NeRF-Editing: Geometry Editing of Neural Radiance Fields

Yu-Jie Yuan [1,2,†] **Yang-Tian Sun** [1,2,†] **Yu-Kun Lai** [3]
Yuewen Ma [4] **Rongfei Jia** [4] **Lin Gao** [1,2*]
[1]Beijing Key Laboratory of Mobile Computing and Pervasive Device,
Institute of Computing Technology, Chinese Academy of Sciences
[2]School of Computer and Control Engineering, University of Chinese Academy of Sciences
[3]School of Computer Science & Informatics, Cardiff University [4]Alibaba Group
{yuanyujie, sunyangtian, gaolin}@ict.ac.cn LaiY4@cardiff.ac.uk
{yuewen.my, rongfei.jrf}@alibaba-inc.com

Abstract

Implicit neural rendering, especially Neural Radiance Field (NeRF), has shown great potential in novel view synthesis of a scene. However, current NeRF-based methods cannot enable users to perform user-controlled shape deformation in the scene. While existing works have proposed some approaches to modify the radiance field according to the user's constraints, the modification is limited to color editing or object translation and rotation. In this paper, we propose a method that allows users to perform controllable shape deformation on the implicit representation of the scene, and synthesizes the novel view images of the edited scene without re-training the network. Specifically, we establish a correspondence between the extracted explicit mesh representation and the implicit neural representation of the target scene. Users can first utilize well-developed mesh-based deformation methods to deform the mesh representation of the scene. Our method then utilizes user edits from the mesh representation to bend the camera rays by introducing a tetrahedra mesh as a proxy, obtaining the rendering results of the edited scene. Extensive experiments demonstrate that our framework can achieve ideal editing results not only on synthetic data, but also on real scenes captured by users.

1. Introduction

Novel view synthesis has been extensively studied in computer vision and computer graphics. In particular, the recently proposed neural radiance field (NeRF) [43] has inspired a large number of follow-up works aiming to achieve better visual effects [36], faster rendering speed [18, 77], generalization to different scenes [78], relighting [4,60], applying to dynamic scenes [48], and reducing the number of inputs [29]. However, as an implicit modeling method, the neural radiance field is difficult for users to edit or modify the scene objects, which is relatively easy with the explicit representation. The mesh representation, as a kind of explicit representation, is commonly used in shape modeling and rendering. There is a lot of research work on mesh deformation or editing [80]. However, it is difficult to obtain an accurate explicit representation of a real-world scene. From a sparse set of images, one can use some Multi-View Stereo (MVS) method [52] to reconstruct the point cloud or mesh representation of the scene, but the quality is generally poor. Rendering the reconstructed representation under novel views will lead to unrealistic results. Therefore, based on the promising novel view synthesis ability of implicit representations, such as NeRFs, further studying how to edit the implicit representation has become a new exploration direction.

Some works have already studied how to edit NeRF. For example, EditingNeRF [38] was the first to propose editing on the implicit radiance field. They train on a set of synthetic models from the same category, such as chairs and tables from ShapeNet [5], and introduce shape code and color code to represent the geometry and appearance of different models, respectively. The user selects a desired color and draw a few coarse scribbles on an image of a specified view to indicate what should be changed. Then local edits are propagated to 3D regions through updating the network based on the loss between the original image and the edited image. This work is limited to color modification or the removal of certain parts of the shape, and it is impossible to make substantial modifications to the shape, such as shape deformation. A recent work, ObjectNeRF [74], proposed to learn a decompositional neural radiance field, which separates the objects and the background. As such, it can dupli-

†: Authors contributed equally
*Corresponding Author is Lin Gao (gaolin@ict.ac.cn)

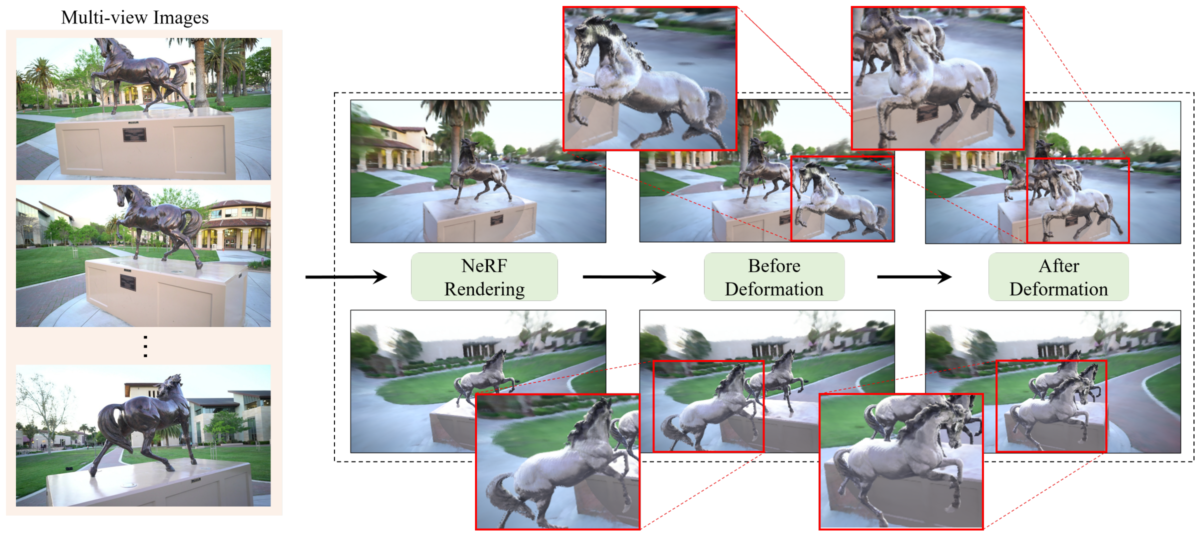

Figure 1. We propose a method to edit a static neural radiance field (NeRF). Users only need to capture multi-view images to build a NeRF representation, and then they can explicitly and intuitively edit the implicit representation of the scene. Our method can perform user-controlled shape deformation on the geometry of the scene, which contains multiple objects.

cate, move or rotate the objects for editable scene rendering. However, it does not support shape deformation either. Meanwhile, some works [48, 67] consider using NeRF to model dynamic scenes and using Multi-Layer Perceptron (MLP) to predict scene changes. However, they either limit the edits to human bodies [46, 82], or can only learn motion information from the recorded videos, and cannot perform active editing [48].

