D-NeRF: Neural Radiance Fields for Dynamic Scenes

Albert Pumarola[1] Enric Corona[1] Gerard Pons-Moll[2,3] Francesc Moreno-Noguer[1]
[1]Institut de Robòtica i Informàtica Industrial, CSIC-UPC
[2]University of Tübingen
[3]Max Planck Institute for Informatics

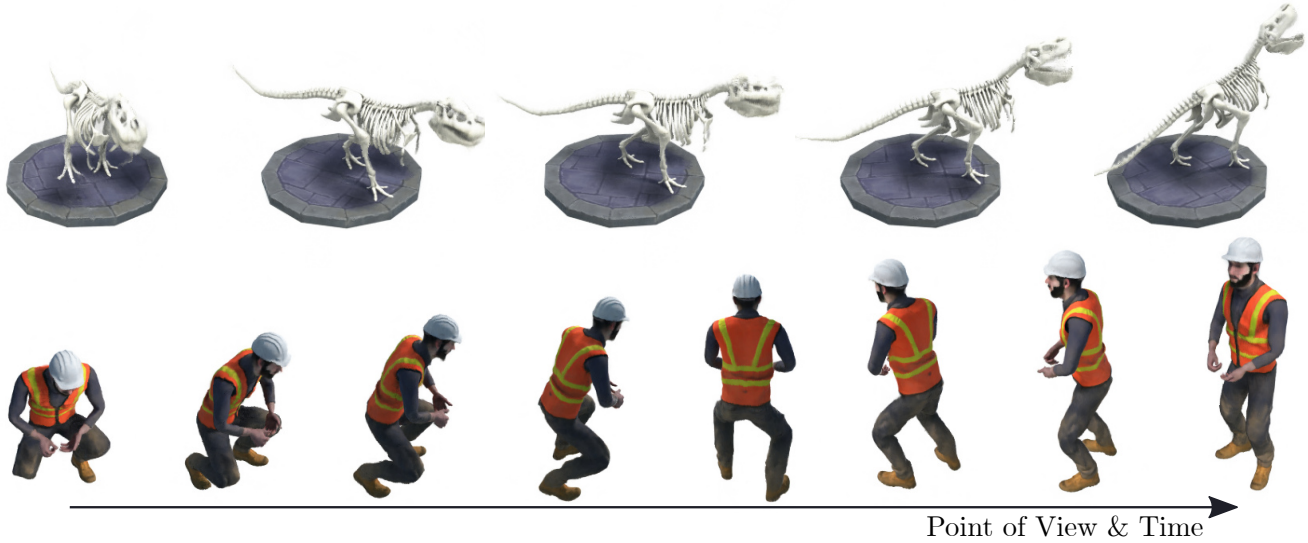

Figure 1: We propose D-NeRF, a method for synthesizing novel views, at an arbitrary point in time, of dynamic scenes with complex non-rigid geometries. We optimize an underlying deformable volumetric function from a sparse set of input monocular views without the need of ground-truth geometry nor multi-view images. The figure shows two scenes under variable points of view and time instances synthesised by the proposed model.

Abstract

Neural rendering techniques combining machine learning with geometric reasoning have arisen as one of the most promising approaches for synthesizing novel views of a scene from a sparse set of images. Among these, stands out the Neural radiance fields (NeRF) [31], which trains a deep network to map 5D input coordinates (representing spatial location and viewing direction) into a volume density and view-dependent emitted radiance. However, despite achieving an unprecedented level of photorealism on the generated images, NeRF is only applicable to static scenes, where the same spatial location can be queried from different images. In this paper we introduce D-NeRF, a method that extends neural radiance fields to a dynamic domain, allowing to reconstruct and render novel images of objects under rigid and non-rigid motions from a single *camera moving around the scene. For this purpose we consider time as an additional input to the system, and split the learning process in two main stages: one that encodes the scene into a canonical space and another that maps this canonical representation into the deformed scene at a particular time. Both mappings are simultaneously learned using fully-connected networks. Once the networks are trained, D-NeRF can render novel images, controlling both the camera view and the time variable, and thus, the object movement. We demonstrate the effectiveness of our approach on scenes with objects under rigid, articulated and non-rigid motions. Code, model weights and the dynamic scenes dataset will be available at [1].*

1. Introduction

Rendering novel photo-realistic views of a scene from a sparse set of input images is necessary for many applications in *e.g.* augmented reality, virtual reality, 3D content production, games and the movie industry. Recent advances in the emerging field of neural rendering, which learn scene representations encoding both geometry and appearance [31, 28, 24, 58, 34, 41], have achieved results that largely surpass those of traditional Structure-

from-Motion [18, 48, 44], light-field photography [22] and image-based rendering approaches [6]. For instance, the Neural Radiance Fields (NeRF) [31] have shown that simple multilayer perceptron networks can encode the mapping from 5D inputs (representing spatial locations (x, y, z) and camera views (θ, ϕ)) to emitted radiance values and volume density. This learned mapping allows then free-viewpoint rendering with extraordinary realism. Subsequent works have extended Neural Radiance Fields to images in the wild undergoing severe lighting changes [28] and have proposed sparse voxel fields for rapid inference [24]. Similar schemes have also been recently used for multi-view surface reconstruction [58] and learning surface light fields [35].

Nevertheless, all these approaches assume a *static* scene without moving objects. In this paper we relax this assumption and propose, to the best of our knowledge, the first end-to-end neural rendering system that is applicable to dynamic scenes, made of both still and moving/deforming objects. While there exist approaches for 4D view synthesis [3], our approach is different in that: 1) we only require a single camera; 2) we do not need to pre-compute a 3D reconstruction; and 3) our approach can be trained end-to-end.

Our idea is to represent the input of our system with a continuous 6D function, which besides 3D location and camera view, it also considers the time component t. Naively extending NeRF to learn a mapping from (x, y, z, t) to density and radiance does not produce satisfying results, as the temporal redundancy in the scene is not effectively exploited. Our observation is that objects can move and deform, but typically do not appear or disappear. Inspired by classical 3D scene flow [51], the core idea to build our method, denoted Dynamic-NeRF (D-NeRF in short), is to decompose learning in two modules. The first one learns a spatial mapping $(x, y, z, t) \rightarrow (\Delta x, \Delta y, \Delta z)$ between each point of the scene at time t and a *canonical scene* configuration. The second module regresses the scene radiance emitted in each direction and volume density given the tuple $(x + \Delta x, y + \Delta y, z + \Delta z, \theta, \phi)$. Both mappings are learned with deep fully connected networks without convolutional layers. The learned model then allows to synthesize novel images, providing control in the continuum (θ, ϕ, t) of the camera views and time component, or equivalently, the dynamic state of the scene (see Fig. 1).

