimized by minimizing the negative log-likelihood as:

$$\min_\theta -\log p_\theta(T) = \frac{1}{N} \sum_{n=1}^{N} -\log p_\theta(\mathbf{c}^n | \mathbf{x}_o^n; \mathbf{d}^n) \propto \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{c}^n - C(\mathbf{x}_o^n; \mathbf{d}^n)\|_2^2 \tag{2}$$

where the quadratic error follows from defining $p_\theta(\mathbf{c}_n^p | \mathbf{x}_n^o; \mathbf{d}_n)) = \mathcal{N}(\mathbf{c}_n^p | \cdot\,; \cdot)$ as a Gaussian distribution with unit variance and a mean defined by the volumetric rendering function:

$$C(\mathbf{x}_o; \mathbf{d}) = \int_{t_s}^{t_f} \frac{\mathbf{r}_t \, \sigma_t}{\exp(\int_{t_s}^{t} \sigma_s ds)} dt \tag{3}$$

where $\mathbf{x}_t = \mathbf{x}^o + t\mathbf{d}$ is a specific spatial-location along a ray with direction $\mathbf{d}$ which crosses the scene from the pixel position in world-coordinates $\mathbf{x}_{t_s}$ to the point $\mathbf{x}_{t_f}$. Additionally, we express $\mathbf{r}(\mathbf{x}_t; \mathbf{d})$ and $\sigma(\mathbf{x}_t)$ as $\mathbf{r}_t$ and σ_t, respectively. More details about the volumetric rendering function in Eq. (3) and how it is approximated can be found in [28].

Stochastic NeRF is a generalization of the previously described framework. Specifically, instead of learning a single radiance field F, S-NeRF models a distribution $p(F)$ over all the possible fields modelling the scene. For that purpose, we consider that for each location-view pair $(\mathbf{x}; \mathbf{d})$, the volume density and emitted radiance are random variables following an unknown joint distribution. In this manner, any radiance field F defined in Eq. (1) can be considered a realization over the distribution $p(F)$. As we will discuss in Sec. 3.3, treating the radiance field as a set of stochastic variables allows to reason about the underlying uncertainty in the implicit 3D representation.

S-NeRF Optimization: Different from the optimization strategy used in original NeRF, S-NeRF adopts a Bayesian approach where the goal is to estimate the posterior distribution of the possible radiance fields F given the observed training set T:

$$p(F|T) = \frac{p(T|F)p(F)}{p(T)} \tag{4}$$

where $p(T|F)$ is the likelihood of T given a radiance field and $p(F)$ is a distribution modelling our prior knowledge about the radiance and density pairs over the different spatial-locations in the scene.

3.2. Learning S-NeRF with Variational Inference

Given that the explicit computation of the posterior in Eq. (4) is intractable, we employ variational inference [5] in order to approximate it. In particular, we define a parametric distribution $q_\theta(F)$ approximating the true posterior and optimize its parameters θ by minimizing the Kullback-Leibler (KL) divergence between both as:

$$\min_\theta \mathrm{KL}(q_\theta(F) \,\|\, p(F|T)) \propto \mathbb{E}_{q_\theta(F)} \log(p(T|F)) + \mathbb{E}_{q_\theta(F)} \log \frac{q_\theta(F)}{p(F)} \tag{5}$$

Intuitively, the first term in Eq. (5) measures the expected training set likelihood over the radiance field distribution $q_\theta(F)$. On the other hand, the second term measures the KL divergence between the approximate posterior and the prior distribution $p(F)$. In the following, we detail how S-NeRF addresses this optimization problem.

Figure 2. Illustration of the pipeline used by S-NeRF to compute the log-likelihood of the pixel color in a given view. From left to right: (i) For each coordinate x along a camera ray with viewing direction d, a neural network predicts the parameters of the radiance and density distributions. (ii) For each spatial-location, we sample a set of radiance-density pairs from the previous distributions. (iii) This generates a set of different radiance-density trajectories along the ray obtained from a distribution of radiance fields. (iiii) The volume rendering equations are used to estimate the RGB values for each trajectory and compute the log-likelihood for a given pixel color. During inference, the mean and variance of these samples are used as the model prediction and to quantify its associated uncertainty.

3.2.1 Modelling the approximate posterior

In order to make the approximate posterior $q(F)$ tractable, we define it as a fully-factorized distribution:

$$q(F) = \prod_{x \in \mathbb{R}^3} \prod_{d \in \mathbb{R}^2} q(r|x,d)\, q(\sigma|x) \tag{6}$$

where we assume that the density and radiance are independent variables given any location-view pair (x, d). In particular, S-NeRF models $q(F)$ with a neural network defining a function $f(x,d) = \{\mu_r, \sigma_r, \mu_\sigma, \sigma_\sigma\}$, where $\mu_r \in \mathbb{R}^3$ and $\sigma_r \in \mathbb{R}^+$ are a mean and standard deviation defining the distribution $q(r|x,d)$. Similarly, $\mu_\sigma \in \mathbb{R}$ and $\sigma_\sigma \in \mathbb{R}^+$ define the density distribution $q(\sigma|x)$.