In this paper, we propose a method for editing neural radiance field that combines the advantage of explicit representations for easy local editing and the advantage of implicit representations for realistic rendering effects. Different from the previous work [38, 74], we focus on the geometric content of the scene, as shown in Fig. 1, supporting users to edit the scene geometry, and can perform photo-realistic rendering from novel views. As far as we know, we are the first to perform user-controlled shape deformation on the NeRF of general scenes. To this end, we first extract an explicit triangular mesh representation from the trained NeRF. The explicit mesh representation is then intuitively deformed by the user. Next, a tetrahedral mesh is built from the triangular mesh representation, which wraps around the triangular mesh. We use the deformation of the triangular mesh to drive the deformation of the tetrahedral mesh, which propagates the deformation of the scene geometric surface to the spatial discrete deformation field. Finally, we use tetrahedral vertex interpolation to complete the propagation from the discrete deformation field to the continuous deformation field. The rays passing through the tetrahedral mesh will be bent accordingly following the continuous deformation field, so that the final rendering result conforms to the user's edits. Our method is general, not limited to specific shapes such as human bodies, and applicable to arbitrary shapes such as animal models and general man-made objects.

2. Related Work

Our NeRF editing framework provides a new paradigm for novel view synthesis of an edited neural implicit scene representation. Here, we summarize related work of novel view synthesis and 3D deformation/editing methods.

Novel view synthesis. To infer the photo-realistic novel view synthesis result from given input images, prior works rely on explicit [6, 20, 21, 56] or implicit [19, 32, 62] geometry representation of the real world scene. Recently, used both as a component in deep neural network pipelines and as a standalone rendering pipeline, neural rendering has achieved immense progress, which is comprehensively summarized in [64, 65]. It adopts deep neural networks to synthesize images, which can be employed on multiple representations, such as voxels [39, 54], point clouds [1, 10], meshes [7,50,51,66], multi-plane images (MPIs) [34,42,87] and implicit fields [31, 55]. As one of the representative works, Neural Radiance Field (NeRF) [43] has attracted a lot of attention, which uses a multi-layer perceptron (MLP) to model the geometry and appearance of a scene. NeRF can achieve photo-realistic synthesis of novel view images with view-dependent effects. However, NeRF still has shortcomings and plenty of work has extended the original NeRF, including better synthesis effects [36, 71, 83], applicable to dynamic scenes [14, 33, 44–48, 67, 72, 82], faster rendering speed [18, 22, 49, 77], generalization to different scenes [8, 69], relighting [4, 60, 85], etc. NeRF-related works have been summarized in [11]. In this work, we focus on geometry editing/deformation for NeRF. As mentioned before, EditingNeRF [38] proposes to edit on the rendered image and uses network optimization to achieve editing transfer to the entire image and novel view images. However, the edits are limited to 2D images, which cannot change the spatial position of the object, let alone change the shape of the object. ObjectNeRF [74] has a decompositional network architecture, which can only duplicate, move or rotate objects. Our framework, however, support editing the geometric shape of the objects in NeRF, which can then be used to synthesize photo-realistic novel view images for visualization.

3D deformation and editing methods. Editing a 3D model means deforming the shape of the model under some controls given by the user. There has been much work about the editing of explicit geometry representation [9,17], which we refer readers to a recent survey [80]. Traditional mesh deformation methods are based on Laplacian coordinates [35, 57, 58], Poisson equation [79], and dual Laplacian coordinates [2]. As a representative work among them, ARAP (As-Rigid-As-Possible) deformation [59] is an interactive mesh editing scheme, which preserves details during the deformation by maintaining the rigidity of the local transformations. Another approach to driving mesh deformation is through a proxy, such as skeletons [27, 41] or cages [53, 76, 86]. These methods need to calculate the weights [13, 28, 81] between the proxy and the mesh vertices, and propagate the transformation of the proxy to the mesh. With the proliferation of geometric models [3], data-driven deformation [15, 16, 61] becomes available which analyzes the deformation prior of existing shapes in the dataset and produces more realistic results. At the same time, plenty of data also allows neural networks to be introduced into 3D editing [37, 63, 73, 75]. In addition to the explicit mesh representation, the implicit field can also be edited in combination with a neural network. Deng *et al.* proposed the deformed implicit field [12], which is capable of modeling dense surface correspondence and shape editing based on the learned information from an object category. Our work also aims to edit implicit representations, in particular NeRFs. The difference is that we take advantage of the intuitive and convenient characteristics of explicit mesh editing. By establishing a connection between the explicit mesh representation and implicit neural representation, well-developed mesh deformation methods are used to edit the geometry of implicit representation.

3. Our Method

Our work is based on the neural radiance field (NeRF) [43], which has promising performance in novel view synthesis. As a result, our method enables users to perform shape deformation on the content of the scene, and can generate new images from arbitrary views after editing. We will first briefly review NeRF pipeline (Sec. 3.1), and then introduce how to extract the explicit triangular mesh representation from the implicit representation of the scene and enable users to edit the mesh representation (Sec. 3.2). After the user edits the triangular mesh representation of the scene, we need to transfer this deformation to the implicit volume representation. We split the transfer into two steps. The first step is to transfer the surface mesh deformation to a volumetric mesh, where we build a tetrahedra mesh that wraps around the surface mesh, and transfer user edits on the surface mesh to discrete deformation fields on the tetrahedra mesh (Sec. 3.3). The next step is to transform the discrete deformation field to a continuous deformation field in the space volume, which is used to guide the bending of the rays to render images conforming to user edits (Sec. 3.4). We will later show that directly transferring the deformation from surface mesh to the implicit volume by interpolation will lead to obvious artifacts compared to our two step strategy in Sec. 4.3. Our method establishes the connection between the explicit mesh representation and the implicit radiance field, enabling users to modify the geometry of radiance field through intuitive edits. The pipeline is shown in Fig. 2.