We thoroughly evaluate D-NeRF on scenes undergoing very different types of deformation, from articulated motion to humans performing complex body poses. We show that by decomposing learning into a canonical scene and scene flow D-NeRF is able to render high-quality images while controlling both camera view and time components. As a side-product, our method is also able to produce complete 3D meshes that capture the time-varying geometry and which remarkably are obtained by observing the scene under a specific deformation only from one single viewpoint.

2. Related work

Neural implicit representation for 3D geometry. The success of deep learning on the 2D domain has spurred a growing interest in the 3D domain. Nevertheless, which is the most appropriate 3D data representation for deep learning remains an open question, especially for non-rigid geometry. Standard representations for rigid geometry include point-clouds [49, 39], voxels [17, 56] and octrees [52, 45]. Recently, there has been a strong burst in representing 3D data in an implicit manner via a neural network [29, 36, 7, 55, 9, 11, 16]. The main idea behind this approach is to describe the information (*e.g.* occupancy, distance to surface, color, illumination) of a 3D point $\mathbf{x}$ as the output of a neural network $f(\mathbf{x})$. Compared to the previously mentioned representations, neural implicit representations allow for continuous surface reconstruction at a low memory footprint.

The first works exploiting implicit representations [29, 36, 7, 55] for 3D representation were limited by their requirement of having access to 3D ground-truth geometry, often expensive or even impossible to obtain for in the wild scenes. Subsequent works relaxed this requirement by introducing a differentiable render allowing 2D supervision. For instance, [25] proposed an efficient ray-based field probing algorithm for efficient image-to-field supervision. [34, 57] introduced an implicit-based method to calculate the exact derivative of a 3D occupancy field surface intersection with a camera ray. In [43], a recurrent neural network was used to ray-cast the scene and estimate the surface geometry. Although these techniques have a great potential to represent 3D shapes in an unsupervised manner, they are typically limited to relatively simple geometries.

NeRF [31] showed that by implicitly representing a rigid scene using 5D radiance fields makes it possible to capture high-resolution geometry and photo-realistically rendering novel views. [28] extended this method to handle variable illumination and transient occlusions to deal with in the wild images. In [24], even more complex 3D surfaces were represented by using voxel-bounded implicit fields. And [58] relaxed the requirement of multiview camera calibration, and Stereo Radiance Fields [10] generalize NeRF to multiple scenes by integrating classical stereo within NeRF. None of the aforementioned methods can deal with dynamic and deformable scenes.

Neural implicit functions have been generalized to articulated objects and non-rigid objects [33, 13] but require full 3D ground-truth supervision. Neural volumes [26] produced high quality reconstruction results via voxel-based representation enhanced with an implicit voxel warp field, but they require a muti-view image capture setting.

To the best of our knowledge, D-NeRF is the first approach able to generate a neural implicit representation for non-rigid and time-varying scenes, trained solely on

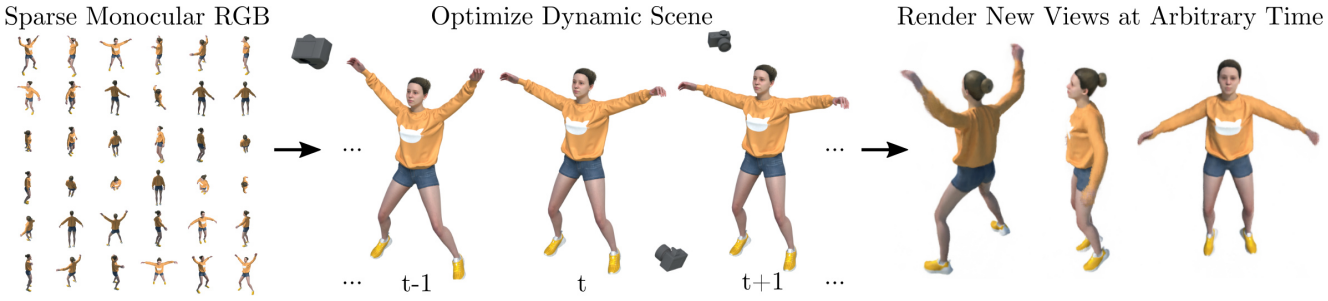

Figure 2: **Problem Definition.** Given a sparse set of images of a dynamic scene moving non-rigidly and being captured by a monocular camera, we aim to design a deep learning model to implicitly encode the scene and synthesize novel views at an arbitrary time. Here, we visualize a subset of the input training frames paired with accompanying camera parameters, and we show three novel views at three different time instances rendered by the proposed method.

monocular data without the need of 3D ground-truth supervision nor a multi-view camera setting. Concurrent to our work, other groups have also introduced dynamic generalizations of NeRF [54, 37, 23, 47].

Novel view synthesis. Novel view synthesis is a long standing vision and graphics problem that aims to synthesize new images from arbitrary view points of a scene captured by multiple images. Most traditional approaches for rigid scenes consist on reconstructing the scene from multiple views with Structure-from-Motion [18] and bundle adjustment [48], while other approaches propose light-field based photography [22]. More recently, deep learning based techniques [42, 20, 14, 12, 30] are able to learn a neural volumetric representation from a set of sparse images.

However, none of these methods can synthesize novel views of dynamic scenes. To tackle non-rigid scenes most methods approach the problem by reconstructing a dynamic 3D textured mesh. 3D reconstruction of non-rigid surfaces from monocular images is known to be severely ill-posed. Structure-from-Template (SfT) approaches [4, 8, 32] recover the surface geometry given a reference known template configuration. Temporal information is another prior typically exploited. Non-rigid-Structure-from-Motion (NRSfM) techniques [46, 2] exploit temporal information. Yet, SfT and NRSfM require either 2D-to-3D matches or 2D point tracks, limiting their general applicability to relatively well-textured surfaces and mild deformations.

