# Reconstructive Latent-Space Neural Radiance Fields for Efficient 3D Scene Representations

Tristan Aumentado-Armstrong[1,2,4*], Ashkan Mirzaei[1,2*], Marcus A. Brubaker[1,3,4], Jonathan Kelly[2,4], Alex Levinshtein[1], Konstantinos G. Derpanis[1,3,4], Igor Gilitschenski[2,4]

*Abstract*— **Neural Radiance Fields (NeRFs) have proven to be powerful 3D representations, capable of high quality novel view synthesis of complex scenes. While NeRFs have been applied to graphics, vision, and robotics, problems with slow rendering speed and characteristic visual artifacts prevent adoption in many use cases. In this work, we investigate combining an autoencoder (AE) with a NeRF, in which latent features (instead of colours) are rendered and then convolutionally decoded. The resulting latent-space NeRF can produce novel views with higher quality than standard colour-space NeRFs, as the AE can correct certain visual artifacts, while rendering over three times faster. Our work is orthogonal to other techniques for improving NeRF efficiency. Further, we can control the tradeoff between efficiency and image quality by shrinking the AE architecture, achieving over 13 times faster rendering with only a small drop in performance. We hope that our approach can form the basis of an efficient, yet high-fidelity, 3D scene representation for downstream tasks, especially when retaining differentiability is useful, as in many robotics scenarios requiring continual learning.**

## I. INTRODUCTION

Neural rendering techniques [1] continue to grow in importance, particularly Neural Radiance Fields [2] (NeRFs), which achieve state-of-the-art performance in novel view synthesis and 3D-from-2D reconstruction. As a result, NeRFs have been utilized for a variety of applications, not only in content creation [3], [4], [5], [6], but also for many robotics tasks, including 6-DoF tracking [7], pose estimation [8], surface recognition [9] or reconstruction [10], motion planning [11], [12], [13], reinforcement learning [14], [15], tactile sensing [16], and data-driven simulation [17], [18]. However, slow rendering and the qualitative artifacts of NeRFs impede further use cases in production.

To render a single pixel, one major bottleneck is the need for multiple forward passes of a multilayer perceptron (MLP). Replacing or augmenting the MLP with alternative representations (e.g., voxel grids [19] or feature hashtables [20]) has been used to improve both training and inference speed. Baking NeRFs into other primitive representations has also been a popular approach [21], [22], [23] for faster rendering. To reduce artifacts (e.g., "floaters" [24]), different sampling methods [25], [26], [27], [28], radiance

[1]Samsung AI Centre Toronto  [2]University of Toronto  [3]York University  [4]Vector Institute for AI. *Equal Contribution. Emails: {a.mirzaei, tristan.a}@partner.samsung.com, {jkelly, gilitschenski}@cs.toronto.edu, {kosta, mab}@eecs.yorku.ca, alex.lev@samsung.com
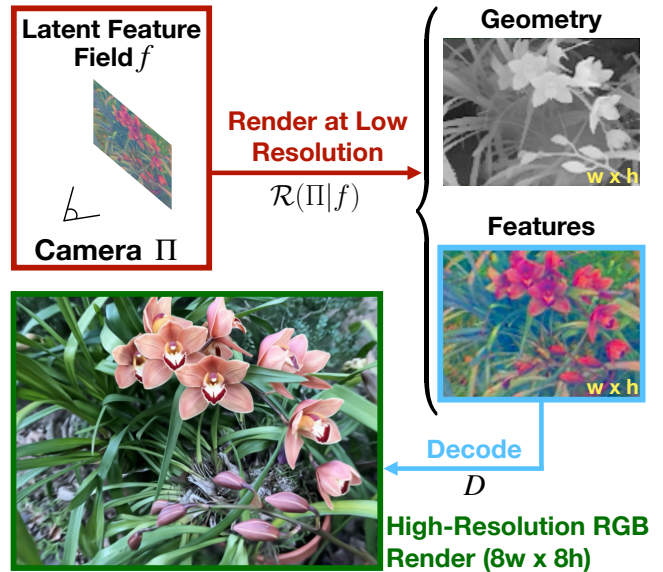
Fig. 1.    Illustration of the Reconstructive Latent-Space NeRF (ReLS-NeRF) rendering pipeline (see §III). As shown in the upper-left inset, given a trained feature field, $f$, and a camera, $\Pi$, we can render a latent feature map at a *low resolution*, as shown in the right inset. The geometry of the scene, encapsulated by the density field, which determines the 3D structure of the feature render, is learned via an RGB component (as in a regular NeRF). A decoder, $D$, can then map the low-resolution feature maps to a high-resolution colour image (lower-left inset). We may view this process, which maps camera parameters to images, as a form of neural rendering.

models [29], and scene contraction functions [30], [28] have been proposed. Despite these advancements, NeRFs still suffer from visual flaws and low rendering frame-rates.

Importantly, such issues hamper the use of NeRFs for downstream tasks, If rendering is too slow, agents will be unable to apply NeRFs as an internal 3D representation of the scene. Further, the solutions considered (often aimed at applications in computer graphics, for instance) may not be compatible with the requirements of other tasks. For example, meshification [22], [31] enables fast rendering, but makes further online learning of the geometry significantly more difficult, due to topological constraints [32] and additional optimization complexity (e.g., to handle self-intersections and other unnatural geometries) [33], [34]. We also do not wish to sacrifice too much representational fidelity (e.g., not including view-dependent effects [35]) for speed, as less accurate visual output can limit downstream opportunities for

scene analysis. We therefore require a model that is capable of fast rendering and intra-task optimization (i.e., learning during an ongoing task), without sacrificing visual quality.

In this paper, we propose an approach for solving these challenges that is orthogonal to existing methods. By leveraging convolutional autoencoders (AEs), we can define a "NeRF" operating in latent feature space (rather than colour space), such that *low*-resolution *latent* renders can be decoded to *high*-resolution RGB renders (see Fig. 1). This offloads expensive MLP-based rendering computations to the low-cost AE, greatly improving efficiency. Thus, we extend the standard NeRF architecture to return point-wise latent vectors, in addition to densities and colours (the latter used only in training). Since the decoder is simply another differentiable neural network, the ability to optimize the underlying 3D NeRF field is largely unchanged. As it is used for scene reconstruction, we denote the resulting combined field a *Reconstructive Latent-Space NeRF* (ReLS-NeRF). Beyond improving rendering speed, the AE can also act as an image prior, fixing some of the artifacts associated with direct NeRF renders, and actually *improving* representational fidelity. However, we also observe that the use of the AE in ReLS-NeRF can introduce unique *temporal* artifacts, which existing image and video do not capture; hence, we define a novel metric that takes advantage of the geometric structure of the NeRF to detect them.