Given that the radiance values need to be bounded between 0 and 1, we use a logistic normal distribution [2] for each RGB channel r^c independently. In this manner, r^c is a random variable defined by:

$$r^c = \text{Sigmoid}(\hat{r}^c), \quad \hat{r}^c \sim \mathcal{N}(\hat{r}^c | \mu_{r^c}, \sigma_{r^c}) \tag{7}$$

resulting from applying a sigmoid function to a Gaussian variable $\mathcal{N}$ with mean μ_{r^c} and standard deviation σ_{r^c}. Similarly, the support for the density distribution $q(\sigma|x,d)$ needs to be positive. Therefore, we model σ as a random variable following a rectified normal distribution [11]:

$$\sigma = \max(0, \hat{\sigma}), \quad \hat{\sigma} \sim \mathcal{N}(\hat{\sigma} | \mu_\sigma, \sigma_\sigma) \tag{8}$$

where $\max(0, \cdot)$ is as a rectified linear unit that sets all the negative values to 0.

Both distributions are illustrated in Figure 3 and have a tractable analytical form. Additionally, it is easy to sample from them by applying the Sigmoid or ReLU operation to a variable obtained from a normal distribution with parameters $\{\mu_r, \sigma_r\}$ and $\{\mu_\sigma, \sigma_\sigma\}$, respectively. See Appendix A.1 for more details.

3.2.2 Computing the log-likelihood

In the following, we introduce how S-NeRF computes the likelihood term in Eq. (5). Firstly, note that given the variational posterior $q(F)$ and the training set $\mathcal{T}$, $\mathbb{E}_{q(F)} \log(p(\mathcal{T}|F))$ is equivalent to:

$$\frac{1}{N} \sum_{n=1}^{N} \mathbb{E}_{q(r^n_{t_s:t_f}, \sigma^n_{t_s:t_f} | x^n_{t_s:t_f}, d^n)} \log(p(c^n | r^n_{t_s:t_f}, \sigma^n_{t_s:t_f})) \tag{9}$$

where: (i) $x^n_{t_s:t_f} = \{o^n + t d^n : t \in [t_s, t_f]\}$ is the set of 3D coordinates along a ray with direction d^n and origin $x^n_{t_s}$, (ii) $\{r^n_{t_s:t_f}, \sigma^n_{t_s:t_f}\}$ is a set of radiance-density pairs for each ray position and (iii) $p(c^n | r^n_{t_s:t_f}, \sigma^n_{t_s:t_f})$ is the probability of the pixel color c^n given the radiance and density values accumulated along the ray. The latter probability is defined similarly to standard NeRF, where we assume that c^n follows a normal distribution with a mean defined by applying the volumetric rendering function Eq. (3) to the radiance-density trajectory $\{r^n_{t_s:t_f}, \sigma^n_{t_s:t_f}\}$ along the ray.

Given previous definitions, Eq. (9) can be computed using a Monte-Carlo approximation:

$$\frac{1}{K} \sum_{k=1}^{K} \log(p(c^n | r^{nk}_{t_s:t_f}, \sigma^{nk}_{t_s:t_f})) \tag{10}$$

Figure 3. Probability Density Functions for (a) logistic normal (b) and rectified normal distributions with different mean and variance parameters. In S-NeRF the logistic-Normal is used to model the distribution over the radiance values given that its support is bounded between 0 and 1. The volume density distribution is defined by the rectified normal whose support is $[0, \infty]$.

where each $\mathbf{r}_t^{nk}$ is a sample from the radiance distribution $q(\mathbf{r}|\mathbf{x}_t^n; \mathbf{d}^n)$. These samples can be generated using Eq. (7) with parameters $\{\mu_r, \sigma_r\}$ obtained by evaluating the network $f(\mathbf{x}_t^n; \mathbf{d}^n)$. Similarly, σ_t^{nk} is a sample from the volume density distribution using Eq. (8) with mean and variance parameters also defined by the network output. An illustration of the whole process is provided in Figure 2. The introduced strategy is used during training to compute the log-likelihood and apply stochastic gradient descent to optimize the parameters. This is possible by using the reparametrization-trick [15] to back-propagate the gradients through the generated samples $\mathbf{r}_t^{nk}$ and σ_t^{nk}. See Appendix A.2 for a more detailed explanation.

3.2.3 Estimating the posterior-prior KL divergence

As previously discussed, the KL term in Eq. (5) measures the difference between the approximated posterior over the radiance fields F learned by S-NeRF and a prior distribution. Similarly to the definition of $q(F)$ in Eq. (6), we model the prior $p(F) = \prod_{\mathbf{x} \in \mathbb{R}^3} \prod_{\mathbf{d} \in \mathbb{R}^2} p(\mathbf{r}) p(\sigma)$ as a fully-factorized distribution,

where the radiance and density priors are assumed to be the same for all the spatial-locations in the scene. Concretely, $p(\mathbf{r})$ is again modelled with a logistic normal distribution as in the case of $q(\mathbf{r}|\mathbf{x}; \mathbf{d})$. In this case, however, the mean parameter is optimized during training and its variance is fixed to 10. This high value models our knowledge that, without considering any observation, the uncertainty over the radiance values must be high. Analogously, $p(\sigma)$ is modelled with a rectified-normal distribution also with an optimized mean and fixed variance ($=10$).

Given previous definitions, the KL term in Eq. (5) can be expressed as:

$$E_{q(F)} \log \frac{q(F)}{p(F)} = \sum_{\mathbf{x} \in \mathbb{R}^3} \sum_{\mathbf{d} \in \mathbb{R}^2} KL(q(\mathbf{r}|\mathbf{x}; \mathbf{d}) \,\|\, p(\mathbf{r})) + \sum_{\mathbf{x} \in \mathbb{R}^3} \sum_{\mathbf{d} \in \mathbb{R}^2} KL(q(\sigma|\mathbf{x}) \,\|\, p(\sigma)) \quad (11)$$

which is equivalent to the sum of the KL divergence between the posterior and prior distributions for all the possible location-view pairs in the scene. During training, Eq. (11) is minimized by sampling random location-view pairs $(\mathbf{x}; \mathbf{d})$ in 3D and approximating the radiance and density KL terms using automatic integration. See Appendix A.3 for more details.