Some of these limitations are overcome by learning based techniques, which have been effectively used for synthesizing novel photo-realistic views of dynamic scenes. For instance, [3, 62, 19] capture the dynamic scene at the same time instant from multiple views, to then generate 4D space-time visualizations. [15, 38, 61] also leverage on simultaneously capturing the scene from multiple cameras to estimate depth, completing areas with missing information and then performing view synthesis. In [59], the need of multiple views is circumvented by using a pre-trained network that estimates a per frame depth. This depth, jointly with the optical flow and consistent depth estimation across frames, are then used to interpolate between images and render novel views. Nevertheless, by decoupling depth estimation from novel view synthesis, the outcome of this approach becomes highly dependent on the quality of the depth maps as well as on the reliability of the optical flow. Very recently, X-Fields [5] introduced a neural network to interpolate between images taken across different view, time or illumination conditions. However, while this approach is able to process dynamic scenes, it requires more than one view. Since no 3D representation is learned, variation in viewpoint is small.

D-NeRF is different from all prior work in that it does not require 3D reconstruction, can be learned end-to-end, and requires *a single view* per time instance. Another appealing characteristic of D-NeRF is that it inherently learns a time-varying 3D volume density and emitted radiance, which turns the novel view synthesis into a ray-casting process instead of a view interpolation, which is remarkably more robust to rendering images from arbitrary viewpoints.

3. Problem Formulation

Given a sparse set of images of a dynamic scene captured with a monocular camera, we aim to design a deep learning model able to implicitly encode the scene and synthesize novel views at an arbitrary time (see Fig. 2).

Formally, our goal is to learn a mapping $\mathcal{M}$ that, given a 3D point $\mathbf{x} = (x, y, z)$, outputs its emitted color $\mathbf{c} = (r, g, b)$ and volume density σ conditioned on a time instant t and view direction $\mathbf{d} = (\theta, \phi)$. That is, we seek to estimate the mapping $\mathcal{M} : (\mathbf{x}, \mathbf{d}, t) \rightarrow (\mathbf{c}, \sigma)$.

An intuitive solution would be to directly learn the transformation $\mathcal{M}$ from the 6D space $(\mathbf{x}, \mathbf{d}, t)$ to the 4D space $(\mathbf{c}, \sigma)$. However, as we will show in the results section, we obtain consistently better results by splitting the mapping $\mathcal{M}$ into Ψ_x and Ψ_t, where Ψ_x represents the scene in canonical configuration and Ψ_t a mapping between the scene at time instant t and the canonical one. More precisely, given a point $\mathbf{x}$ and viewing direction $\mathbf{d}$ at time instant t we first transform the point position to its canonical configuration

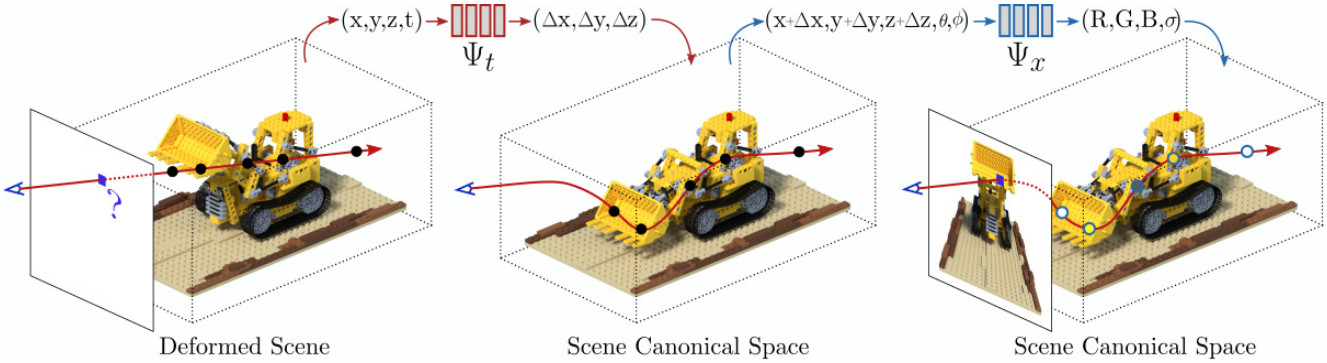

Figure 3: **D-NeRF Model**. The proposed architecture consists of two main blocks: a deformation network Ψ_t mapping all scene deformations to a common canonical configuration; and a canonical network Ψ_x regressing volume density and view-dependent RGB color from every camera ray.

as $\Psi_t : (\mathbf{x}, t) \rightarrow \Delta\mathbf{x}$. Without loss of generality, we chose $t = 0$ as the canonical scene $\Psi_t : (\mathbf{x}, 0) \rightarrow \mathbf{0}$. By doing so the scene is no longer independent between time instances, and becomes interconnected through a common canonical space anchor. Then, the assigned emitted color and volume density under viewing direction $\mathbf{d}$ equal to those in the canonical configuration $\Psi_x : (\mathbf{x} + \Delta\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$.

We propose to learn Ψ_x and Ψ_t using a sparse set of T RGB images $\{\mathbf{I}_t, \mathbf{T}_t\}_{t=1}^{T}$ captured with a monocular camera, being $\mathbf{I}_t \in \mathbb{R}^{H \times W \times 3}$ the image acquired under camera pose $\mathbf{T}_t \in \mathbb{R}^{4\times4}$ SE(3), at time t. Although we could assume multiple views per time instance, we want to test the limits of our method, and assume a single image per time instance. That is, we do not observe the scene under a specific configuration/deformation state from different viewpoints.

4. Method

We now introduce D-NeRF, our novel neural renderer for view synthesis trained solely from a sparse set of images of a dynamic scene. We build on NeRF [31] and generalize it to handle non-rigid scenes. Recall that NeRF requires multiple views of a rigid scene In contrast, D-NeRF can learn a volumetric density representation for continuous non-rigid scenes trained with a single view per time instant.