Overall, by fine-tuning a powerful pretrained AE, our model is able to render views several times faster, while empirically improving in multiple image and video quality metrics. Further, we demonstrate a tradeoff between visual quality and rendering efficiency: by reducing the AE size, we obtain a 13-fold speed-up, with only a minor drop in quality. In summary, we contribute (i) a novel approach to reconstructive 3D scene representation, via a latent-space NeRF that both improves rendering efficiency and outperforms existing work on standard image and video quality metrics; (ii) a new evaluation metric, designed to detect temporal artifacts due to view inconsistencies, which existing metrics do not appear to capture; and (iii) the ability to trade-off image quality and rendering speed via varying the AE architecture.

## II. RELATED WORK

### A. Improving NeRF efficiency

While NeRFs produce results of extraordinary quality, the speed of fitting (training) and rendering (inference) remains a bottleneck for adoption in a variety of applications (e.g., [28], [17], [36]). This has prompted a myriad of approaches to increasing their efficiency. Feature grid architectures have proven effective in expediting fitting convergence (e.g., [37], [38], [39], [26], [40], [41], [19], [20]). Other approaches include utilizing depth [42], better initializations [43], and pretraining conditional fields (e.g., [44], [45], [46]). Such improvements can be readily utilized in our own framework. Similarly, a number of methods have been proposed to enhance the efficiency of the volume rendering operation, which relies on an expensive Monte Carlo integration involving many independent neural network calls per pixel.

These include architectural modifications [47], [48], [49], [50], [51], spatial acceleration structures [52], "baking" (precomputing and storing network outputs) [21], [23], improved sampling strategies [53], [54], [55], [56], [57], or altering the integration method itself [58], [59]. Finally, several works eschew volume rendering itself. A number of representations [60], [61], [62], [63], [64], [65] use only a single sample per pixel, but struggle with geometric consistency and scalability. Similarly, one can move to a mesh-based representation and use rasterization instead [22], [66], [31]; however, this loses certain properties, such as amenability to further optimization or differentiable neural editing. Though our approach improves rendering efficiency, it is orthogonal to these methods, as it reduces the number of MLP calls per image by changing the output space of the NeRF itself.

### B. Feature-space NeRFs

Other models have utilized *neural feature fields* (NFFs), as opposed to "radiance" fields, where rendering is altered to output learned features instead. Some NFFs [67], [68] learn to produce the outputs of pretrained 2D feature extractors; similarly, several works have considered the use of language-related features [69], [70], [71] and other segmentation signals [72], [73], [74], [5] to embed semantics into the NFF. More closely related to our work are generative modelling NFFs that decode rendered features into images via generative adversarial networks [75], [76], [77] or diffusion models [78], [79], [80]. In contrast, this paper considers the scene reconstruction problem, using a latent representation potentially amenable to downstream tasks, and investigates issues related to view consistency. In particular, the artifacts of generative methods are similar to those detected by our novel quality metric (namely, appearance inconsistencies across close frames or camera viewpoints; e.g., see [76]).

## III. METHODS

As in the standard NeRF scenario, we expect only a set of multiview posed images, $S_I = \{(I_i, \Pi_i)\}_i$. The goal is to learn a 3D scene representation in an autoencoder (AE) latent space, capable of novel view synthesis. Thus, our model includes two neural modules (§III-A): (i) a modified NeRF, $f$, which outputs a latent vector (in addition to its standard outputs), and (ii) an AE, with encoder and decoder networks, $E$ and $D$. To fit the model, we apply a multi-stage process: training the AE, fitting the NeRF, and then fine-tuning $D$ (see §III-B).

### A. ReLS-NeRF Neural Architecture

We first extend the standard colour-density field of NeRF to include a latent feature vector, $z$, via $f(x, r) = (\sigma \in \mathbb{R}_+, c \in [0,1]^3, z \in \mathbb{R}^n)$, where $x$ and $r$ represent the input position and direction, and $\sigma$ and $c$ represent the output density and colour. We refer to the $\sigma$ and $c$ fields as an "RGB-NeRF", to distinguish them from the latent component of the ReLS-NeRF. Note that the RGB-NeRF is used only in training, to learn the density field and produce renders to help
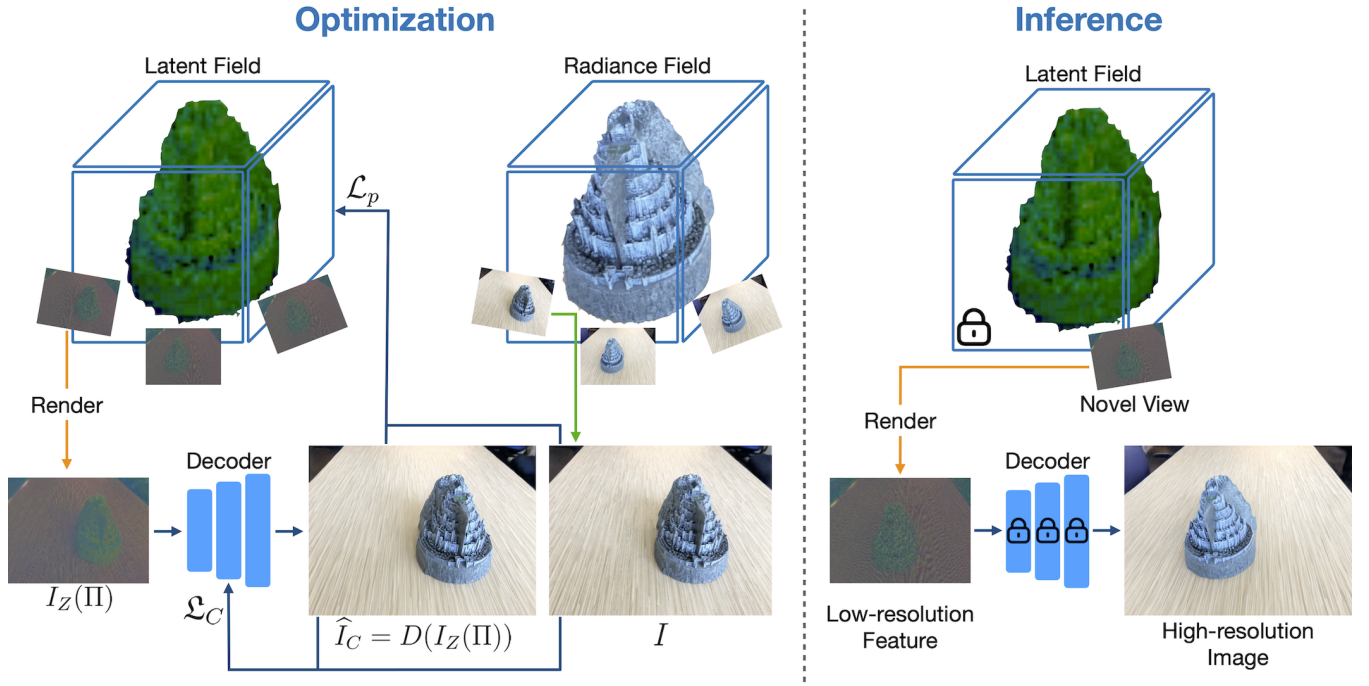
Fig. 2. An overview of the ReLS-NeRF fitting and inference processes. *Left: optimization approach*. The radiance (colour) field is fit to RGB captures, as in the standard NeRF [2]. Given camera parameters, $\Pi$, ReLS-NeRF renders feature maps in the latent $Z$-space defined by a convolutional autoencoder (AE), $D \circ E$, for which arbitrary views can be decoded into image space. The discrepancy between the decoded features and the corresponding images (from an RGB-NeRF or real images) enables training the $Z$-space NeRF and the AE. *Right: inference approach*. After freezing the latent field and decoder, one can render the scene from an arbitrary viewpoint, obtaining a latent feature map that can be decoded into a high-resolution image.