In our preliminary experiments, we observed that it is also beneficial to consider a different prior distributions for location-view pairs belonging to the rays traced to estimate the pixel-color likelihood in Eq. (9). The reason is that, in these locations, we know that the provided observations are reducing the uncertainty about the radiance-density. Therefore, setting a prior with a high variance for the distributions contradicts this prior knowledge. For these reasons, we also compute the KL term for the spatial-locations sampled along these rays but, in this case, the variances defining the prior distributions $p(\mathbf{r})$ and $p(\sigma)$ are not fix but also optimized during training. A pseudo-algorithm summarizing the learning process of S-NeRF is provided in Appendix B.

3.3. Inference and Uncertainty Estimation

By learning a distribution over radiance fields, S-NeRF is able to quantify the uncertainty associated with rendered views for any given camera pose. For this purpose, we firstly sample a set of K color values $\mathbf{c}^k$ for each pixel in the rendered image. As illustrated in Figure 2, these values are obtained by applying the volume rendering equation to different radiance-density trajectories $\{\mathbf{r}_{t_s:t_f}^k, \sigma_{t_s:t_f}^k\}$ along a ray traced from the pixel coordinates. Intuitively, each of the sampled colors $\mathbf{c}^k$ represents an estimate produced by a single radiance field in the learned distribution. Finally, we treat the mean and variance over the K samples $\mathbf{c}^k$ as the predicted pixel color and its associated uncertainty.

Similar to the case of image rendering, S-NeRF is also able to quantify the uncertainty associated with estimated depth-maps. In this case, we ignore the radiance values and use K trajectories $\{\sigma_{t_s:t_f}^k\}$ obtained by sampling density values along the ray. Then, for each sampled trajectory, we compute the expected termination depth of the ray as in [28]. In this way, we can get K samples for each pixel in the depth maps. The mean and variance of these samples correspond to the estimated depth and its uncertainty.

4. Experiments

4.1. Experimental setup

Datasets. We conduct our experiments over the LLFF benchmark dataset introduced in [28]. It contains multiple views with calibrated camera poses for 8 different scenes including indoor (Horns, Trex, Room, Fortress, Fern) and outdoor environments (Flower, Leaves, Orchids). Given that our goal is to evaluate the reliability of the quantified

	Neg. Log. Likelihood (NLL) #				MSE-Uncertainty Correlation "			
	MC-DO [9]	D. Ens. [18]	NeRF-W [25]	S-NeRF	MC-DO [9]	D. Ens. [18]	NeRF-W [25]	S-NeRF
Flower	4.63	1.63	1.71	1.27	0.38	0.59	0.49	0.63
Frotress	5.19	2.29	1.04	-0.03	0.24	0.37	0.44	0.55
Leaves	2.72	2.66	0.79	0.68	0.39	0.57	0.65	0.73
Horns	4.18	2.17	0.78	0.60	0.43	0.50	0.50	0.70
Trex	4.10	2.28	1.91	1.37	0.42	0.53	0.66	0.68
Fern	4.90	2.47	2.16	2.01	0.50	0.65	0.59	0.69
Orchids	5.74	2.23	2.24	1.95	0.50	0.60	0.60	0.65
Room	5.06	2.13	4.93	2.35	0.46	0.65	0.38	0.74
Avg.	4.57	2.23	1.95	1.27	0.40	0.56	0.54	0.67

Table 1.Results for the uncertainty estimation metrics obtained by all the evaluated methods on the NeRF LLFF dataset [28].

	MC-DO [9]	D. Ens. [18]	NeRF-W [25]	S-NeRF
Inference (sec.)	38.26	34.77	7.08	6.06

Table 2.Rendering time(s) required by the evaluated methods for a single view with resolution 500400.

uncertainty, we use a more reduced number of scene views during training compared to the experimental setups used in the original paper. The rationale is the following: in low-data regimes, uncertainty estimation is of particular importance given that the model should be able to identify the parts of the scene that are not covered by the training. In these cases, the model is expected to automatically assign a high uncertainty to these regions. Motivated by this observation, we randomly choose only a 20% of the total views for training and use the rest for testing.

Baselines. As discussed in Sec. 2, only NeRF-W [25] has attempted to quantify uncertainty in Neural Radiance Fields. For this reason, we also compare S-NeRF with state-of-the-art approaches that have been proposed in other domains for the same purpose. In particular, we consider MC-dropout [9] and Deep-Ensembles [18]. In the first case, we add a dropout layer after each odd layer in the network to sample multiple outputs using random dropout configurations. Considering a trade-off between computation and performance, we use five samples in our experiments and compute their variance as the uncertainty value.

On the other hand, in Deep-Ensembles we train and evaluate five different NeRF models in parallel. Again, the variance of their outputs is used as the uncertainty associated with the prediction. Finally, we also compare S-NeRF with the proposed strategy used in NeRF-W [25] for uncertainty estimation. Given that there are no variable illumination or moving objects in the scene of the evaluated datasets, we remove the latent embedding component of their approach and keep only the uncertainty estimation layers.