3.1. Neural Radiance Fields

Neural Radiance Field or NeRF [43] proposes to use a multi-layer perceptron (MLP) network to model the geometry and appearance of the scene from a sparse set of images. Given the known camera parameters, the image pixels can be transformed to the world coordinate system and connected with the camera position to generate the light rays that are directed toward the scene. NeRF samples points on the ray and uses volume rendering [30] to obtain the color of each ray. The spatial coordinates $\mathbf{p} = (x, y, z)$ of each sampled point and the ray direction $\mathbf{d} = (\theta, \phi)$ will go through positional encoding $\zeta(\cdot)$, and then input into the fully connected network to predict the volume density σ and RGB value $\mathbf{c}$: $F_\Theta : (\zeta(\mathbf{p}), \zeta(\mathbf{d})) \rightarrow (\sigma, \mathbf{c})$, where Θ represents the network weights. The predicted density value σ can be interpreted as the differentiable probability of the ray terminated at the sampled point, and the color $\hat{C}(\mathbf{r})$ of the image pixel corresponding to the ray $\mathbf{r}(t)$ can be calculated through discrete integration:

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)(1 - \exp(-\sigma_i \delta_i))\mathbf{c}_i, \quad (1)$$

where $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples. The network is supervised by the RGB loss function, which is calculated between the generated color $\hat{C}(\mathbf{r})$ and the ground truth color $C(\mathbf{r})$ of the ray.

3.2. Editing of Explicit Surface Mesh Representation

After the NeRF network is trained, an explicit triangular mesh representation can be extracted directly from the neu-

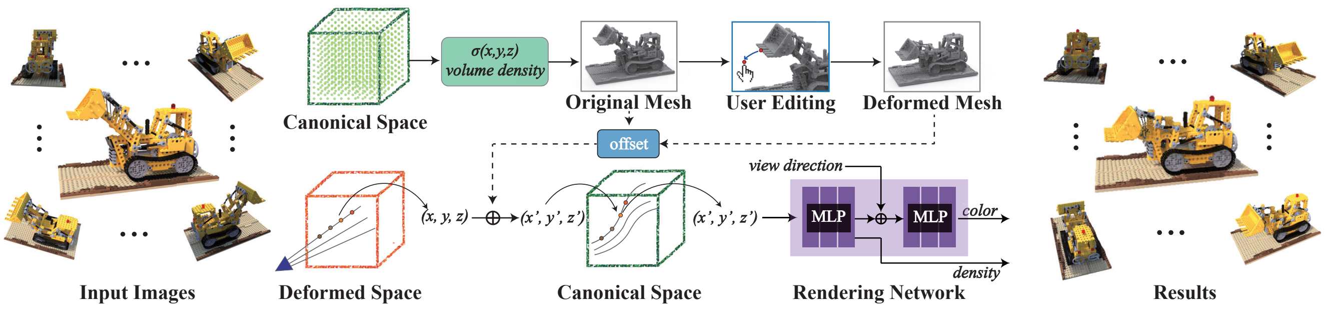

Figure 2. The pipeline of our NeRF editing framework. The user edits the reconstructed mesh and a continuous deformation field is built to bend the rays accordingly.

ral radiance fields using Marching Cubes [40]. However, the mesh extracted from the original NeRF network is often with rough surface. In order to obtain a satisfactory editing representation, we adopt the reconstruction method proposed in NeuS [68], which takes a bias-free volume rendering manner to learn the geometry as a neural signed distance function (SDF) representation. The mesh representation extracted from the zero-level set of SDF will serve as the user's editing object, allowing users to edit the scene content intuitively. In this paper, we use the classic ARAP (as-rigid-as-possible) deformation method [59] to enable users to interactively deform the mesh. It should be noted that any other mesh deformation method can be used here, including skeleton-based and cage-based methods.

The extracted triangular mesh is denoted as S, and $N(i)$ represents the index set of vertices adjacent to vertex i. We further denote $\boldsymbol{v}_i \in \mathbb{R}^3$ as the position of the vertex i on the mesh S. After the user's edits, the mesh S is transformed to the deformed mesh S' with the same connectivity and different vertex positions $\boldsymbol{v}'_i$, treating user editing as boundary conditions. The overall ARAP deformation energy is to measure the rigidity of the entire mesh and is the sum of the distortion energies of each deformation cell, including vertex i and its 1-ring neighbors, shown in Eq. 2.

$$E(S') = \sum_{i=1}^{n} \sum_{j \in N(i)} w_{ij} \left\| (\boldsymbol{v}'_i - \boldsymbol{v}'_j) - \mathbf{R}_i(\boldsymbol{v}_i - \boldsymbol{v}_j) \right\|^2. \quad (2)$$

Here, $w_{ij} = \frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij})$ is the cotangent weight, and α_{ij}, β_{ij} are the angles opposite to the mesh edge (i, j). $\mathbf{R}_i$ is the local rotation at vertex i. The deformed shape S' is obtained by minimizing the ARAP energy, which can be efficiently solved by alternately optimizing local rotations $\mathbf{R}_i$ and deformed positions $\boldsymbol{v}'_i$. We refer the readers to [59] for the specific optimization process.

3.3. Deformation Transfer to Discrete Volume

After the user edits the triangular mesh representation of the scene, the deformation needs to be transferred to the implicit volume representation. As introduced before, we split the transfer into two steps. In the first step, we build

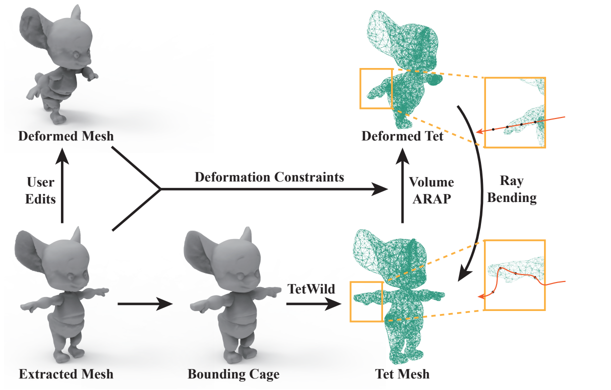

Figure 3. We use discrete deformations specified by users to bend the rays.