As shown in Fig. 3, D-NeRF consists of two main neural network modules, which parameterize the mappings explained in the previous section Ψ_t, Ψ_x. On the one hand we have the *Canonical Network*, an MLP (multilayer perceptron) $\Psi_x(\mathbf{x}, \mathbf{d}) \mapsto (\mathbf{c}, \sigma)$ is trained to encode the scene in the canonical configuration such that given a 3D point $\mathbf{x}$ and a view direction $\mathbf{d}$ returns its emitted color $\mathbf{c}$ and volume density σ. The second module is called *Deformation Network* and consists of another MLP $\Psi_t(\mathbf{x}, t) \mapsto \Delta\mathbf{x}$ which predicts a deformation field defining the transformation between the scene at time t and the scene in its canonical configuration. We next describe in detail each one of these blocks (Sec. 4.1), their interconnection for volume rendering (Sec. 4.2) and how are they learned (Sec. 4.3).

4.1. Model Architecture

Canonical Network. With the use of a canonical configuration we seek to find a representation of the scene that brings together the information of all corresponding points in all images. By doing this, the missing information from a specific viewpoint can then be retrieved from that canonical configuration, which shall act as an anchor interconnecting all images.

The canonical network Ψ_x is trained so as to encode volumetric density and color of the scene in canonical configuration. Concretely, given the 3D coordinates $\mathbf{x}$ of a point, we first encode it into a 256-dimensional feature vector. This feature vector is then concatenated with the camera viewing direction $\mathbf{d}$, and propagated through a fully connected layer to yield the emitted color $\mathbf{c}$ and volume density σ for that given point in the canonical space.

Deformation Network. The deformation network Ψ_t is optimized to estimate the deformation field between the scene at a specific time instant and the scene in canonical space. Formally, given a 3D point $\mathbf{x}$ at time t, Ψ_t is trained to output the displacement $\Delta\mathbf{x}$ that transforms the given point to its position in the canonical space as $\mathbf{x} + \Delta\mathbf{x}$. For all experiments, without loss of generality, we set the canonical scene to be the scene at time $t = 0$:

$$\Psi_t(\mathbf{x}, t) = \begin{cases} \Delta\mathbf{x}, & \text{if } t \neq 0 \\ 0, & \text{if } t = 0 \end{cases} \tag{1}$$

As shown in previous works [40, 50, 31], directly feeding raw coordinates and angles to a neural network results in low performance. Thus, for both the canonical and the deformation networks, we first encode $\mathbf{x}$, $\mathbf{d}$ and t into a higher dimension space. We use the same positional encoder as in [31] where $\gamma(p) =< (\sin(2^l\pi p), \cos(2^l\pi p)) >_0^L$. We independently apply the encoder $\gamma(\cdot)$ to each coordinate and camera view component, using $L = 10$ for $\mathbf{x}$, and $L = 4$ for $\mathbf{d}$ and t.

4.2. Volume Rendering

We now adapt NeRF volume rendering equations to account for non-rigid deformations in the proposed 6D neural radiance field. Let $\mathbf{x}(h) = \mathbf{o} + h\mathbf{d}$ be a point along the camera ray emitted from the center of projection $\mathbf{o}$ to a pixel p. Considering near and far bounds h_n and h_f in that ray, the expected color C of the pixel p at time t is given by:

$$C(p,t) = \int_{h_n}^{h_f} \mathcal{T}(h,t)\sigma(\mathbf{p}(h,t))\mathbf{c}(\mathbf{p}(h,t),\mathbf{d})dh, \quad (2)$$

$$\text{where} \quad \mathbf{p}(h,t) = \mathbf{x}(h) + \Psi_t(\mathbf{x}(h),t), \quad (3)$$

$$[\mathbf{c}(\mathbf{p}(h,t),\mathbf{d}), \sigma(\mathbf{p}(h,t))] = \Psi_x(\mathbf{p}(h,t),\mathbf{d}), \quad (4)$$

$$\text{and} \quad \mathcal{T}(h,t) = \exp\left(-\int_{h_n}^{h} \sigma(\mathbf{p}(s,t))ds\right). \quad (5)$$

The 3D point $\mathbf{p}(h,t)$ denotes the point on the camera ray $\mathbf{x}(h)$ transformed to canonical space using our Deformation Network Ψ_t, and $\mathcal{T}(h,t)$ is the accumulated probability that the ray emitted from h_n to h_f does not hit any other particle. Notice that the density σ and color $\mathbf{c}$ are predicted by our Canonical Network Ψ_x.

As in [31] the volume rendering integrals in Eq. (2) and Eq. (5) can be approximated via numerical quadrature. To select a random set of quadrature points $\{h_n\}_{n=1}^{N} \in [h_n, h_f]$ a stratified sampling strategy is applied by uniformly drawing samples from evenly-spaced ray bins. A pixel color is approximated as:

$$C'(p,t) = \sum_{n=1}^{N} \mathcal{T}'(h_n,t)\alpha(h_n,t,\delta_n)\mathbf{c}(\mathbf{p}(h_n,t),\mathbf{d}), \quad (6)$$

$$\text{where} \quad \alpha(h,t,\delta) = 1 - \exp(-\sigma(\mathbf{p}(h,t))\delta), \quad (7)$$

$$\text{and} \quad \mathcal{T}'(h_n,t) = \exp\left(-\sum_{m=1}^{n-1} \sigma(\mathbf{p}(h_m,t))\delta_m\right), \quad (8)$$

and $\delta_n = h_{n+1} - h_n$ is the distance between two quadrature points.

4.3. Learning the Model

The parameters of the canonical Ψ_x and deformation Ψ_t networks are simultaneously learned by minimizing the mean squared error with respect to the T RGB images $\{\mathbf{I}_t\}_{t=1}^{T}$ of the scene and their corresponding camera pose matrices $\{\mathbf{T}_t\}_{t=1}^{T}$. Recall that every time instant is only acquired by a single camera.

At each training batch, we first sample a random set of pixels $\{p_{t,i}\}_{i=1}^{N_s}$ corresponding to the rays cast from some camera position $\mathbf{T}_t$ to some pixels i of the corresponding RGB image t. We then estimate the colors of the chosen pixels using Eq. (6). The training loss we use is the mean squared error between the rendered and real pixels:

$$\mathcal{L} = \frac{1}{N_s}\sum_{i=1}^{N_s} \left\| \hat{C}(p,t) - C'(p,t) \right\|_2^2 \quad (9)$$

where $\hat{C}$ are the pixels' ground-truth color.

5. Implementation Details

Both the canonical network Ψ_x and the deformation network Ψ_t consists on simple 8-layers MLPs with ReLU activations. For the canonical network a final sigmoid non-linearity is applied to $\mathbf{c}$ and σ. No non-linearlity is applied to $\mathbf{\Delta x}$ in the deformation network.