Evaluation Metrics. Previous works typically evaluate the rendered novel-views using image-quality metrics such as PSNR, SSIM, and LPIPS. However, these validation criteria are not informative in our context given that we aim to measure the reliability of the uncertainty estimates. For this reason, we use two alternative metrics: the negative log-likelihood (NLL) and the correlation between the Mean Squared Error (MSE) and the obtained uncertainty values. The use of the NLL is motivated by the observation that all the evaluated methods provide uncertainty estimations based on a predicted variance for each estimated pixel color. In this manner, we can compute the NLL for each pixel by computing the probability of the ground-truth given a Gaussian distribution with mean equivalent to the estimated color and variance equal to the predicted uncertainty. More intuitively, this metric measures the average MSE error with respect to the color gound-truth weighted by the model confidence associated with each pixel color. In the second metric, we compute the correlation between the MSE error for each pixel and the estimated uncertainty values. Note that this correlation will be better if the model assigns higher uncertainty to estimations that are more likely to be inaccurate. Therefore, this metric indicates whether the uncertainty estimates can be used as a predictive value for the expected error in real scenarios where no ground-truth is available.

Implementation details. To implement the different compared baselines, we use the same network architecture, hyper-parameters and optimization process employed in the original NeRF paper[1]. For S-NeRF implementation, we also use the same architecture to implement the function $f(x;d)$. The only introduced modification is in the last layer, where we double the number of outputs to account for the mean and variance parameters of the density and radiance distributions. During training, we uniformly sample 128 spatial-locations across each ray. Then, for each location, we sample $k = 128$ radiance-density pairs $f^k_{t_s:t_f}; {}^k_{t_s:t_f}$ from the distributions defined by the ouput parameters. Finally, to compute the volume rendering formula in Eq. (3), we approximate its integral using the trapezoidal rule (detailed in Appendix A.4). In our preliminary experiments, this integration method showed better stability than the original alpha-compositing used in standard NeRF.

4.2. Uncertainty estimation in novel-view synthesis

Quantitative results of S-NeRF and the other evaluated methods can be found in Table 1. As we can observe, our method outperforms all the previous approaches across all the scenes and metrics. In particular, S-NeRF improves over the previous state-of-the-art with an average decrease of 35% for NLL and with more than 10% increased MSE-Uncertainty correlation. The better results obtained by our

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

Figure 4.Qualitative results obtained by the evaluated methodsRendered view, Mean-Squared-Error with the ground-truth image, and estimated uncertainty are shown respectively from the rst to the third row. Compared to other approaches, S-NeRF produces uncertainty estimates that are more correlated with the predictive error (R=0.63). Additionally, note that the right part of the image corresponds to a region not covered by the training views. As a consequence, all the methods obtain a high estimation error in the pixels belonging to this area. S-NeRF, however, is the only method that is able to correctly assign high uncertainty values to these pixels.

	Neg. Log. Likelihood(NLL) #			MSE-Uncertainty Correlation "		
	w/o KL	w/ KL	S-NeRF	w/o KL	w/ KL	S-NeRF
Flower	1.41	2.18	1.27	0.54	0.51	0.63
Frotress	1.26	0.82	-0.03	0.31	0.53	0.55
Leaves	0.98	0.96	0.68	0.60	0.65	0.73
Horns	3.08	0.75	0.60	0.53	0.64	0.70
Trex	1.82	1.54	1.37	0.58	0.66	0.68
Fern	2.46	1.63	2.01	0.51	0.66	0.69
Orchids	4.45	2.37	1.95	0.31	0.59	0.65
Room	4.39	3.49	2.35	0.54	0.65	0.74
Avg.	2.48	1.72	1.28	0.49	0.61	0.67

Table 3.The effect of the prior constraints. Conducting KL divergence constraints by a prior with high xed variance (w/KL) dramatically improves the model's performance on NLL and MSE-Uncertainty correlation. An additional prior with optimizable variance over the training set further enhances the model's performance as well as stability.

approach can be explained by the following reasons. Firstly, the quality of the uncertainty estimates provided by Deep-Ensembles and MC-Dropout typically increases when more models in the ensemble or dropout samples are used, respectively. In our experiments, we have limited this number to 5 which can partially explain the worse results of MC-Dropout and Deep-Ensembles compared to S-NeRF. While increasing the number of model evaluations in these approaches could improve their performance, this strategy is not practical for NeRF given that the rendering time also grows dramatically. This can be seen in Table 2, where we report the rendering time for a single scene required by the different methods. Note that MC-Dropout and Deep-Ensembles increase the computational complexity of NeRF by a factor of5. Whereas NeRF-W has a similar computational complexity to S-NeRF, our method also obtains signi cantly improved results in all metrics. As discussed in Sec. 2, NeRF-W treats the variance for each pixel as an additional value rendered in the same manner as pixel RGB intensities. This strategy is not theoretically founded and can lead to sub-optimal results. In contrast, our S-NeRF obtains the uncertainty estimates by sampling multiple color values from the posterior distribution over the radiance elds modeling the scene. As we have shown empirically, this strategy produces more accurate uncertainty estimates without increasing the rendering time.

Qualitative results. To give more insights on the previous experiments, Figure 4 shows an example of qualitative results produced by the evaluated models on a testing view. It is important to notice that the right part of the rendered image corresponds to a region that is not covered by the training views used to learn the model. Therefore, we expect to obtain high error and uncertainty estimates in the pixels belonging to this area. As can be observed, the uncertainty values estimated by MC-dropout, Deep-Ensembles and NeRF-W are poorly correlated with their predictive error. As expected, the MSE is high in the right image region that was not observed in the training set views. However, the corresponding uncertainty values provided by these methods are low. In contrast, S-NeRF is able to assign a high uncertainty to the pixels belonging to the scene region that was not covered by the training views. The ability of our model to identify these regions can be explained by the the high-variance de ning our prior distributions. Concretely, note that the minimized KL divergence in

Eq. (11) forces the learned posterior distribution to resemble this prior when no spatial-locations are observed. As a consequence, the rendered parts of the scene which were not covered by the training views will have an associated high uncertainty. In the following, we conduct an ablation study in order to analyse effect of the prior distributions.