train the latent component (see §III-B). Volume rendering is unchanged: for a single feature at a pixel position, $p$, we use

$$Z(p) = \int_{t_{\min}}^{t_{\max}} \mathcal{T}(t)\sigma(t)z(t)\,dt, \qquad (1)$$

to obtain the feature value at $p$, where $\mathcal{T}(t)$ is the transmittance [81], and $z(t) = z(x(t), r(t))$ is obtained by sampling the ray defined by $p$. For camera parameters $\Pi$, we denote the latent image rendering function as $\mathcal{R}(\Pi|f) = I_Z(\Pi)$, where $I_Z[p] = Z(p)$. Replacing $z(t)$ with $c(t)$, for instance, would render colour in the standard manner, giving a colour image, $I_C(\Pi)$ (that does *not* use $z$). To obtain a colour image from $I_Z$, we simply pass it to the decoder, $D$; i.e., view synthesis is $\widehat{I}_C(\Pi) = D(I_Z(\Pi))$, which can be viewed as a form of *neural rendering* (e.g., [76], [82], [83]). The benefit of using $\widehat{I}_C$ is that significantly fewer pixels need to be rendered, compared to $I_C(\Pi)$; it also enables placing a prior on $\widehat{I}_C$ by choosing $D$ appropriately.

We considered two choices of AE: (i) the *pretrained* VAE from Stable Diffusion [84], which we denote SD-VAE, and (ii) a smaller residual block-based AE [85], [86] (R32, when using a 32D latent space) that is randomly initialized. Both encoders provide an $8\times$ downsampling of the image.

*B. Fitting Process*

A ReLS-NeRF is optimized in three stages: (A) AE training, (B) joint NeRF fitting, and (C) decoder fine-tuning. **AE training (A).** The first phase simply trains (or fine-tunes) the AE to reconstruct the training images of the scenes, using the mean-squared error.

**Joint NeRF fitting (B).** In the second phase, we train the RGB and Latent components of the NeRF in conjunction with the decoder, $D$. Our total loss function,

$$\mathfrak{L}_B = \mathcal{L}_r + \lambda_d\mathcal{L}_d + \lambda_{\mathrm{gr}}\mathcal{L}_{\mathrm{gr}} + \mathcal{L}_p, \qquad (2)$$

consists of the standard RGB loss on random rays, $\mathcal{L}_r$, the DS-NeRF [42] depth loss, $\mathcal{L}_d$, the geometry regularizing distortion loss [28], $\mathcal{L}_{\mathrm{gr}}$, and a patch-based loss for training the latent component, $\mathcal{L}_p$. Given a posed image, $(I, \Pi)$, the latter loss is simply the error between a sample patch, $\mathcal{P} \sim I$, and the corresponding rendered then decoded patch,

$$\mathcal{L}_p = \mathbb{E}_{\mathcal{P} \sim I, (I,\Pi) \sim S_I}\mathrm{MSE}(\mathcal{P}, D(I_Z(\Pi))). \qquad (3)$$

**Decoder fine-tuning (C).** Finally, we fine-tune $D$, utilizing a combination of the multiview posed images, $S_I$, and renders from the RGB component of the ReLS-NeRF. First, we sample random renders, $\widetilde{S}_I = \{(I_C(\Pi_s), \Pi_s) \,|\, \Pi_s \sim \Gamma(S_\Pi)\}_s$, where $\Gamma(S_\Pi)$ is the uniform distribution over camera extrinsics, obtained by interpolating between any triplet in $S_\Pi$. Optimizing

$$\mathfrak{L}_C = \gamma\delta(S_I) + (1 - \gamma)\delta(\widetilde{S}_I), \qquad (4)$$

where $\delta(S) = \mathbb{E}_{(I,\Pi) \sim S}\mathrm{MSE}(I, \widehat{I}_C(\Pi))$ and $\gamma \in [0, 1]$ is a weighting hyper-parameter, distills information from the RGB-NeRF into latent renderer. See Fig. 2. Note that the real training images, $S_I$, are used; hence, the RGB-NeRF is *not* strictly a ceiling on performance (further, the presence of $D$ implies different generalization properties).

3

## C. Implementation Details

We utilize the neural graphics primitives [20] architecture, via the `tiny-cuda-nn` library [87]. All phases use Adam [88] for optimization. We remark that the loss gradient from the latent component of the NeRF (i.e., from $\mathcal{L}_p$) is not back-propagated to the colour, $c$, and density, $\sigma$, fields. Further, we use separate features for the latent feature vector, $z$, and $c$, but render with the same $\sigma$. In other words, RGB-NeRF training is unaffected by $z$. For additional details, we refer the reader to our appendix.

## IV. EXPERIMENTS

### A. Evaluation Metrics

*1) Pixelwise and perceptual distances:* We measure performance with novel view synthesis on held-out test views. In addition to the standard pixelwise peak signal-to-noise ratio (PSNR), we use perceptual losses to measure similarity, including LPIPS [89] and DreamSim [90]. LPIPS provides more human-like responses to low-level distortions (e.g., noise, small colour/spatial shifts), whereas DreamSim is designed to be "mid-level" metric, better capturing large-scale and semantic differences than LPIPS (without being as high-level as, e.g., CLIP-based metrics [91], [92], [93]).