For all experiments we set the canonical configuration as the scene state at $t = 0$ by enforcing it in Eq. (1). To improve the networks convergence, we sort the input images according to their time stamps (from lower to higher) and then we apply a curriculum learning strategy where we incrementally add images with higher time stamps.

The model is trained with 400×400 images during $800k$ iterations with a batch size of $N_s = 4096$ rays, each sampled 64 times along the ray. As for the optimizer, we use Adam [21] with learning rate of $5e-4$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and exponential decay to $5e-5$. The model is trained with a single Nvidia® GTX 1080 for 2 days.

6. Experiments

This section provides a thorough evaluation of our system. We first test the main components of the model, namely the canonical and deformation networks (Sec. 6.1). We then compare D-NeRF against NeRF and T-NeRF, a variant in which does not use the canonical mapping (Sec. 6.2). Finally, we demonstrate D-NeRF ability to synthesize novel views at an arbitrary time in several complex dynamic scenes (Sec. 6.3).

In order to perform an exhaustive evaluation we have extended NeRF [31] rigid benchmark with eight scenes containing dynamic objects under large deformations and realistic non-Lambertian materials. As in the rigid benchmark of [31], six are rendered from viewpoints sampled from the upper hemisphere, and two are rendered from viewpoints sampled on the full sphere. Each scene contains between 100 and 200 rendered views depending on the action time span, all at 800 × 800 pixels. We will release the path-traced images with defined train/validation/test splits for these eight scenes.

6.1. Dissecting the Model

This subsection provides insights about D-NeRF behaviour when modeling a dynamic scene and analyze the two main modules, namely the canonical and deformation networks.

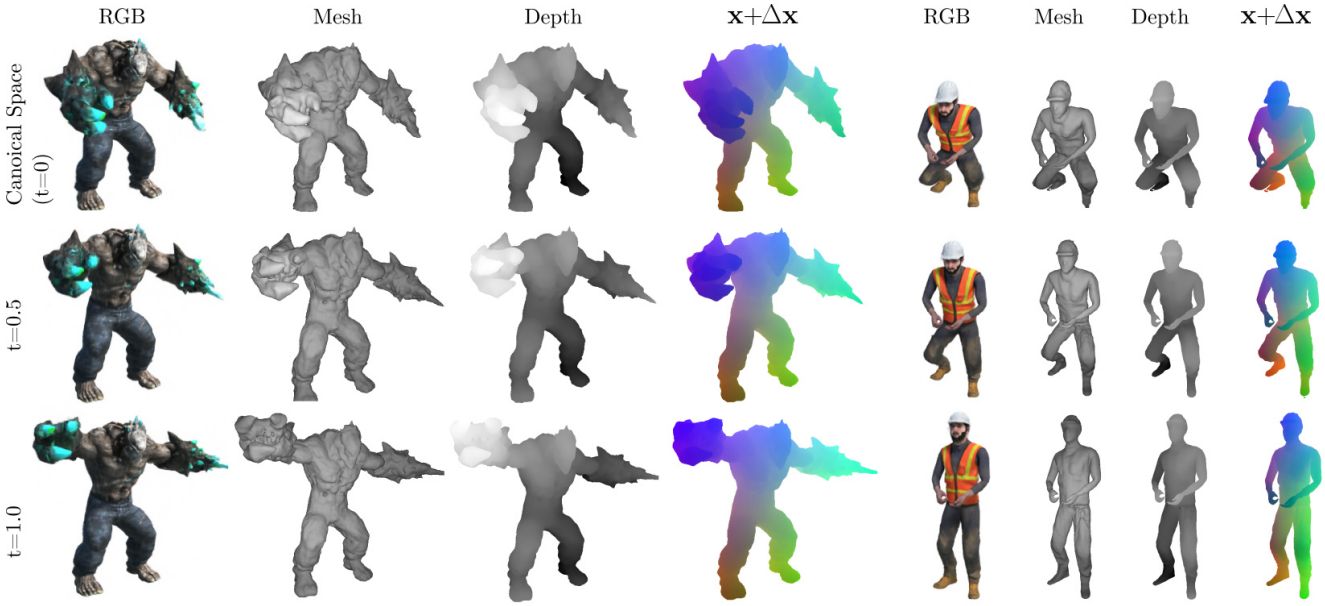

Figure 4: **Visualization of the Learned Scene Representation.** From left to right: the learned radiance from a specific viewpoint, the volume density represented as a 3D mesh and a depth map, and the color-coded points of the canonical configuration mapped to the deformed meshes based on $\Delta\mathbf{x}$. The same colors on corresponding points indicate the correctness of such mapping.

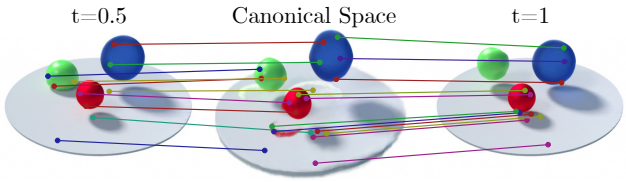

Figure 5: **Analyzing Shading Effects.** Pairs of corresponding points between the canonical space and the scene at times $t = 0.5$ and $t = 1$.

We initially evaluate the ability of the canonical network to represent the scene in a canonical configuration. The results of this analysis for two scenes are shown the first row of Fig. 4 (columns 1-3 in each case). The plots show, for the canonical configuration ($t = 0$), the RGB image, the 3D occupancy network and the depth map, respectively. The rendered RGB image is the result of evaluating the canonical network on rays cast from an arbitrary camera position applying Eq. (6). To better visualize the learned volumetric density we transform it into a mesh applying marching cubes [27], with a 3D cube resolution of 256^3 voxels. Note how D-NeRF is able to model fine geometric and appearance details for complex topologies and texture patterns, even when it was only trained with a set of sparse images, each under a different deformation.

In a second experiment we assess the capacity of the network to estimate consistent deformation fields that map the canonical scene to the particular shape at each input image. The second and third rows of Fig. 4 show the result of applying the corresponding translation vectors to the canonical space for $t = 0.5$ and $t = 1$. The fourth column in each of the two examples visualizes the displacement field, where the color-coded points in the canonical shape ($t = 0$) at mapped to the different shape configurations at $t = 0.5$ and $t = 1$. Note that the colors consistency along time, indicating that the displacement field is correctly estimated.