4.3. Analysing the effect of the prior distribution

The prior distribution de ned by S-NeRF allows to identify regions in the scene that are not observed in the training views. Additionally, we also impose a prior with learned variance for the spatial-locations belonging to rays crossing the scene from the observed pixels in the training set (see Sec. 3.2.3). To validate the effectiveness of this approach, we have evaluated three different variants of S-NeRF trained by using different strategies: (i) minimizing only the negative log-likelihood term de ned in Sec. 3.2.2 and ignoring the KL term, (ii) considering also the KL divergence in Eq. (11) between the prior and posterior distribution, (iii) using the proposed optimization objective where we also minimize the additional prior over the "observed" spatial locations.

According to the reported results in Table 3, compared to the case where only the log-likelihood is optimized (w/o KL), minimizing the KL divergence using a high-variance prior signi cantly improves the performance on both NLL and MSE-Uncertainty correlation (w/KL). As previously discussed, this allows S-NeRF to identify the scene regions which are not observed in the training views and assign a high-uncertainty to the corresponding pixels in rendered images. However, we can also observe a signi cant improvement for our proposed S-NeRF optimization process, where we impose a learned prior for the observed spatial-locations. The reason is that de ning a high-variance prior in these areas can lead to sub-optimal results, given that the KL term hinders the model to minimize the negative log-likelihood. In contrast, this is effectively addressed by applying our proposed prior with learned parameters in observed spatial-locations.

4.4. Uncertainty estimation in depth-map synthesis

One of the main advantages of S-NeRF compared to the evaluated methods is that it is also able to quantify the uncertainty associated with the 3D geometry of the scene. In order to illustrate this, Figure 5 shows estimated depth-maps and their associated uncertainty generated for different scenes. By looking at the gure, we can see that our framework can also provide useful information about the model's con dence on the underlying 3D geometry of the scene. For instance, we can observe high uncertainty at the border of foreground objects. This is because this borders correspond to discontinuous changes in the depth-map which produces highly-uncertain estimations. Additionally, we can also observe in the bottom example how S-NeRF is able to assign low con dence values to the depth associated with pixels corresponding to areas of the scene that were not observed in the training set.

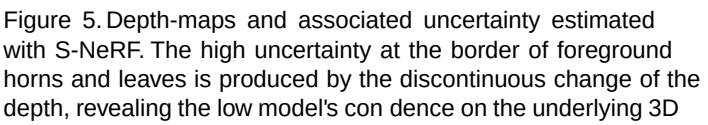


Figure 5. Depth-maps and associated uncertainty estimated with S-NeRF. The high uncertainty at the border of foreground horns and leaves is produced by the discontinuous change of the depth, revealing the low model's con dence on the underlying 3D geometry in those regions.

5. Conclusions

We have presented Stochastic-NeRF, a novel framework to address the problem of uncertainty estimation in neural volume rendering. The proposed approach is a probabilistic generalization of the original NeRF, which is able to produce uncertainty estimates by modelling a distribution over all the possible radiance elds modelling the scene. Compared to state-of-the-art approaches that can be applied for this problem, we have shown that the proposed method achieves signi cantly better results without increasing the computational complexity. Additionally, we have also illustrated the ability of S-NeRF to provide uncertainty estimates for different tasks such as depth-map estimation. To conclude, it is also worth mentioning that our formulation is generic and can be combined with any existing or future method based on the NeRF framework in order to incorporate uncertainty estimation in neural 3D representations.

Acknowledgements

This work is supported partly by the Chinese Scholarship Council (CSC) under grant (201906120031), by the Spanish government under project MoHuCo PID2020-120049RB-I00, the ERA-Net Chistera project IPALM PCI2019-103386 and María de Maeztu Seal of Excellence MDM-2016-0656. Adria Ruiz acknowledges nancial support from MICINN through the program Juan de la Cierva.

References

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. A review of uncertainty quanti cation in deep learning: Techniques, applications and challenges. Information Fusion, 76:243–297, 2021. 2

[2] J. Aitchison and S. M. Shen. Logistic-normal distributions:Some properties and uses. Biometrika, 67(2):261–272, 08 1980. 4

[3] Rohith Aralikatti, D. Margam, Tanay Sharma, Abhinav Thanda, and S. Venkatesan. Global snr estimation of speech signals using entropy and uncertainty estimates from dropout networks. In INTERSPEECH, 2018. 2

[4] J. M. Bernardo and A. F. Smith. Bayesian Theory, volume 405. John Wiley & Sons, 2009. 2

[5] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. Journal of the American statistical Association, 2017. 2, 3

[6] A. Blum, N. Haghtalab, and A. Procaccia. Variational dropout and the local reparameterization trick. Adv. Neural Inform. Process. Syst., 2015. 2

[7] Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. In Adv. Neural Inform. Process. Syst., 2020. 2

[8] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4D facial avatar reconstruction. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[9] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In Int. Conf. Machine Learning, 2016. 2, 6

[10] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In Int. Conf. Machine Learning, pages 1321–1330, 2017. 2