a tetrahedral mesh (a discrete volumetric representation) to cover the extracted triangular mesh. Starting from the extracted triangular mesh S, we first calculate a cage mesh that wraps the mesh S. This can be achieved by enlarging the triangular mesh by making a certain distance offset from the mesh surface in the normal direction. We set the default value to 5% of the averaged distance from the camera position to object center. The internal space of the cage mesh can be regarded as the "effective space" of the implicit volume, because the area near real geometry surface of the scene is enclosed by this cage mesh. When editing larger scenes with multiple objects, this design also ensures other objects not being edited are not affected. We use the tetrahedralization method, TetWild [25], to tetrahedronize the cage mesh to obtain a tetrahedral mesh representation T. It should be noted that the extracted triangular mesh S is also wrapped in the tetrahedral mesh T. We visualize some extracted triangular mesh S and the corresponding tetrahedral mesh T in the supplementary material. We use the displacement of the triangular mesh vertices $\boldsymbol{v}_i$ to drive the deformation of the tetrahedral mesh T, which transfers the surface deformation to the tetrahedral mesh. The deformed tetrahedral mesh is denoted as T', and $\boldsymbol{t}_k$ and $\boldsymbol{t}'_k$ denote the vertices of the tetrahedral mesh before and after deformation respectively, where k is the vertex index. Here, we also use the ARAP deformation method to deform the tetrahedral mesh T under the constraints of the surface mesh deformation. Eq. 2 can be extended from the triangular mesh to the tetrahedral mesh straightforwardly. The only difference is that the constraints are changed from user-specified control points to the triangular mesh vertices. We can find which tetrahedron each triangular mesh vertex is located in, and calculate its barycentric coordinates relative to the four vertices of the tetrahedron. Then, the optimization problem is,

$$\min E(T'), \text{ subject to } \boldsymbol{A}\boldsymbol{t}' = \boldsymbol{v}', \tag{3}$$

where $\boldsymbol{A}$ is the barycentric weight matrix. This optimization problem can be converted into linear equations using the Lagrangian multiplier method. Please refer to the supplementary material for the specific derivation.

3.4. Ray Bending

After transferring the surface deformation to the tetrahedral mesh, we can obtain the discrete deformation field of the "effective space". We now utilize these discrete transformations to bend the casting rays. To generate an image of the deformed radiance field, we cast rays to the space containing the deformed tetrahedral mesh. For each sampled point on the ray, we find which tetrahedron of the deformed tetrahedral mesh T' it is located in. Using the correspondence between T and T', the displacement from the vertices after deformation to the vertices before deformation can be obtained. Through barycentric interpolation of the displacements of the four vertices of the tetrahedron where the sampled point is located, the displacement of the sampled point back to the original "effective space" Δp can be obtained. We add the displacement Δp to the input coordinate of the sampled point to predict the density and RGB values.

$$(\zeta(\mathbf{p} + \Delta\mathbf{p}), \zeta(\mathbf{d})) \rightarrow (\sigma, \mathbf{c}). \tag{4}$$

The density and RGB values of the sampled points along the ray are used to calculate the corresponding pixel color using Eq. 1. It should be noted that the sampled points that are not within the tetrahedral mesh T' will not be moved, i.e., the part of the ray outside the tetrahedral mesh will not be bent. The process of building deformation field is illustrated in Fig. 3.

4. Experiments and Evaluations

In this section, we conduct several qualitative and quantitative experiments, including showing editing results on both synthetic data and captured real scenes, comparisons with baseline methods, and ablation study.

4.1. Datasets and metrics

We demonstrate our method on several public synthetic data, including some characters in the mixamo [26], the Lego bulldozer and chair from NeRF [43]. Moreover, we also test our method on a real captured horse statue from

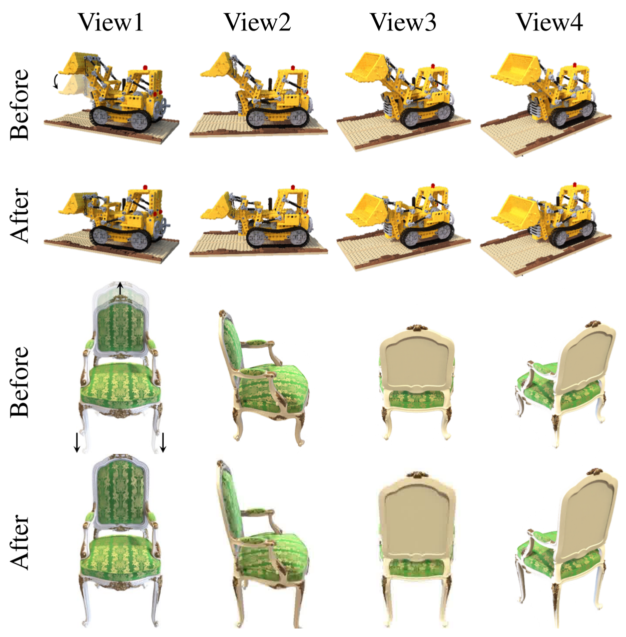

Figure 4. We show the editing results (row "After") compared with NeRF rendering results (row "Before") on synthetic data under different views, including a Lego bulldozer and a chair, which illustrate that our method can edit the NeRF of general models. Different columns show different views.

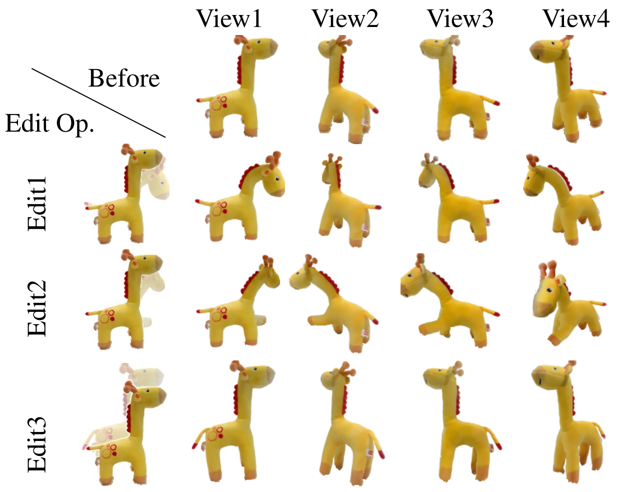

Figure 5. Results of our NeRF editing framework (rows "Edit1", "Edit2", "Edit3") compared with original NeRF results (row "Before") on a real captured giraffe soft toy. Different columns show different views. We can edit the object into different poses and render it under different views. "Edit Op." denotes " Edit Operation".