*2) Local consistency:* When examining generative models of NeRFs that use decoders, we can qualitatively see a "shimmering" effect in time (e.g., [76], [75]), which is also reminiscent of generative video model artifacts (e.g., [94], [95]). This jittering appears related to local appearance inconsistencies: since each latent pixel corresponds to an RGB *patch*. As $\Pi$ changes, interpolating in $z$-space does not perfectly approximate the correct appearance changes. This behaviour is distinct from the artifacts observed in standard NeRFs and we devise a simple metric to detect it: the *Reprojective Colour Consistency (RCC) metric*. The RCC measures sudden changes in appearance as $\Pi$ changes, relying on the NeRF geometry to obtain correspondences. Specifically, we reproject one render, $I_i$, into the reference frame of another, $I_{i+1}$, using the NeRF depth, $D_i$, so

$$\text{RCC} = \text{PSNR}\left(\mathbb{E}_i[\text{MSE}(I_{i+1}, \text{Reproj}_{D_i, \Pi_{i+1}} I_i)]\right), \quad (5)$$

where $I_i$ and $I_{i+1}$ are adjacent video frames. Notice that occlusions and view-dependent lighting effects will confound the RCC; however, these effects will (i) be relatively minimal across adjacent frames and (ii) be shared for the same scene, enabling it to be a fair comparative metric.

*3) Video quality:* As noted above, adding a temporal dimension can make certain artifacts more perceptually detectable. We therefore applied a recent video quality metric, DOVER [96], to NeRF-rendered videos. DOVER has two components: DOVER-aesthetic (DoA), which focuses on high-level semantics, and DOVER-technical (DoT), which detects low-level distortions (e.g., blur and noise). DOVER and the RCC are applied to 120-frame "spiral video" renders from the NeRF (as in LLFF [97]).

| NeRF | Reference-based | | | Reference-free | | |
|---|---|---|---|---|---|---|
| | PSNR↑ | LPIPS↓ | DS↓ | DoA↑ | DoT↑ | RCC↑ |
| RGB | 23.52 | 0.37 | **1.18** | 80.2 | 72.9 | **25.6** |
| Ours-SD | **23.81** | **0.35** | 1.44 | **81.5** | **77.3** | 25.5 |
| Ours-R32 | 23.37 | 0.40 | 1.71 | 76.4 | 74.3 | 25.3 |

Tab. 1. Test-view evaluation on eight LLFF scenes [97]. Reference-based metrics include PSNR, LPIPS [89], and DreamSim (DS; ×10) [90]. For reference-free metrics, we use DOVER-technical (DoT), DOVER-aesthetic (DoA), and our reprojective colour consistency (RCC) measure, computed on rendered videos. Rows correspond to the standard RGB NeRF, the SDVAE-based ReLS-NeRF, and the R32-based ReLS-NeRF. ReLS-NeRF-SDVAE outperforms the RGB-space NeRF on the lower-level reference-based (PSNR and LPIPS) and reference-free (DoT) metrics, but has mixed performance on the more semantic metrics (DS and DoA). Our RCC metric, designed to detect the "shimmer" present in decoded (neural rendered) videos, detects slightly more inconsistency with ReLS-NeRF. Using R32 reduces accuracy, but enables much faster rendering time (see Table 2).

| NeRF | Rendering Time | Fitting Time | | |
|---|---|---|---|---|
| | | (A) | (B) | (C) |
| RGB | 132.1s [1×] | – | 1h | – |
| Ours-SD | 43.1s [3×] | 10m | 2h | 2.5h |
| Ours-R32 | **10.2s [13×]** | 40m | 1.5h | 1.5h |

Tab. 2. Timings for inference (rendering 120 frames) and fitting. Changing the decoder architecture, $D$, trades off between efficiency and image quality. We measure the RGB-NeRF rendering time without the latent component.

### B. Reconstruction Quality and Timing

We display our evaluation in Table 1, as well as timing measurements in Table 2, using eight LLFF scenes [97] (see also Fig. 3 for qualitative examples)*, at $1008 \times 756$ resolution. We see that ReLS-NeRF (i.e., decoding a rendered latent feature map) with the SDVAE actually has superior novel view image quality, while having superior inference speed (three times faster). In particular, the low-level metrics, including PSNR, LPIPS, and DoT, all prefer ReLS-NeRF-SD over the standard colour NeRF. This is likely due to the fine-tuned decoder fixing artifacts incurred by the colour NeRF, as can be seen in Fig. 3. The higher-level, more semantic metrics are more mixed: DreamSim prefers the RGB-NeRF, while DoA slightly favours ReLS-NeRF-SD.

Among reference-based metrics, the semantically-oriented DreamSim is the only one by which the RGB-NeRF outperforms ReLS-NeRF-SD. Since DreamSim is a single-image metric, it is insensitive to temporal artifacts; however, DreamSim is known to be more sensitive to foreground objects [90]. Interestingly, we qualitatively observe that ReLS-NeRF tends to improve image quality the most in scene areas far from the camera, where geometry is generally poorer quality – i.e., in the background (see Fig. 3). Thus, one might speculate that such improvements are simply going unnoticed for DreamSim, which tends to focus on foreground objects of greater semantic importance.