Another question that we try to answer is how D-NeRF manages to model phenomena like shadows/shading effects, that is, how the model can encode changes of appearance of the same point along time. We have carried an additional experiment to answer this. In Fig. 5 we show a scene with three balls, made of very different materials (plastic –green–, translucent glass –blue– and metal –red–). The figure plots pairs of corresponding points between the canonical configuration and the scene at a specific time instant. D-NeRF is able to synthesize the shading effects by warping the canonical configuration. For instance, observe how the floor shadows are warped along time. Note that the points in the shadow of, *e.g.* the red ball, at $t = 0.5$ and $t = 1$ map at different regions of the canonical space.

6.2. Quantitative Comparison

We next evaluate the quality of D-NeRF on the novel view synthesis problem and compare it against the original NeRF [31], which represents the scene using a 5D input (x, y, z, θ, ϕ), and T-NeRF, a straight-forward extension of NeRF in which the scene is represented by a 6D input $(x, y, z, \theta, \phi, t)$, without considering the intermediate canonical configuration of D-NeRF.

Table 1 summarizes the quantitative results on the 8 dynamic scenes of our dataset. We use several metrics for the evaluation: Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM) [53] and Learned Perceptual Image Patch Similarity (LPIPS) [60].

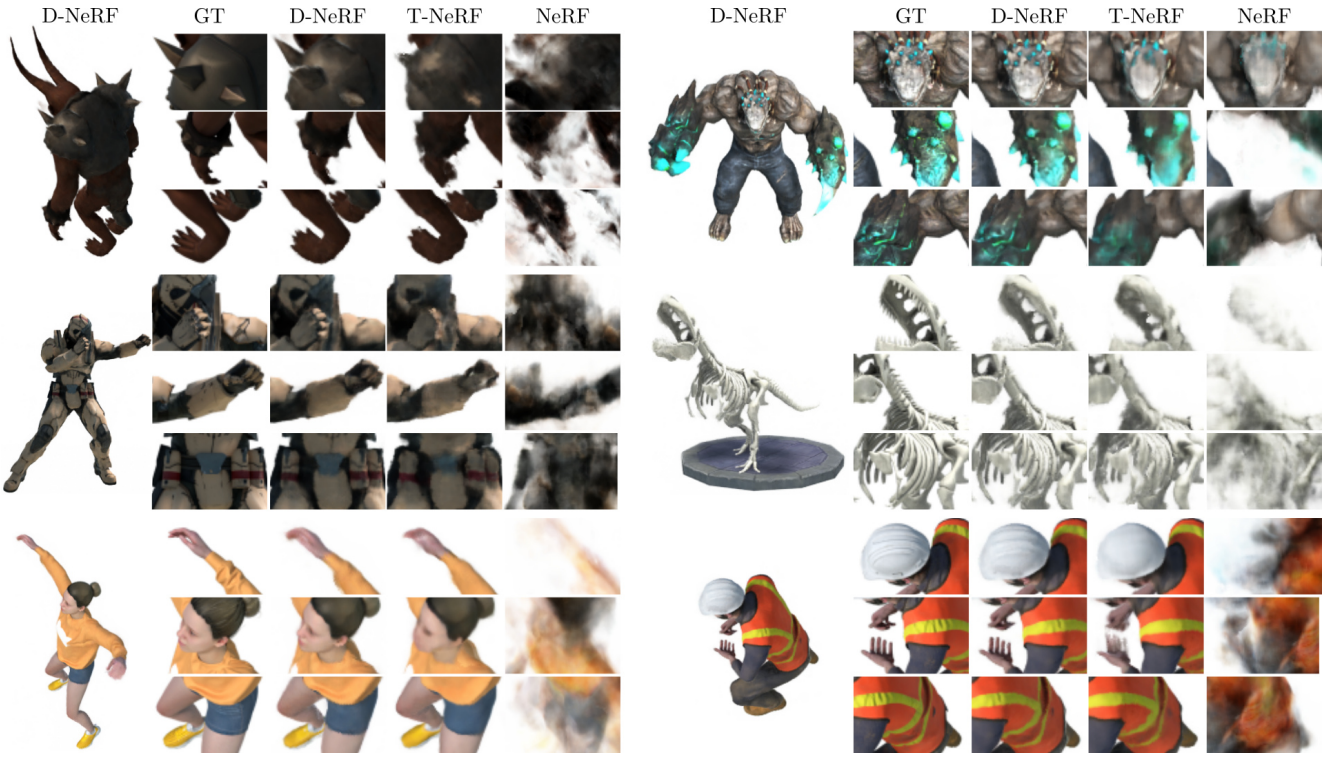

Figure 6: **Qualitative Comparison.** Novel view synthesis results of dynamic scenes. For every scene we show an image synthesised from a novel view at an arbitrary time by our method, and three close-ups for: ground-truth, NeRF, T-NeRF, and D-NeRF (ours).

	Hell Warrior				Mutant				Hook				Bouncing Balls			
Method	MSE↓	PSNR↑	SSIM↑	LPIPS↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓
NeRF	44e-3	13.52	0.81	0.25	9e-4	20.31	0.91	0.09	21e-3	16.65	0.84	0.19	94e-4	20.26	0.91	0.2
T-NeRF	47e-4	23.19	0.93	0.08	8e-4	30.56	0.96	0.04	18e-4	27.21	0.94	**0.06**	16e-5	37.81	0.98	0.12
D-NeRF	**31e-4**	**25.02**	**0.95**	**0.06**	**7e-4**	**31.29**	**0.97**	**0.02**	**11e-4**	**29.25**	**0.96**	0.11	**12e-5**	**38.93**	**0.98**	**0.1**
	Lego				T-Rex				Stand Up				Jumping Jacks			
Method	MSE↓	PSNR↑	SSIM↑	LPIPS↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓
NeRF	9e-3	20.30	0.79	0.23	3e-3	24.49	0.93	0.13	1e-2	18.19	0.89	0.14	1e-2	18.28	0.88	0.23
T-NeRF	**3e-4**	**23.82**	**0.90**	**0.15**	9e-3	30.19	0.96	0.13	7e-4	31.24	0.97	0.02	6e-4	32.01	0.97	0.03
D-NeRF	6e-4	21.64	0.83	0.16	**6e-3**	**31.75**	**0.97**	**0.03**	**5e-4**	**32.79**	**0.98**	**0.02**	**5e-4**	**32.80**	**0.98**	**0.03**

Table 1: **Quantitative Comparison.** We report MSE/LPIPS (lower is better) and PSNR/SSIM (higher is better).