FVS dataset [50] and several real scenes captured by ourselves. The characters in the mixamo are rendered by ourselves. We generate 100 random views from the upper hemisphere with Blender for training. For the data from NeRF datasets, we use the default training setting of the datasets. For the real scenes captured by ourselves, we leave one image for validation, and the other images are used for training. More information about self-captured dataset is included in supplementary document.

It needs to be noted that different from dynamic NeRF methods [48], it is difficult to obtain the ground truths of the novel view synthesis results after user editing, especially on real scenes, as such edited scenes do not physically exist. So we mainly evaluate our approach quantitatively and qualitatively on the characters in the mixamo. Specifically, we rig the mixamo character model, render the deformed characters as the ground truths and compare them with the outputs of our NeRF editing method. We use Structural Similarity Index Measure (SSIM) [70], Learned Perceptual Image Patch Similarity (LPIPS) [84] and Peak Signal-to-Noise Ratio (PSNR) as the metrics to evaluate the performance of our approach. We also evaluate the Fréchet Inception Distance [23] (FID) score on real scenes to measure the similarity between the results before editing and after editing, since the ground truths are not indispensable.

4.2. Editing Results

Shape editing results under different views. We first show NeRF editing results rendered from different views in Figs. 4- 6 for synthetic data and real captured objects. For comparison, we also show the results under the same views before editing. In Fig. 4, the first set is a Lego bulldozer from the NeRF dataset. We put down its shovel and achieve the editing of complex synthetic data. The second set is a synthetic chair from the NeRF dataset. We stretch the back and legs of the chair, which demonstrates that our method can edit the local parts of man-made objects. In Fig. 5, we present the editing results on a giraffe soft toy captured by ourselves. It can be seen that users can edit the giraffe to different poses, as well as scale local areas, which demonstrates the usability of our method. In Fig. 6, we show four more sets of results from real scenes to illustrate that our method can be applied to different objects. The wings of the toy dragon are deformed to make them spread out. This can further realize the animation of the dragon flapping its wings while viewing it from different directions. We also show an example of a horse statue from the FVS dataset, where we can deform the horse's head and raise its hoof. On the example of a laptop, we can rotate its panel to be at different angles. For the real captured chair, we bend the legs of the chair to present another design style, and at the same time enlarge the backrest, which make the chair more comfortable to sit on. These results show that our approach is able to deform static neural radiance fields according to the user's editing. In Fig. 1, we show an example of shape deformation for multiple objects in a scene. We first split the mesh of the horse statue from the scene, then copy it into multiple ones, place them in different locations, and deform them differently.

Deformation transfer results. In addition to user-controlled shape deformation, we can also use deformation transfer methods to transfer the deformations from the existing deformation sequence to the real captured objects. This can achieve some interesting applications. For example, we can transfer the movements of a human face from a video clip to a head sculpture. We show some example results in supplementary material.

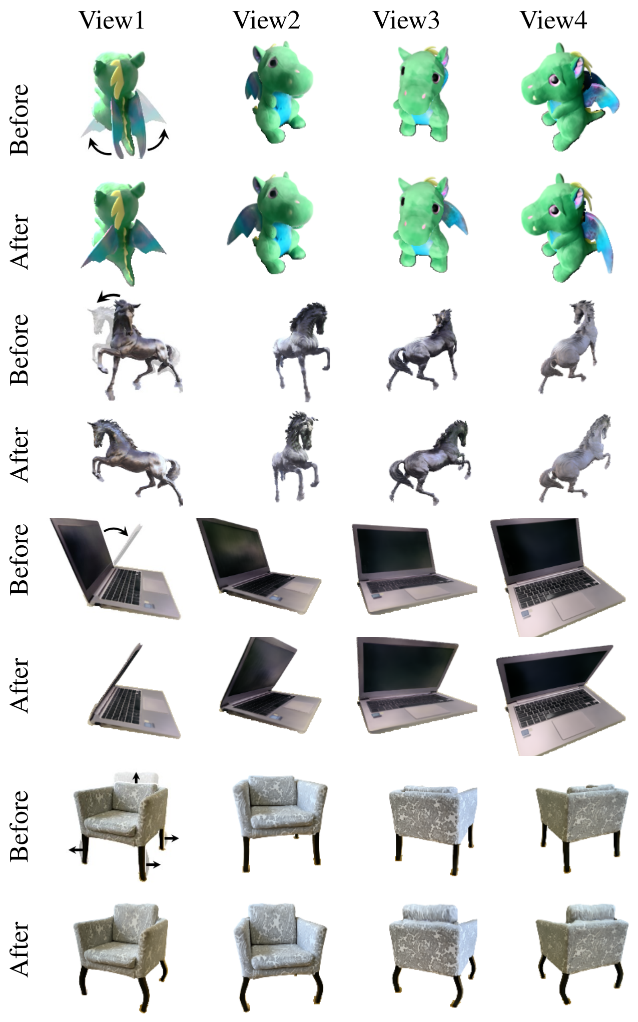

Figure 6. Results of our NeRF editing framework (row "After") compared with original NeRF results (row "Before") on multiple real captured data. Different columns show different views. We edit the static neural radiance fields and exhibit the deformed results under different views.

4.3. Comparisons

As we are the first to perform general geometric shape deformation on NeRF, we propose three baseline methods for comparisons. For the first baseline, we adopt a naive way to build the correspondence between the extracted triangular mesh and continuous volume space. We no longer construct a tetrahedral mesh and use it as a proxy, but instead directly find the closest point of the sampled point on

Method	SSIM ↑	LPIPS ↓	PSNR ↑	FID (real) ↓
Closest Point	0.928	0.055	22.38	300.8
3NN	0.941	0.042	24.30	291.7
Ours	**0.975**	**0.024**	**29.62**	**253.7**

Table 1. Quantitative comparison with the "Closest Point" and "3NN" baselines. It can be seen that our framework achieves better results. Note that the first three metrics are calculated on mixamo synthetic data, while the last FID is calculated on real data.

the extracted triangular mesh surface, and use the displacement of the closest point as the displacement of the sampled point. We denote the first one as "Closest Point". The second baseline is similar to the first one. The difference is that we linearly interpolate the displacements of three nearest triangular mesh vertices, with the coefficients inversely related to distances, to obtain the displacement of the sampled point. We denote this one as "3NN". The last baseline is mesh rendering. The extracted triangular mesh is with vertex color information, which can be directly rendered after user-controlled shape deformation or deformation transfer.