In addition, we find that the RCC prefers the RGB-NeRF over ReLS-NeRF. Though it is hard to see in still images, ReLS-NeRF has slight temporal "jittering" artifacts, which the RCC is designed to detect. We remark that other algorithms show similar view-inconsistencies across close frames (e.g., 3D generative models [76] or video generators [94]),
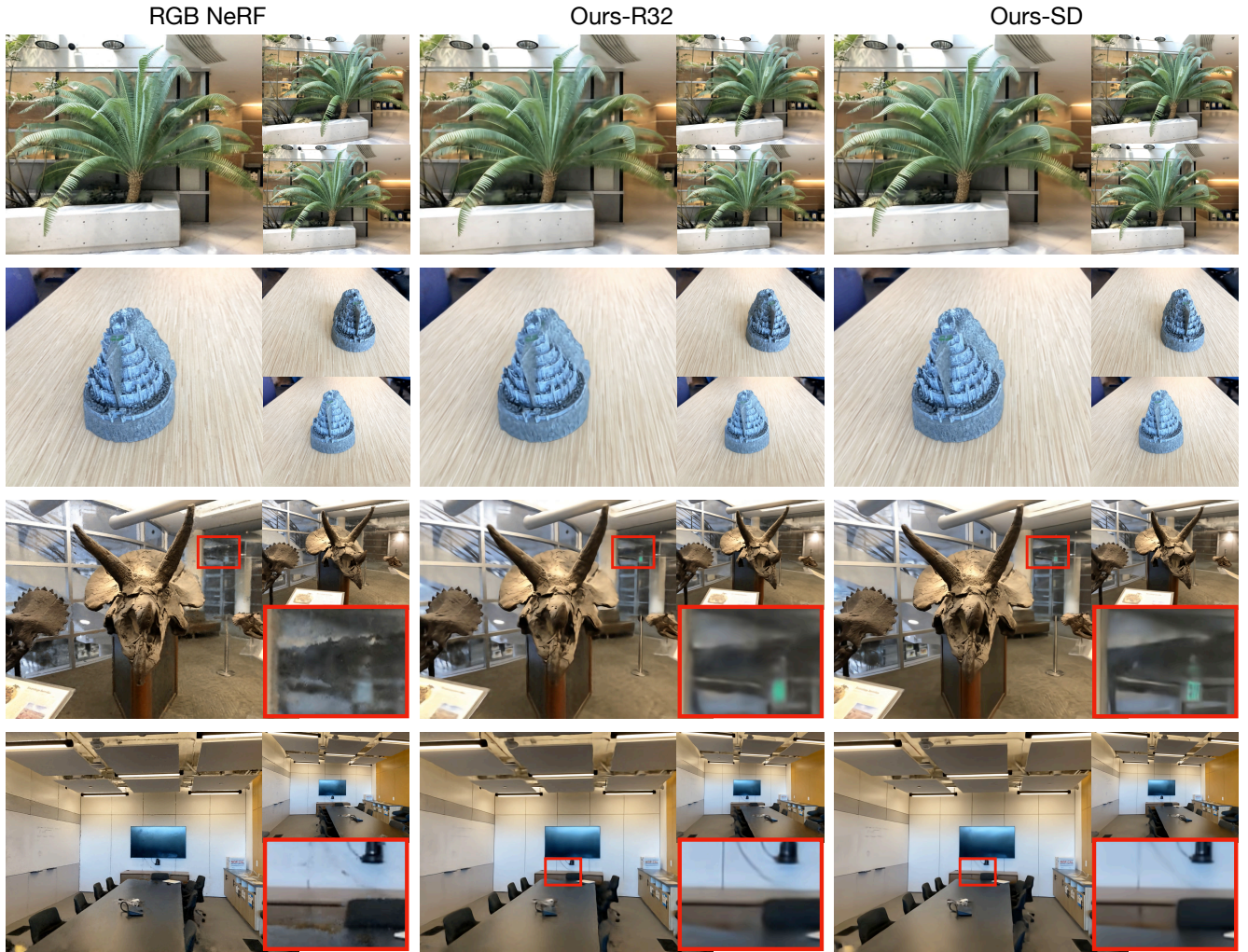
Fig. 3. Qualitative comparison of NeRF renders. In the zoomed insets, we show how ReLS-NeRF-SD fixes some of the artifacts of the RGB-NeRF, despite being trained in part on its renders (see §III-B, phase C) One can also see the slight blur incurred by using the faster R32 AE (middle column). Notice that improvement in visual quality can actually have significant *semantic* effects, in a manner that could impact downstream tasks (e.g., navigation): in the third row, for instance, one can actually read the "exit" sign in ReLS-NeRF-SD, but not in the other two cases.

and could potentially benefit from RCC estimates. We illustrate this phenomenon with some examples in Fig. 4. Due to the learned decoder, unexpected appearance changes can occur across viewpoints. However, per-frame metrics, such as the traditionally applied LPIPS and PSNR, do not capture such inconsistencies; hence, ReLS-NeRF outperforms the RGB-NeRF on them (Table 1). Interestingly, even the video metrics (DoT and DoA) prefer ReLS-NeRF, suggesting such algorithms are more sensitive to the cloudiness and noise artifacts of the standard NeRF, compared to the small jitters incurred by the neural rendering process. In other words, by most metrics of quality (including the primary standard ones, PSNR and LPIPS), ReLS-NeRF is superior.

Finally, we show that the trade-off between rendering efficiency and image quality can be controlled by changing the AE architecture. Using R32 reduces inference time by ∼92%, while decreasing test-view PSNR by only 0.15, compared to the RGB-NeRF rendering process. In contrast to ReLS-NeRF-SD, while ReLS-NeRF-R32 does sacrifice

some image quality (e.g., ∼0.4 PSNR loss), it also reduces inference time by ∼76%. One can imagine choosing an architecture with the right level of trade-off for a given task.

### C. Ablations

We find that removing phase C is devastating to ReLS-NeRF, causing PSNR to drop to 22.85 (SD) and 20.87 (R32). Since the SDVAE is pretrained, ablating phase A has little effect on ReLS-NeRF-SD; however, doing so for ReLS-NeRF-R32 reduces PSNR by 0.1. Note that the latter case trains the decoder, $D$, alongside the NeRF and then alone, in phases B and C.

## V. DISCUSSION

We have shown that ReLS-NeRF can improve image quality, while being several times faster to render. In particular, the SD-based ReLS-NERF is superior on the main metrics commonly used to evaluate NeRFs on test views (i.e., PSNR and LPIPS), as well as on a state-of-the-art reference-free

Fig. 4. Examples of adjacent video frames from ReLS-NeRF-SD (top row) and the RGB NeRF (bottom row). Each pair of images are temporally adjacent renders from a video. Notice, as in Fig. 3, that ReLS-NeRF-SD has better per-frame image quality, as measured in the quantitative results of Table 1. For example, see the upper half of the leftward zoomed insets, where the RGB NeRF is more "cloudy". However, there are *temporal* artifacts that cannot be detected in a single frame (i.e., small cross-view appearance inconsistencies). For instance, as can be seem in the highlighted areas of the zoomed insets, small spots can suddenly appear (left), or the shape of highlights can change (right). This does not occur in the RGB NeRF, as volume rendering colours directly encourages view consistency, whereas the learned decoder in ReLS-NERF can introduce inconsistent appearances. This showcases the utility and need for our new reprojective colour consistency (RCC) metric (see §IV-A.2), which can capture these temporal aspects more directly.

video quality estimator. Empirically, we observed that current image and video evaluation metrics do not obviously capture temporal artifacts that are characteristic of ReLS-NeRF, caused by view-inconsistent appearance changes (due to the learned component within the rendering process). Hence, we introduced a simple metric for detecting such anomalies. Further, we have demonstrated a tradeoff between efficiency and quality, which can be controlled by the architecture of the AE. Importantly, to obtain its speedup, ReLS-NeRF does not "bake" the scene or transform to a mesh; hence, e.g., it could still be continually trained online in the standard fashion. In other words, it retains a number of useful properties of standard NeRFs (e.g., differentiability and access to an implicit 3D shape field), while gaining additional efficiency and image quality.