[11] Markus Harva and Ata Kabán. Variational learning for rectified factor analysis. Signal Processing, 2007. 4

[12] S. Hernández, Diego Vergara, Matias Valdenegro-Toro, and Felipe Jorquera. Improving predictive uncertainty estimation using dropout–hamiltonian monte carlo. Soft Computing, 24:4307–4322, 2020. 2

[13] José Miguel Hernández-Lobato and Ryan P. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In Int. Conf. Machine Learning, 2015. 2

[14] Siddhartha Jain, Ge Liu, Jonas Mueller, and David Gifford. Maximizing overall diversity for improved uncertainty estimates in deep ensembles. AAAI, 2020. 2

[15] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Int. Conf. Learn. Represent., 2014. 5, 11

[16] A. Kohli, V. Sitzmann, and G. Wetzstein. Semantic implicit neural scene representations with semi-supervised training. In 3DV, 2020. 1

[17] Adam R Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Sona Mokra, and Danilo Jimenez Rezende. NeRF-VAE: A geometry aware 3d scene generative model. PMLR, 2021. 2

[18] Balaji Lakshminarayanan, A. Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. Adv. Neural Inform. Process. Syst., 2017. 2, 6

[19] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[20] D. B.* Lindell, J. N. P.* Martel, and G. Wetzstein. AutoInt: Automatic integration for fast neural volume rendering. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[21] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. Adv. Neural Inform. Process. Syst., 2020. 2

[22] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. ACM Trans. Graph., 38(4):65:1–65:14, July 2019. 1, 2

[23] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason M. Saragih. Mixture of volumetric primitives for efficient neural rendering. ACM Trans. Graph., 40:1 – 13, 2021. 2

[24] Wesley Maddox, T. Garipov, Pavel Izmailov, D. Vetrov, and A. G. Wilson. A simple baseline for bayesian uncertainty in deep learning. In Adv. Neural Inform. Process. Syst., 2019. 2

[25] 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 IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2, 6

[26] Nelson Max. Optical models for direct volume rendering. IEEE Trans. Vis. Comput. Graph., 1995. 2

[27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy Networks: Learning 3d reconstruction in function space. In IEEE Conf. Comput. Vis. Pattern Recog., 2019. 1

[28] 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 Eur. Conf. Comput. Vis., 2020. 1, 2, 3, 5, 6, 12

[29] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. Comput. Graph. Forum, 40(4), 2021. 2

[30] Michael Niemeyer and Andreas Geiger. GIRAFFE: Representing scenes as compositional generative neural feature fields. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[31] Jeong Joon Park, P. Florence, J. Straub, Richard A. Newcombe, and S. Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In IEEE Conf. Comput. Vis. Pattern Recog., 2019. 1

[32] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural Body:

Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[33] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[34] Amit Raj, Michael Zollhoefer, Tomas Simon, Jason Saragih, Shunsuke Saito, James Hays, and Stephen Lombardi. PVA: Pixel-aligned volumetric avatars. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[35] Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In Adv. Neural Inform. Process. Syst., 2019. 2

[36] Dustin Tran, Michael W. Dusenberry, Mark van der Wilk, and Danijar Hafner. Bayesian layers: A module for neural network uncertainty. In Adv. Neural Inform. Process. Syst., 2019. 2

[37] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

[38] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. iNeRF: Inverting neural radiance fields for pose estimation. In IROS, 2021. 1, 2

[39] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In IEEE Conf. Comput. Vis. Pattern Recog., 2021. 2

Supplemental Materials

A. Methods

In the following, we provide more technical details about our proposed Stochastic Neural Radiance Fields described in Sec. 3.

A.1. Distributions

We present the explicit mathematical expression of the speci c distributions used by S-NeRF to model the radiance and density distributions $q(\mathbf{r}^c|\mathbf{x};\mathbf{d})$ and $q(\sigma|\mathbf{x})$, respectively. Given that the radiance values need to be bounded between 0 and 1, we use a *logistic normal distribution* for each RGB channel $\mathbf{r}^c$ independently. Concretely, its probability density function is de ned as:

$$f(r^c;\mu_{r^c};\sigma_{r^c}) = \frac{1}{r^c}\frac{1}{\sqrt{2\pi\sigma_{r^c}^2}}\frac{1}{r^c(1-r^c)}e^{-\frac{(\text{logit}(r^c)-\mu_{r^c})^2}{2\sigma_{r^c}^2}}; \tag{12}$$

where μ_{r^c} and σ_{r^c} are the mean and std. deviation of the logit form of variable r^c, which is output by the neural network $f_\theta(\mathbf{x};\mathbf{d})$. Similarly, we model the positive density value σ as a random variable following a *recti ed normal distribution*. Its cumulative density function $\Phi()$ and probability density function $f()$ are:

$$\Phi(\sigma;\mu;\sigma) = \frac{1}{2}\int_{-\infty}^{\sigma} e^{-(\mu+t)^2/2\sigma^2}dt \tag{13}$$

$$f(\sigma;\mu;\sigma) = \begin{cases} (0;\mu;\sigma); & \text{if } \sigma \leq 0 \\ \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(\sigma-\mu)^2}{2\sigma^2}} & \text{if } \sigma > 0 \end{cases} \tag{14}$$

where μ and σ are again the mean and std. deviation of the random variable σ output by the S-NeRF network.