We compare our method with the "Closest Point" and "3NN" baselines on the synthetic data mixamo which has ground truth edited results. The visual comparison results are shown in Fig. 7, and the quantitative comparison is shown in Table 1. We also show a visual comparison on a real captured scene in the last row of Fig. 7. Due to the absence of ground truth, we visualize the NeRF rendering result before deformation and corresponding deformation results. It can be seen that the "Closest Point" and "3NN" baselines may cause discontinuities, so the rendering results have obvious artifacts, while our method adopts two-step deformation transfer, and the results are more visually satisfactory and has quantitative advantages.

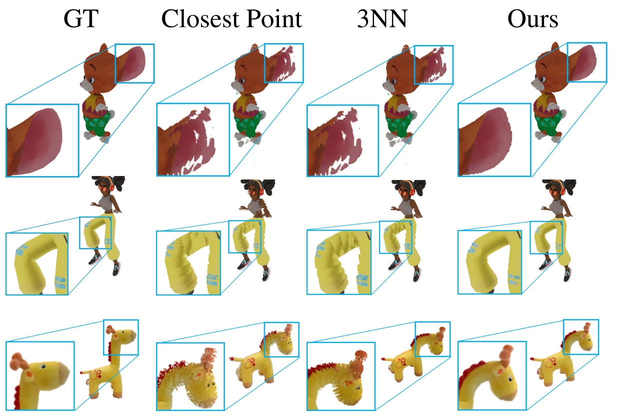

Figure 7. Visual comparisons with the "Closest Point" and "3NN" baselines. The "Closest Point" and "3NN" baselines may cause discontinuities, so the resulting rendering results have obvious artifacts. Note that the last row is a real captured toy giraffe, so ground truth does not exist and we instead visualize the NeRF rendering results before deformation for reference.

Then we compare our method with mesh rendering baseline on the Lego data from NeRF. It should be noted that although our method uses an explicit triangular mesh representation for interactive editing, it has a certain degree of error tolerance in terms of the mesh reconstruction, and the reconstructed triangular mesh does not need to be perfect. This is because the mesh is only used as an intermediate representation and our final images are still obtained through volume rendering. The direct mesh rendering requires a high quality mesh, and all artifacts on the mesh will appear in the rendered images. As shown in Fig. 8, the reconstructed mesh in the lego is not of good quality, and the result of mesh rendering is not ideal, while our method can still perform editing, and with the help of volume rendering, desired results can still be obtained.

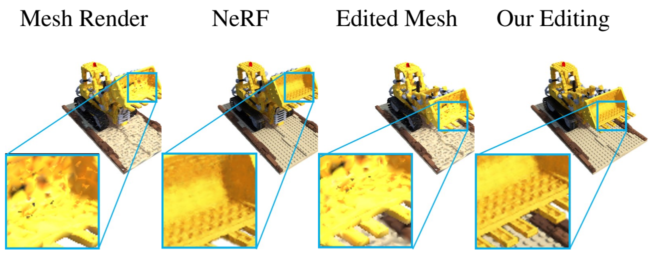

Figure 8. Comparisons with mesh rendering on the synthetic data. Mesh rendering has obvious artifacts when the mesh quality is not good, while this does not affect our editing and image synthesis.

4.4. Ablation Study

We conduct ablation studies on the synthetic data with respect to the novel view synthesis results after editing. First, as we introduce a tetrahedral mesh in our method as a proxy between the triangular mesh and the continuous volume, we compare the results of editing on the triangular mesh and editing on the tetrahedral mesh, and verify the necessity of editing on the triangular mesh and transferring deformation by our method. Second, in order to evaluate the influence of the reconstructed triangular mesh on our results, we compare the results of the triangular mesh extracted by the original NeRF and that extracted by NeuS which improves the quality of reconstruction. Tables 2 and 3 summarize the quantitative results of the ablation studies.

Necessity of edit on triangular mesh. Table 2 shows the quantitative comparisons between editing on the tetrahedral mesh and triangular mesh, which indicates that editing on triangular mesh performs better. The qualitative results are presented in Fig. 9. The results of editing on tetrahedral mesh have obvious artifacts due to poor tetrahedral mesh quality.

Impact of mesh quality. Table 3 evaluates the influence of the reconstructed mesh quality to our method. It can be seen that the result of using the mesh from NeuS is better than that of NeRF, but the difference is small. The visual comparisons are shown in Fig. 10, where the results of using the mesh from NeRF have some artifacts in detail, but the

Method	SSIM ↑	LPIPS ↓	PSNR ↑
Edit on tetrahedral mesh	0.934	0.049	24.37
Edit on triangular mesh	**0.975**	**0.024**	**29.62**

Table 2. Evaluation on the necessity of editing on the triangular mesh. Editing on the triangular mesh leads to better results than directly editing on the tetrahedral mesh.

overall result is not bad. This illustrates that mesh quality has little effect on our results.

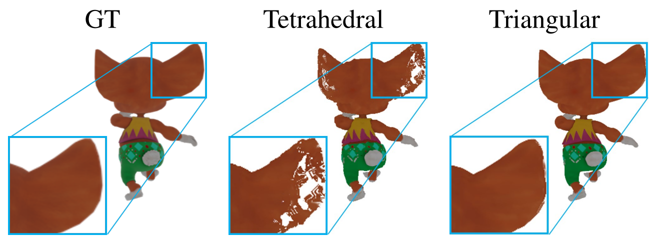

Figure 9. Ablation study of editing on the tetrahedral mesh or triangular mesh. It can be seen that editing on the tetrahedral mesh will bring in artifacts in rendered results.

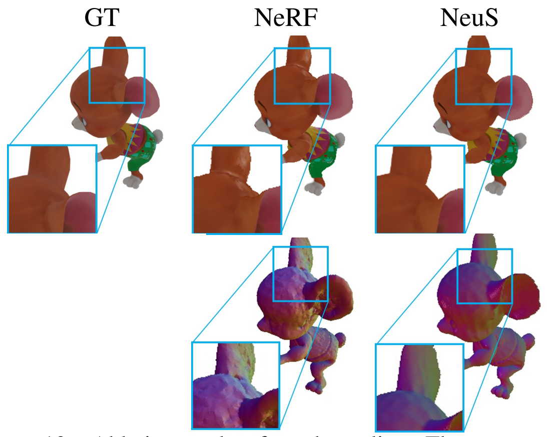

Figure 10. Ablation study of mesh quality. The reconstructed mesh from NeRF is worse than that of NeuS, leading to some artifacts in the rendered results. We visualize the rendered results (first row) and the mesh colored with vertex normals (second row).