For many robotics tasks, *fast differentiable rendering* is a key component for online learning of 3D scene representations. This includes simultaneous localization and mapping, navigation, and modelling the dynamics of the environment (i.e., ensuring the internal representation is up-to-date, given perceptual inputs). We feel that ReLS-NeRF is well-suited for such situations, as it retains differentiability, while improving rendering efficiency and even image quality as well. Other promising future directions include utilizing different AEs to provide task-specific biases (e.g., for 3D scene editing, faster speed, or higher image quality), improving the AE architecture to suit this scenario (e.g., devising a geometry-aware decoder), and better customizing the volume rendering process to latent space rendering (e.g., using a learned mapping instead of volume integration).

## APPENDIX

### A. Additional Implementation Details

When training, we used $\lambda_d = 0.1$, $\gamma = 0.7$, and $\lambda_{gr} = 10^{-3}/2$. The NeRF architecture was the same as previous works based on Instant-NGP (see [5]). The LLFF scenes used were fern, horns, orchids, flower, leaves, room_tv, trex, and fortress.

### B. Fitting Hyper-Parameters

**Phase A.** The SDVAE/R32 NeRFs were optimized for 500/3000 iterations, using learning rates of $10^{-4}/4 \times 10^{-4}$. The learning rates were halved at 150, 300, and 450 iterations (SDVAE) and every 500 iterations for R32. Patches of size $512^2$ were used, with batch sizes of 3/5.

**Phase B.** The joint optimization was run for 20K iterations. We used 4096 rays for the colour and DS-NeRF losses, each. The latent loss, $\mathcal{L}_p$, is computed via $32^2$ latent-space patches. The learning rate (excluding the VAE) starts from $10^{-2}$ and is decayed according to $10^{-2} \times (10^{-1})^{t/\tau}$, where $t$ is the step iteration and $\tau = 10^4$. The VAE is optimized with a fixed learning rate of $10^{-4}$.

**Phase C.** Decoder fine-tuning proceeds for 3000/10000 iterations for the SDVAE/R32 architecture. A batch size of three was used (one from $S_I$ and two from $\widetilde{S}_I$). Note that we render 512 images from the RGB-NeRF to act as supervision (i.e., $|\widetilde{S}_I| = 512$). The process starts from a learning rate of $10^{-4}$, and is decayed by 0.5 every 1000/2500 iterations.

### C. R32 Architecture

The encoder, $E$, has the following structure: C5, RBIN, HD, RBIN, HD, RBIN, HD, RBIN, C1. The components are as follows: C5 is a conv-5×5-norm-elu block; RBIN is two residual blocks [85], each using conv-3×3 and norm; HD is a bilinear halving downscaler; and C1 is just a conv-1×1. The encoder has layer sizes of (32,128,128,256,256).

The decoder, $D$, has the following structure: C1, RBIN, HU, RBIN, HU, RBIN, HU, RBIN, C1, sigmoid. Components are the same, except that HU is a bilinear doubling upscaler. The decoder has layer sizes of (256,256,128,128,32).

Both networks use the ELU non-linearity [98] and instance normalization [99] as norm.

## REFERENCES

[1] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, Y. Wang, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi, T. Simon, C. Theobalt, M. Niessner, J. T. Barron, G. Wetzstein, M. Zollhoefer, and V. Golyanik, "Advances in neural rendering," in *Proceedings of SIGGRAPH*, 2021. 1

[2] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *European Conference on Computer Vision (ECCV)*, 2020. 1, 3

[3] A. Haque, M. Tancik, A. Efros, A. Holynski, and A. Kanazawa, "Instruct-NeRF2NeRF: Editing 3D scenes with instructions," *arXiv preprint arXiv:2303.12789*, 2023. 1

[4] K. Zhang, N. Kolkin, S. Bi, F. Luan, Z. Xu, E. Shechtman, and N. Snavely, "ARF: Artistic radiance fields," in *European Conference on Computer Vision (ECCV)*, 2022. 1

[5] A. Mirzaei, T. Aumentado-Armstrong, K. G. Derpanis, J. Kelly, M. A. Brubaker, I. Gilitschenski, and A. Levinshtein, "SPIn-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 6

[6] A. Mirzaei, T. Aumentado-Armstrong, M. A. Brubaker, J. Kelly, A. Levinshtein, K. G. Derpanis, and I. Gilitschenski, "Reference-guided controllable inpainting of neural radiance fields," in *International Conference on Computer Vision (ICCV)*, 2023. 1

[7] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Muller, A. Evans, D. Fox, J. Kautz, and S. Birchfield, "BundleSDF: Neural 6-DoF tracking and 3D reconstruction of unknown objects," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[8] M. Z. Irshad, S. Zakharov, R. Ambrus, T. Kollar, Z. Kira, and A. Gaidon, "ShAPO: Implicit representations for multi object shape appearance and pose optimization," *European Conference on Computer Vision (ECCV)*, 2022. 1

[9] R.-Z. Qiu, Y. Sun, J. M. Correia Marques, and K. Hauser, "Real-time semantic 3D reconstruction for high-touch surface recognition for robotic disinfection," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2022. 1

[10] S. Lee, C. Le, W. Jiahao, A. Liniger, S. Kumar, and F. Yu, "Uncertainty guided policy for active robotic 3D reconstruction using neural radiance fields," *IEEE Robotics and Automation Letters*, 2022. 1

[11] R. Ni and A. H. Qureshi, "NTFields: Neural time fields for physics-informed robot motion planning," *International Conference on Learning Representations (ICLR)*, 2023. 1

[12] M. Kurenkov, A. Potapov, A. Savinykh, E. Yudin, E. Kruzhkov, P. Karpyshev, and D. Tsetserukou, "NFOMP: Neural field for optimal motion planner of differential drive robots with nonholonomic constraints," *IEEE Robotics and Automation Letters*, 2022. 1

[13] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, "Vision-Only Robot Navigation in a Neural Radiance World," *IEEE Robotics and Automation Letters*, 2022. 1

[14] D. Driess, I. Schubert, P. Florence, Y. Li, and M. Toussaint, "Reinforcement learning with neural radiance fields," *Neural Information Processing Systems (NeurIPS)*, 2022. 1

[15] D. Shim, S. Lee, and H. J. Kim, "SNeRL: Semantic-aware neural radiance fields for reinforcement learning," *arXiv preprint arXiv:2301.11520*, 2023. 1

[16] S. Zhong, A. Albini, O. P. Jones, P. Maiolino, and I. Posner, "Touching a NeRF: Leveraging neural radiance fields for tactile sensory data generation," in *Conference on Robot Learning*, 2023. 1

[17] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-NeRF: Scalable large scene neural view synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[18] Z. Yang, S. Manivasagam, Y. Chen, J. Wang, R. Hu, and R. Urtasun, "Reconstructing objects in-the-wild for realistic sensor simulation," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023. 1

[19] Sara Fridovich-Keil and Alex Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[20] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (TOG)*, 2022. 1, 2, 4