In Fig. 6 we show samples of the estimated images under a novel view for visual inspection. As expected, NeRF is not able to model the dynamics scenes as it was designed for rigid cases, always converging to a blurry mean representation of all deformations. On the other hand, T-NeRF baseline is able to capture reasonably well the dynamics, although is not able to retrieve high frequency details. For example, in Fig. 6 top-left image it fails to encode the shoulder pad spikes, and in the top-right scene it is not able to model the stones and cracks. D-NeRF, instead, retains high details of the original image in the novel views. This is quite remarkable, considering that each deformation state has only been seen from a single viewpoint.

6.3. Additional Results

We finally show additional results to showcase the wide range of scenarios that can be handled with D-NeRF (Fig. 7). The first column displays the canonical configuration. Note that we are able to handle several types of dynamics: articulated motion in the *Tractor* scene; human motion in the *Jumping Jacks* and *Warrior* scenes; and asynchronous motion of several *Bouncing Balls*. Also note that the canonical configuration is a sharp and neat scene, in all cases, expect for the Jumping Jacks, where the two arms appear to be blurry. This, however, does not harm the quality of the rendered images, indicating that the network is able warp the canonical configuration so as to maximize the rendering quality. This is indeed consistent with Sec. 6.1 insights on how the network is able to encode shading.

D-NeRF has two main failure cases: (i) Poor camera poses (as in NeRF). (ii) Large deformations between temporally consecutive input images prevents the model from converging to a consistent deformation field. This can be solved by increasing the capture frame rate.

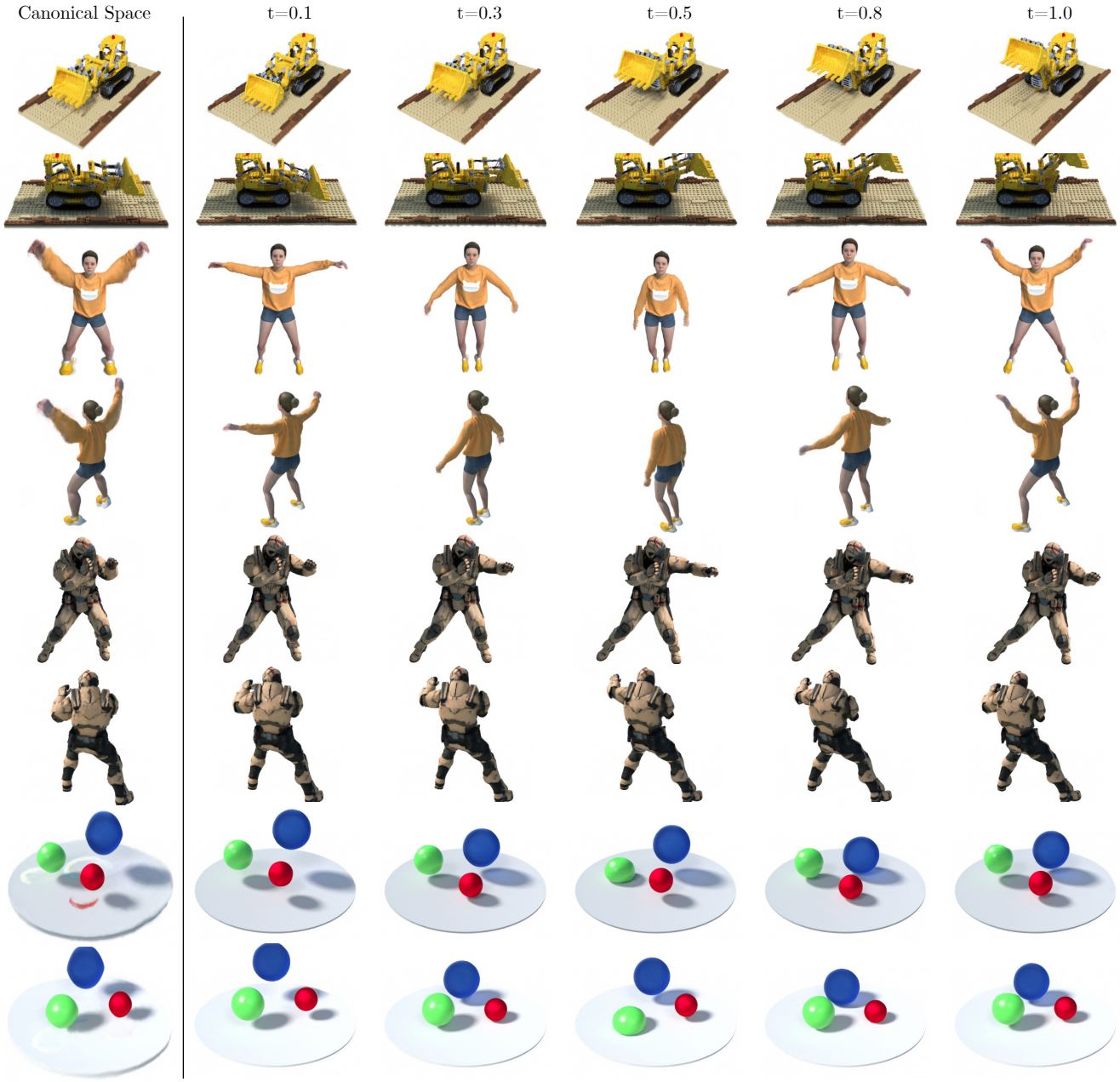

Figure 7: **Time & View Conditioning.** Results of synthesising diverse scenes from two novel points of view across time and the learned canonical space. For every scene we also display the learned scene canonical space in the first column.

7. Conclusion

We have presented D-NeRF, a novel neural radiance field approach for modeling dynamic scenes. Our method can be trained end-to-end from only a sparse set of images acquired with a moving camera, and does not require precomputed 3D priors nor observing the same scene configuration from different viewpoints. The main idea behind D-NeRF is to represent time-varying deformations with two modules: one that learns a canonical configuration, and another that learns the displacement field of the scene at each time instant w.r.t. the canonical space. A thorough evaluation demonstrates that D-NeRF is able to synthesise high quality novel views of scenes undergoing different types of deformation, from articulated objects to human bodies performing complex body postures.