Method	SSIM ↑	LPIPS ↓	PSNR ↑
NeRF	0.969	0.027	28.95
NeuS	**0.975**	**0.024**	**29.62**

Table 3. Evaluation on the impact of the extracted mesh quality. The reconstructed mesh from NeuS is better than that from NeRF, leading to better editing results.

4.5. Limitations

Our method is the first step for geometric shape deformation on NeRFs and still has several limitations. First of all, the biggest limitation is that we cannot modify the color and also the light and shadow based on the editing results. If an object part that is in the shadow during capturing is deformed to face the light, its color will still be dim instead of bright, as shown in Fig. 11. This could be dealt with by incorporating some recent NeRF-based relighting work [4, 85] to achieve correct color rendering by decoupling lighting. Second, our method cannot support real-time editing by users. The user can only select a viewing angle for image synthesis after editing the mesh representation. At present, the main time bottleneck is still in the rendering part of NeRF. Recently, there are some works on the acceleration of NeRF rendering [18, 22, 77]. The combination with these methods will help with real-time rendering of interactive editing results.

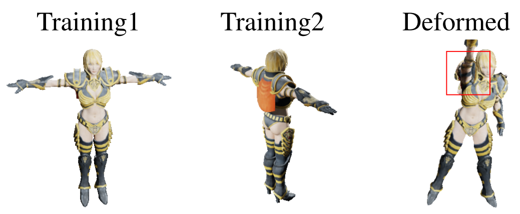

Figure 11. Failure case. Our approach does not edit the appearance along with geometry deformation. In this example, since the underarm is occluded from the light in the T-pose during training, it will always be gloomy even when the woman raises her arm, which is implausible.

5. Conclusion

In this paper, we propose the first method to support user-controlled shape deformation to the geometry of neural radiance field network. By establishing a correspondence between the explicit mesh representation and the implicit volume representation, our method can use the well-developed triangular mesh deformation method to deform the implicit representation. With the novel view synthesis capability of NeRF, users can visualize the editing results from arbitrary views. Our method is suitable for general real scenes which can edit scene objects including human bodies, animals, man-made models, etc. Compared with the previous editing methods for NeRF, our method has a higher degree of freedom and can support the editing of details. In the future, we will further explore the combination of relighting methods. After editing the scene geometry, the corresponding colors can be modified to make the light and shadow in the rendering results more natural. In future work, we will implement our proposed approach in Jittor [24], which is a fully just-in-time (JIT) compiled deep learning framework.

Acknowledgement

This work was supported by the Beijing Municipal Natural Science Foundation for Distinguished Young Scholars (No. JQ21013), the National Natural Science Foundation of China (No. 62061136007 and No. 61872440), Royal Society Newton Advanced Fellowship (No. NAF\R2\192151), the Alibaba Innovative Research (AIR) Program and the Youth Innovation Promotion Association CAS.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *European Conference on Computer Vision*, pages 696–712. Springer, 2020. 2

[2] OK-C Au, Chiew-Lan Tai, Ligang Liu, and Hongbo Fu. Dual Laplacian editing for meshes. *IEEE Transactions on Visualization and Computer Graphics*, 12(3):386–395, 2006. 3

[3] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2014. 3

[4] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. NeRD: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021. 1, 2, 8

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1

[6] Gaurav Chaurasia, S. Duchêne, O. Sorkine-Hornung, and G. Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32:30:1–30:12, 2013. 2

[7] Anpei Chen, Minye Wu, Yingliang Zhang, Nianyi Li, Jie Lu, Shenghua Gao, and Jingyi Yu. Deep surface light fields. In *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, volume 1, pages 1–17. ACM New York, NY, USA, 2018. 2

[8] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021. 2

[9] Shu-Yu Chen, Lin Gao, Yu-Kun Lai, and Shihong Xia. Rigidity controllable as-rigid-as-possible shape deformation. *Graphical Models*, 91:13–21, 2017. 3

[10] Peng Dai, Yinda Zhang, Zhuwen Li, Shuaicheng Liu, and Bing Zeng. Neural point cloud rendering via multi-plane projection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7830–7839, 2020. 2

[11] Frank Dellaert and Lin Yen-Chen. Neural volume rendering: Nerf and beyond, 2021. 2

[12] Yu Deng, Jiaolong Yang, and Xin Tong. Deformed implicit field: Modeling 3D shapes with learned dense correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 3

[13] Michael S Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20(1):19–27, 2003. 3

[14] Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4D facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8649–8658, 2021. 2

[15] Lin Gao, Yu-Kun Lai, Dun Liang, Shu-Yu Chen, and Shihong Xia. Efficient and flexible deformation representation for data-driven surface modeling. *ACM Transactions on Graphics (TOG)*, 35(5):158:1–158:17, 2016. 3

[16] Lin Gao, Yu-Kun Lai, Jie Yang, Zhang Ling-Xiao, Shihong Xia, and Leif Kobbelt. Sparse data driven mesh deformation. *IEEE Transactions on Visualization and Computer Graphics*, 27(3):2085–2100, 2019. 3

[17] Lin Gao, GuoXin Zhang, and YuKun Lai. L p shape deformation. *Science China Information Sciences*, 55(5):983–993, 2012. 3

[18] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021. 1, 2, 8

[19] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen. The lumigraph. *Proceedings of the 23rd annual conference on Computer Graphics and Interactive Techniques*, 1996. 2

[20] Peter Hedman, Suhib Alsisan, R. Szeliski, and J. Kopf. Casual 3D photography. *ACM Transactions on Graphics (TOG)*, 36:1 – 15, 2017. 2

[21] Peter Hedman, Tobias Ritschel, G. Drettakis, and G. Brostow. Scalable inside-out image-based rendering. *ACM Transactions on Graphics (TOG)*, 35:1 – 11, 2016. 2