[21] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," in *International Conference on Computer Vision (ICCV)*, 2021. 1, 2

[22] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi, "MobileNeRF: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2

[23] C. Reiser, R. Szeliski, D. Verbin, P. P. Srinivasan, B. Mildenhall, A. Geiger, J. T. Barron, and P. Hedman, "MERF: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes," *arXiv preprint arXiv:2302.12249*, 2023. 1, 2

[24] F. Warburg, E. Weber, M. Tancik, A. Holynski, and A. Kanazawa, "Nerfbusters: Removing ghostly artifacts from casually captured NeRFs," *arXiv preprint arXiv:2304.10532*, 2023. 1

[25] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, J. Kerr, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, and A. Kanazawa, "NeRFstudio: A modular framework for neural radiance field development," in *Proceedings of SIGGRAPH*, 2023. 1

[26] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Zip-NeRF: Anti-aliased grid-based neural radiance fields," *arXiv preprint arXiv:2304.06706*, 2023. 1, 2

[27] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields," *International Conference on Computer Vision (ICCV)*, 2021. 1

[28] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-NeRF 360: Unbounded anti-aliased neural radiance fields," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3

[29] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-NeRF: Structured view-dependent appearance for neural radiance fields," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[30] K. Zhang, G. Riegler, N. Snavely, and V. Koltun, "NeRF++: Analyzing and improving neural radiance fields," *arXiv preprint arXiv:2010.07492*, 2020. 1

[31] Z. Wan, C. Richardt, A. Božič, C. Li, V. Rengarajan, S. Nam, X. Xiang, T. Li, B. Zhu, R. Ranjan, and J. Liao, "Learning neural duplex radiance fields for real-time view synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2

[32] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia, "Deep mesh reconstruction from single RGB images via topology modification networks," in *International Conference on Computer Vision (ICCV)*, 2019. 1

[33] H. Kato, Y. Ushiku, and T. Harada, "Neural 3D mesh renderer," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[34] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2Mesh: Generating 3D mesh models from single RGB images," in *European Conference on Computer Vision (ECCV)*, 2018. 1

[35] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit mapping and positioning in real-time," in *International Conference on Computer Vision (ICCV)*, 2021. 1

[36] H. Turki, D. Ramanan, and M. Satyanarayanan, "Mega-NeRF: Scalable construction of large-scale nerfs for virtual fly-throughs," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[37] P. Wang, Y. Liu, Z. Chen, L. Liu, Z. Liu, T. Komura, C. Theobalt, and W. Wang, "F$^2$-NeRF: Fast neural radiance field training with free camera trajectories," *arXiv preprint arXiv:2303.15951*, 2023. 2

[38] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[39] ——, "Improved direct voxel grid optimization for radiance fields reconstruction," *arXiv preprint arXiv:2206.05085*, 2022. 2

[40] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "TensoRF: Tensorial radiance fields," in *European Conference on Computer Vision (ECCV)*, 2022. 2

[41] A. Chen, Z. Xu, X. Wei, S. Tang, H. Su, and A. Geiger, "Factor fields: A unified framework for neural fields and beyond," *arXiv preprint arXiv:2302.01226*, 2023. 2

[42] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, "Depth-supervised NeRF: Fewer views and faster training for free," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022. 2, 3

[43] M. Tancik, B. Mildenhall, T. Wang, D. Schmidt, P. P. Srinivasan, J. T. Barron, and R. Ng, "Learned initializations for optimizing coordinate-based neural representations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[44] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "PixelNeRF: Neural radiance fields from one or few images," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[45] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser, "IBRNet: Learning multi-view image-based rendering," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[46] M. M. Johari, Y. Lepoittevin, and F. Fleuret, "GeoNeRF: Generalizing nerf with geometry priors," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[47] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, "FastNeRF: High-fidelity neural rendering at 200fps," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[48] K. Wadhwani and T. Kojima, "SqueezeNeRF: Further factorized FastNeRF for memory-efficient inference," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[49] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "KiloNeRF: Speeding up neural radiance fields with thousands of tiny MLPs," in *International Conference on Computer Vision (ICCV)*, 2021. 2

[50] A. Kurz, T. Neff, Z. Lv, M. Zollhöfer, and M. Steinberger, "AdaNeRF: Adaptive sampling for real-time rendering of neural radiance fields," in *European Conference on Computer Vision (ECCV)*, 2022. 2

[51] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics (TOG)*, 2023. 2

[52] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," in *International Conference on Computer Vision (ICCV)*, 2021. 2

[53] M. Piala and R. Clark, "TermiNeRF: Ray termination prediction for efficient neural rendering," in *International Conference on 3D Vision (3DV)*, 2021. 2

[54] J. Fang, L. Xie, X. Wang, X. Zhang, W. Liu, and Q. Tian, "Neusample: Neural sample field for efficient view synthesis," *arXiv preprint arXiv:2111.15552*, 2021. 2

[55] T. Neff, P. Stadlbauer, M. Parger, A. Kurz, J. H. Mueller, C. R. A. Chaitanya, A. Kaplanyan, and M. Steinberger, "DONeRF: Towards real-time rendering of compact neural radiance fields using depth oracle networks," in *Computer Graphics Forum*, 2021. 2

[56] H. Lin, S. Peng, Z. Xu, Y. Yan, Q. Shuai, H. Bao, and X. Zhou, "Efficient neural radiance fields for interactive free-viewpoint video," in *SIGGRAPH Asia 2022 Conference Papers*, 2022. 2

[57] N. Kondo, Y. Ikeda, A. Tagliasacchi, Y. Matsuo, Y. Ochiai, and S. S. Gu, "VaxNeRF: Revisiting the classic for voxel-accelerated neural radiance field," *arXiv preprint arXiv:2111.13112*, 2021. 2

[58] D. B. Lindell, J. N. Martel, and G. Wetzstein, "AutoInt: Automatic integration for fast neural volume rendering," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[59] L. Wu, J. Y. Lee, A. Bhattad, Y.-X. Wang, and D. Forsyth, "Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[60] V. Sitzmann, S. Rezchikov, B. Freeman, J. Tenenbaum, and F. Durand, "Light field networks: Neural scene representations with single-evaluation rendering," *Neural Information Processing Systems (NeurIPS)*, 2021. 2

[61] C. O. Smith, H.-X. Yu, S. Zakharov, F. Durand, J. B. Tenenbaum, J. Wu, and V. Sitzmann, "Unsupervised discovery and composition of object light fields," *Transactions on Machine Learning Research*, 2023. 2