Acknowledgments This work is supported in part by a Google Daydream Research award and by the Spanish government with the project HuMoUR TIN2017-90086-R, the ERA-Net Chistera project IPALM PCI2019-103386 and María de Maeztu Seal of Excellence MDM-2016-0656. Gerard Pons-Moll is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 409792180 (Emmy Noether Programme, project: Real Virtual Humans).

References

[1] https://www.albertpumarola.com/research/D-NeRF/index.html. 1

[2] Antonio Agudo and Francesc Moreno-Noguer. Simultaneous pose and non-rigid shape with particle dynamics. In *CVPR*, 2015. 3

[3] Aayush Bansal, Minh Vo, Yaser Sheikh, Deva Ramanan, and Srinivasa Narasimhan. 4d visualization of dynamic events from unconstrained multi-view videos. In *CVPR*, 2020. 2, 3

[4] Adrien Bartoli, Yan Gérard, Francois Chadebecq, Toby Collins, and Daniel Pizarro. Shape-from-template. *T-PAMI*, 37(10), 2015. 3

[5] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *TOG*, 39(6), 2020. 3

[6] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *SIGGRAPH*, 2001. 2

[7] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 2

[8] Ajad Chhatkuli, Daniel Pizarro, and Adrien Bartoli. Stable template-based isometric 3d reconstruction in all imaging conditions by linear least-squares. In *CVPR*, 2014. 3

[9] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6970–6981, 2020. 2

[10] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In *CVPR*. IEEE, jun 2021. 2

[11] Julian Chibane, Aymen Mir, and Gerard Pons-Moll. Neural unsigned distance fields for implicit function learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, December 2020. 2

[12] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *CVPR*, 2019. 3

[13] Boyang Deng, JP Lewis, Timothy Jeruzalski, Gerard Pons-Moll, Geoffrey Hinton, Mohammad Norouzi, and Andrea Tagliasacchi. Nasa: Neural articulated shape approximation. In *The European Conference on Computer Vision (ECCV)*, August 2020. 2

[14] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *CVPR*, 2019. 3

[15] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016. 3

[16] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4857–4866, 2020. 2

[17] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016. 2

[18] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2, 3

[19] Hanqing Jiang, Haomin Liu, Ping Tan, Guofeng Zhang, and Hujun Bao. 3d reconstruction of dynamic scenes with multiple handheld cameras. In *ECCV*, 2012. 3

[20] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *NeurIPS*, 2017. 3

[21] Diederik Kingma and Jimmy Ba. ADAM: A method for stochastic optimization. In *ICLR*, 2015. 5

[22] Marc Levoy and Pat Hanrahan. Light field rendering. In *SIGGRAPH*, 1996. 2, 3

[23] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv preprint arXiv:2011.13084*, 2020. 3

[24] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *arXiv preprint arXiv:2007.11571*, 2020. 1, 2

[25] Shichen Liu, Shunsuke Saito, Weikai Chen, and Hao Li. Learning to infer implicit surfaces without 3d supervision. In *NeurIPS*, 2019. 2

[26] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *TOG*, 38(4), 2019. 2

[27] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 1987. 6

[28] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *arXiv preprint arXiv:2008.02268*, 2020. 1, 2

[29] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 2

[30] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *TOG*, 2019. 3

[31] 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. *arXiv preprint arXiv:2003.08934*, 2020. 1, 2, 4, 5, 6

[32] F. Moreno-Noguer and P. Fua. Stochastic exploration of ambiguities for nonrigid shape recovery. *T-PAMI*, 35(2), 2013. 3

[33] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *ICCV*, 2019. 2

[34] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learn-

ing implicit 3d representations without 3d supervision. In *CVPR*, 2020. 1, 2

[35] Michael Oechsle, Michael Niemeyer, Lars Mescheder, Thilo Strauss, and Andreas Geiger. Learning implicit surface light fields. *arXiv preprint arXiv:2003.12406*, 2020. 2

[36] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[37] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo-Martin Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020. 3

[38] Julien Philip and George Drettakis. Plane-based multi-view inpainting for image-based rendering in large scenes. In *SIGGRAPH*, 2018. 3

[39] Albert Pumarola, Stefan Popov, Francesc Moreno-Noguer, and Vittorio Ferrari. C-flow: Conditional generative flow models for images and 3d point clouds. In *CVPR*, 2020. 2

[40] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *ICML*, 2019. 4

[41] Konstantinos Rematas and Vittorio Ferrari. Neural voxel renderer: Learning an accurate and controllable rendering tool. In *CVPR*, 2020. 1

[42] Liyue Shen, Wei Zhao, and Lei Xing. Patient-specific reconstruction of volumetric computed tomography images from a single projection view via deep learning. *Nature biomedical engineering*, 3(11), 2019. 3

[43] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *NeurIPS*, 2019. 2

[44] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *SIGGRAPH*, 2006. 2

[45] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs. In *ICCV*, 2017. 2

[46] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *IJCV*, 9(2), 1992. 3

[47] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a deforming scene from monocular video. *arXiv preprint arXiv:2012.12247*, 2020. 3

[48] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, 1999. 2, 3

[49] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *CVPR*, 2017. 2

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 4

[51] Sundar Vedula, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. *IEEE transactions on pattern analysis and machine intelligence*, 27(3):475–480, 2005. 2

[52] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis. *TOG*, 36(4), 2017. 2

[53] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 13(4), 2004. 6

[54] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. *arXiv preprint arXiv:2011.12950*, 2020. 3

[55] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In *NeurIPS*, 2019. 2

[56] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective Transformer Nets: Learning Single-view 3D object Reconstruction without 3D Supervision. In *NIPS*, 2016. 2

[57] Lior Yariv, Matan Atzmon, and Yaron Lipman. Universal differentiable renderer for implicit neural representations. *arXiv preprint arXiv:2003.09852*, 2020. 2

[58] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *NeurIPS*, 2020. 1, 2

[59] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *CVPR*, 2020. 3

[60] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 6

[61] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *TOG*, 37(4), 2018. 3

[62] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *TOG*, 23(3), 2004. 3