A.2. Backpropagation through sampling with the reparametrization-trick

In the following, we provide a detailed explanation on how to properly sample from the learned distributions and back-propagate their gradients. Sampling directly from the learned distribution for density $\sigma \sim q(\sigma|\mathbf{x})$ and radiance $\mathbf{r}^c \sim q(\mathbf{r}^c|\mathbf{x};\mathbf{d})$ is not differentiable, which prevents gradient computation of their parameters $\{\mu_\sigma, \sigma_\sigma, \mu_{r^c}, \sigma_{r^c}, \ldots\}$ during backpropagation. Inspired by [15], we introduce a normally distributed variable to reparameterize the variables density σ and radiance $\mathbf{r}^c$. Concretely, we sample density values as $\sigma = \mu_\sigma + \sigma_\sigma \epsilon$ in Eq. (8) and radiance variables as $\mathbf{r}^c = \text{Sigmoid}(\mu_{r^c} + \sigma_{r^c}\epsilon)$ in Eq. (7), where $\epsilon \sim \mathcal{N}(0;1)$ is a unitary gaussian variable with mean 0 and std. dev. 1. In this manner, the gradients of the distribution parameters can be computed given that this process is fully-differentiable w.r.t them.

A.3. The posterior-prior KL divergence

As we discuss in Sec. 3.2, S-NeRF optimization involves the minimization of the KL term in Eq. (5). This divergence measures the difference between the approximated posterior learned by S-NeRF and a prior distribution over the radiance elds modelling the scene. Given that computing the KL divergence for all the possible location-view pairs in the scene is intractable, we approximate the sum in Eq. (11) by sampling random 3D spatial-locations in the scene as follows. Firstly, we de ne the space bounds at each 3D axis from the captured images of the scene. For instance, we use x_l and x_r to denote the left and right bound of x axis respectively. Then we partition $[x_l, x_r]$ into N evenly-spaced bins and then randomly draw a sample within each bin. After applying this strati ed sampling strategy on each axis, we obtain $N \times N \times N$ points paired with random directions. Secondly, for each sampled location-view pair, we use the network $f_\theta(\mathbf{x};\mathbf{d})$ to compute the posterior distribution parameters $\{\mu_r, \sigma_r, \ldots\}$. Finally, we compute the KL divergence with the prior for the density and radiance variables at each spatial-location. Given that the logistic normal and recti ed normal distributions (Sec. A.1), the KL divergence between the prior and the posterior in both cases has the following explicit expression:

Mathematical derivations for KL $(q(\ |jx)||p(\ ))$ in Eq. (11):

$$
\begin{aligned}
&\sum_{x\in R^3}\sum_{d\in R^2} KL(q(\ |jx)\,\|\,p(\ )) \\
&= E_{q(\ |jx)}\log\left(\frac{q(\ |jx)}{p(\ )}\right) \\
&= \int_{-1}^{1} q(\ |jx)\log\left(\frac{q(\ |jx)}{p(\ )}\right)d\ \\
&= \int_{-1}^{0} q(\ |jx)\log\left(\frac{q(\ |jx)}{p(\ )}\right)d\ + \int_{0}^{1} q(\ |jx)\log\left(\frac{q(\ |jx)}{p(\ )}\right)d\ \\
&= q(\ =0|jx)\log\left(\frac{q(\ =0|jx)}{p(\ =0)}\right) + \int_{0}^{1} q(\ |jx)\log\left(\frac{q(\ |jx)}{p(\ )}\right)d\ \\
&= (0;\ _q;\ _q)[\log((0;\ _q;\ _q)\ \ \log(0;\ _p;\ _p)] + \int_{0}^{1} q(\ |jx)\log\left(\frac{q(\ |jx)}{p(\ )}\right)d\
\end{aligned}
\tag{15}
$$

Where $(0;\ _q;\ _q)$ and $q(\ |jx) = f(\ ;\ ;\ )$ are computed by Eq. (13) and Eq. (14) respectively. For the fth equality, note that the density value is bounded to be positive after a rectified linear unit $(x) = \max(0;\ )$ that sets all the negative values to 0. We have shown the intuitive graphics in Figure 3. In practice, we use a Monte-Carlo estimator over density variable during optimization to approximate the integration the previous equations.

Mathematical derivations for KL $(q(r|jx;d)||p(r))$ in Eq. (11):

$$
\begin{aligned}
\sum_{x\in R^3}\sum_{d\in R^2} KL(q(r|jx;d)\,\|\,p(r)) &= \sum_{c=\{r;g;b\}} E_{q(r^c|jx;d)}\log\left(\frac{q(r^c|jx;d)}{p(r^c)}\right) \\
&= \sum_{c=\{r;g;b\}} \int_{-1}^{1} q(r^c|jx;d)\log\left(\frac{q(r^c|jx;d)}{p(r^c)}\right)dr^c
\end{aligned}
$$

Where $q(r^c|jx;d)$ is equal to $f(r^c;\ ;\ )$ for each radiance channel c, described in Eq. (12). Again, the integration is approximated by a Monte-Carlo estimator over radiance variable.