[62] T. Yenamandra, A. Tewari, N. Yang, F. Bernard, C. Theobalt, and D. Cremers, "FIRe: Fast inverse rendering using directional and signed distance functions," *arXiv preprint arXiv:2203.16284*, 2022. 2

[63] B. Y. Feng, Y. Zhang, D. Tang, R. Du, and A. Varshney, "PRIF: Primary ray-based implicit function," in *European Conference on Computer Vision (ECCV)*, 2022. 2

[64] T. Aumentado-Armstrong, S. Tsogkas, S. Dickinson, and A. D. Jepson, "Representing 3D shapes with probabilistic directed distance fields," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[65] T. Houchens, C.-Y. Lu, S. Duggal, R. Fu, and S. Sridhar, "NeuralODF: Learning omnidirectional distance fields for 3D shape representation," *arXiv preprint arXiv:2206.05837*, 2022. 2

[66] Y.-C. Guo, Y.-P. Cao, C. Wang, Y. He, Y. Shan, X. Qie, and S.-H. Zhang, "VMesh: Hybrid volume-mesh representation for efficient view synthesis," *arXiv preprint arXiv:2303.16184*, 2023. 2

[67] V. Tschernezki, I. Laina, D. Larlus, and A. Vedaldi, "Neural feature fusion fields: 3D distillation of self-supervised 2D image representations," *arXiv preprint arXiv:2209.03494*, 2022. 2

[68] S. Kobayashi, E. Matsumoto, and V. Sitzmann, "Decomposing NeRF for editing via feature field distillation," *Neural Information Processing Systems (NeurIPS)*, 2022. 2

[69] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, "LeRF: Language embedded radiance fields," *arXiv preprint arXiv:2303.09553*, 2023. 2

[70] K. Blomqvist, F. Milano, J. J. Chung, L. Ott, and R. Siegwart, "Neural implicit vision-language feature fields," *arXiv preprint arXiv:2303.10962*, 2023. 2

[71] N. M. M. Shafiullah, C. Paxton, L. Pinto, S. Chintala, and A. Szlam, "CLIP-fields: Weakly supervised semantic fields for robotic memory," *arXiv preprint arXiv:2210.05663*, 2022. 2

[72] S. Zhi, E. Sucar, A. Mouton, I. Haughton, T. Laidlow, and A. J. Davison, "iLabel: Interactive neural scene labelling," *arXiv preprint arXiv:2111.14637*, 2021. 2

[73] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-place scene labelling and understanding with implicit scene representation," in *International Conference on Computer Vision (ICCV)*, 2021. 2

[74] A. Mirzaei, Y. Kant, J. Kelly, and I. Gilitschenski, "LaTeRF: Label and text driven object radiance fields," in *European Conference on Computer Vision (ECCV)*, 2022. 2

[75] J. Gu, L. Liu, P. Wang, and C. Theobalt, "StyleNeRF: A style-based 3D-aware generator for high-resolution image synthesis," *arXiv preprint arXiv:2110.08985*, 2021. 2, 4

[76] M. Niemeyer and A. Geiger, "GIRAFFE: Representing scenes as compositional generative neural feature fields," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 4

[77] Y. Xue, Y. Li, K. K. Singh, and Y. J. Lee, "GIRAFFE-HD: A high-resolution 3D-aware generative model," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[78] G. Metzer, E. Richardson, O. Patashnik, R. Giryes, and D. Cohen-Or, "Latent-NeRF for shape-guided generation of 3D shapes and textures," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[79] H. Seo, H. Kim, G. Kim, and S. Y. Chun, "DITTO-NeRF: Diffusion-based iterative text to omni-directional 3D model," *arXiv preprint arXiv:2304.02827*, 2023. 2

[80] E. R. Chan, K. Nagano, M. A. Chan, A. W. Bergman, J. J. Park, A. Levy, M. Aittala, S. De Mello, T. Karras, and G. Wetzstein, "Generative novel view synthesis with 3D-aware diffusion models," *arXiv preprint arXiv:2304.02602*, 2023. 2

[81] A. Tagliasacchi and B. Mildenhall, "Volume rendering digest (for NeRF)," *arXiv preprint arXiv:2209.02417*, 2022. 3

[82] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. M. Saragih, M. Nießner, R. Pandey, S. R. Fanello, G. Wetzstein, J. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B. Goldman, and M. Zollhofer, "State of the art on neural rendering," *Computer Graphics Forum*, vol. 39, no. 2, pp. 701–727, 2020. 3

[83] S. M. A. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, and D. Hassabis, "Neural scene representation and rendering," *Science*, vol. 360, no. 6394, pp. 1204–1210, 2018. 3

[84] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[85] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3, 6

[86] H. Huang, Z. Li, R. He, Z. Sun, and T. Tan, "IntroVAE: Introspective variational autoencoders for photographic image synthesis," *Neural Information Processing Systems (NeurIPS)*, 2018. 3

8

[87] T. Müller, "Tiny CUDA neural network framework," 2021, https://github.com/nvlabs/tiny-cuda-nn. 4

[88] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, 2015. 4

[89] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4

[90] S. Fu*, N. Tamir*, S. Sundaram*, L. Chai, R. Zhang, T. Dekel, and P. Isola, "DreamSim: Learning new dimensions of human visual similarity using synthetic data," *arXiv preprint arXiv:2306.09344*, 2023. 4

[91] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning (ICML)*, 2021. 4

[92] C. Chan, F. Durand, and P. Isola, "Learning to generate line drawings that convey geometry and semantics," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 4

[93] Y. Vinker, Y. Alaluf, D. Cohen-Or, and A. Shamir, "CLIPascene: Scene sketching with different types and levels of abstraction," *arXiv preprint arXiv:2211.17256*, 2022. 4

[94] J. Ho, W. Chan, C. Saharia, J. Whang, R. Gao, A. Gritsenko, D. P. Kingma, B. Poole, M. Norouzi, D. J. Fleet, and T. Salimans, "Imagen video: High definition video generation with diffusion models," *arXiv preprint arXiv:2210.02303*, 2022. 4

[95] Y. He, T. Yang, Y. Zhang, Y. Shan, and Q. Chen, "Latent video diffusion models for high-fidelity video generation with arbitrary lengths," *arXiv preprint arXiv:2211.13221*, 2022. 4

[96] H. Wu, E. Zhang, L. Liao, C. Chen, J. H. Hou, A. Wang, W. S. Sun, Q. Yan, and W. Lin, "Exploring video quality assessment on user generated contents from aesthetic and technical perspectives," in *International Conference on Computer Vision (ICCV)*, 2023. 4

[97] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, 2019. 4

[98] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," *arXiv preprint arXiv:1511.07289*, 2015. 6

[99] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv preprint arXiv:1607.08022*, 2016. 6