A.4. Trapezoidal rule

To estimate the continuous integral in Eq. (3), firstly we follow the original NeRF [28] to use a stratified sampling strategy to sample N spatial-locations along each ray. Concretely, we partition $[t_s;t_f]$ into N evenly-spaced bins and then randomly draw a sample within each bin:

$$
t_i \ \sim\ U\left[t_s + \frac{i-1}{N}(t_f - t_s);\ t_s + \frac{i}{N}(t_f - t_s)\right] \tag{16}
$$

For these samples, we can utilize our framework to produce K density-radiance pairs $\{r^{nk}_{t_s:t_f};\ ^{nk}_{t_s:t_f}\}$ for each spatial-location along each ray. As mentioned in Sec. 4.1, we use a different strategy to estimate the continuous integral in Eq. (3) compared to S-NeRF. The motivation is that, in our preliminary experiments, we observed that using an alternative trapezoidal rule [2] to approximate the volume rendering integral is much stable than the traditional alpha compositing used in the original paper [28]. The reason is that the latter can produce extremely large density values when large variances are associated with the sampled distributions. As a consequence, this produces numerical instabilities during optimization. The alternative trapezoidal method used to approximated the aforementioned integral is able to address this limitation and can be expressed as:

$$
\hat{C}^k(x_o;d) = \sum_{i=1}^{N} \frac{1}{2}\ T^k_{i\ 1} r^k_{i\ 1\ \ i\ 1} + T^k_i r^k_{i\ \ i}\ _i; \quad \text{where } T^k_i = \exp\left(-\sum_{j=1}^{i-1}\frac{1}{2}(\ ^k_{j\ 1} + \ ^k_j)\ _j\right) \tag{17}
$$

where $_i = t_i\ \ t_{i\ 1}$ is the distance between adjacent sampled spatial-locations along the ray, $(r^k_i; ^k_i)$ is the kth sample of the K density-radiance pairs at the ith ray location.

[2] Rahman, Qazi I.; Schmeisser, Gerhard (December 1990), "Characterization of the speed of convergence of the trapezoidal rule", Numerische Mathematik, 57 (1): 123–138.

B. Pseudo-algorithm

In Algorithm 1, we provide the pseudocode for the learning process of S-NeRF.

Algorithm 1 Pseudo-algorithm for learning procedure

1: Given N training triplets $\{c^N; x_o^N; d^N\}$, trace a ray from each camera origin x_o^N along direction d^N and sample n points as input pairs $\{x_t^n; d^n\}$
2: For each input pair, use $(x_t^n; d^n)$ to output approximated posterior $q(r|x; d)$ in Eq. (7) and $q(\ |x)$ in Eq. (8)
3: Sample K radiance-density pairs $\{r^n_{t_s:t_f}; \ ^n_{t_s:t_f}\}$ from $q(r|x; d)$ and $q(\ |x)$ for each input pair, and then get K radiance-density trajectories for each ray
4: Integrate each trajectory by volume rendering in Eq. (3) to get K samples of the pixel color c_k^N
5: Compute the log-likelihood with c^N and c_k^N by Eq. (9) and Eq. (10)
6: Sample from all the scene coordinates uniformly and compute the KL term by Eq. (11)
7: Back-propagate and optimize and distribution parameters $\{_{r^c}; \ _{r^c}; \ ; \ \}$ by SGD

C. Evaluation on Image Quality Metrics

The goal of the evaluation conducted in the main paper is to measure the reliability of uncertainty estimations. However, it is also interesting to analyse the results of the different compared models in terms of the rendered images quality. For this reason, in Table 4 we provide results in terms of the SSIM and LPIPS metrics. We highlight the best and the second-best method in the Table. It can be observed that S-NeRF achieves comparable or better average performance than the rest of the baselines. Additionally, note all the individual models of D. Ens are trained using exactly the same loss as standard NeRF and, therefore, it is expected to obtain the same or better results than the latter.

	SSIM "				LPIPS #			
	MC-DO	D. Ens.	NeRF-W	S-NeRF	MC-DO	D. Ens.	NeRF-W	S-NeRF
Flower	0.74	0.72	0.82	[0.85]	0.23	0.16	0.11	[0.10]
Frotress	0.78	[0.92]	0.90	[0.92]	0.40	[0.05]	0.10	0.08
Leaves	0.67	0.75	[0.79]	[0.79]	0.32	[0.17]	0.18	[0.17]
Horns	0.81	[0.89]	0.84	0.88	0.27	0.14	0.21	[0.12]
Trex	0.81	[0.89]	0.74	0.86	0.22	[0.09]	0.33	0.13
Fern	0.78	[0.83]	0.80	[0.83]	0.35	[0.19]	0.31	0.26
Orchids	0.57	0.65	[0.70]	0.68	0.31	0.19	[0.16]	0.18
Room	0.79	0.79	0.83	[0.85]	0.32	0.27	0.28	[0.21]
Avg.	0.74	0.81	0.80	[0.83]	0.30	[0.16]	0.21	[0.16]

Table 4. Quality evaluation of rendered images.

It is also worth mentioning that the NLL metric used in the main paper jointly evaluates the reliability of uncertainty estimates and the image quality. The reason is that in Eq. (18), the numerator of the second term is proportional to the PSNR metric usually employed as an image quality metric. However, in NLL, the PSNR for each pixel is weighted by the predicted variance (i.e. uncertainty). Therefore, when the variance is small, the PSNR term becomes more important. On the other hand, for large variances, the PSNR value is considered less significant since the model identifies that the uncertainty associated with the prediction is high and we can expect a large error.

$$\log p(y|x) = \frac{\log \ ^2(x)}{2} + \frac{(y \quad (x))^2}{2 \ ^2(x)} \tag{18}$$

D. Additional Qualitative Results

In Figure 6, we show additional qualitative obtained by our S-NeRF results across different scenes in the evaluated dataset. For each scene, we show not only the quantified uncertainty (third column) associated with the rendered novel view (second column), but also the estimated uncertainty (fifth column) associated with the generated depth-map (fourth column).

Figure 6.Additional qualitative results obtained by our S-NeRF.