[22] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5875–5884, 2021. 2, 8

[23] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017. 6

[24] Shi-Min Hu, Dun Liang, Guo-Ye Yang, Guo-Wei Yang, and Wen-Yang Zhou. Jittor: a novel deep learning framework with meta-operators and unified graph execution. *Science China Information Sciences*, 63(222103):1–21, 2020. 8

[25] Yixin Hu, Teseo Schneider, Bolun Wang, Denis Zorin, and Daniele Panozzo. Fast tetrahedral meshing in the wild. *ACM Transactions on Graphics (TOG)*, 39:117:1 – 117:18, 2020. 4

[26] Adobe Inc. Mixamo. `https://www.mixamo.com`. 5

[27] Alec Jacobson, Ilya Baran, Ladislav Kavan, Jovan Popović, and Olga Sorkine. Fast automatic skinning transformations. *ACM Transactions on Graphics (TOG)*, 31(4):1–10, 2012. 3

[28] Alec Jacobson, Ilya Baran, Jovan Popovic, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Transactions on Graphics (TOG)*, 30(4):78–1, 2011. 3

[29] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting NeRF on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021. 1

[30] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH Computer Graphics*, 18(3):165–174, 1984. 3

[31] Petr Kellnhofer, Lars C Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4287–4297, 2021. 2

[32] M. Levoy and P. Hanrahan. Light field rendering. *Proceedings of the 23rd Annual Conference on Computer Graphics and interactive techniques*, 1996. 2

[33] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 2

[34] Zhengqi Li, Wenqi Xian, Abe Davis, and Noah Snavely. Crowdsampling the plenoptic function. In *European Conference on Computer Vision*, pages 178–196. Springer, 2020. 2

[35] Yaron Lipman, Olga Sorkine, Marc Alexa, Daniel Cohen-Or, David Levin, Christian Rössl, and Hans-Peter Seidel. Laplacian framework for interactive mesh editing. *International Journal of Shape Modeling*, 11(01):43–61, 2005. 3

[36] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33, 2020. 1, 2

[37] Lijuan Liu, Youyi Zheng, Di Tang, Yi Yuan, Changjie Fan, and Kun Zhou. NeuroSkinning: Automatic skin binding for production characters with deep graph networks. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 3

[38] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5773–5783, 2021. 1, 2

[39] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2

[40] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Computer Graphics*, 21(4):163–169, 1987. 4

[41] Nadia Magnenat-Thalmann, Richard Laperrire, and Daniel Thalmann. Joint-dependent local deformations for hand animation and object grasping. In *In Proceedings on Graphics interface'88*. Citeseer, 1988. 3

[42] 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. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2

[43] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pages 405–421. Springer, 2020. 1, 2, 3, 5

[44] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2

[45] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. HyperNeRF: a higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics (TOG)*, 40(6):1–12, 2021. 2

[46] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14314–14323, 2021. 2

[47] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9054–9063, 2021. 2

[48] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1, 2, 6

[49] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny MLPs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021. 2

[50] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, pages 623–640. Springer, 2020. 2, 5

[51] Gernot Riegler and Vladlen Koltun. Stable view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[52] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision*, 2016. 1

[53] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. *SIGGRAPH Comput. Graph.*, 20(4):151–160, 1986. 3

[54] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. DeepVoxels: Learning persistent 3D feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019. 2

[55] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: continuous 3D-structure-aware neural scene representations. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 1121–1132, 2019. 2

[56] Noah Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. In *SIGGRAPH 2006*, 2006. 2

[57] Olga Sorkine. Laplacian mesh processing. *Eurographics (STARs)*, 29, 2005. 3

[58] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and H-P Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 175–184, 2004. 3

[59] Olga Sorkine-Hornung and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry Processing*, 2007. 3, 4

[60] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7495–7504, 2021. 1, 2

[61] Robert W Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. Mesh-based inverse kinematics. *ACM Transactions on Graphics (TOG)*, 24(3):488–495, 2005. 3

[62] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 517–524. IEEE, 1998. 2

[63] Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. Variational autoencoders for deforming 3D mesh models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5841–5850, 2018. 3

[64] Ayush Tewari, Ohad Fried, Justus Thies, V. Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason M. Saragih, Matthias Nießner, R. Pandey, S. Fanello, G. Wetzstein, Jun-Yan Zhu, C. Theobalt, Maneesh Agrawala, E. Shechtman, Dan B. Goldman, and Michael Zollhofer. State of the art on neural rendering. *Computer Graphics Forum*, 39, 2020. 2

[65] Anju Tewari, Otto Fried, Justus Thies, Vincent Sitzmann, S. Lombardi, Z Xu, Tanaba Simon, Matthias Nießner, Edgar Tretschk, L. Liu, Ben Mildenhall, Pranatharthi Srinivasan, R. Pandey, Sergio Orts-Escolano, S. Fanello, M. Guo, Gordon Wetzstein, J y Zhu, Christian Theobalt, Manju Agrawala, Donald B. Goldman, and Michael Zollhöfer. Advances in neural rendering. *ACM SIGGRAPH 2021 Courses*, 2021. 2

[66] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 2

[67] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhofer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 2

[68] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems*, volume 34, 2021. 4

[69] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021. 2

[70] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6

[71] Suttisak Wizadwongsa, Pakkapon Phongthawee, Jiraphon Yenphraphai, and Supasorn Suwajanakorn. NeX: Real-time view synthesis with neural basis expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8534–8543, 2021. 2

[72] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9421–9431, 2021. 2

[73] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. RigNet: Neural rigging for articulated characters. *ACM Transactions on Graphics (TOG)*, 39(4):1–14, 2020. 3

[74] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13779–13788, October 2021. 1, 2, 3

[75] Jie Yang, Lin Gao, Qingyang Tan, Yi-Hua Huang, Shihong Xia, and Yu-Kun Lai. Multiscale mesh deformation component analysis with attention-based autoencoders. *IEEE Transactions on Visualization and Computer Graphics*, 2021. 3

[76] Wang Yifan, Noam Aigerman, Vladimir G. Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3D deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 75–83, 2020. 3

[77] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021. 1, 2, 8

[78] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 1

[79] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with Poisson-based gradient field manipulation. In *ACM SIGGRAPH 2004 Papers*, pages 644–651. 2004. 3

[80] Yu-Jie Yuan, Yu-Kun Lai, Tong Wu, Lin Gao, and Ligang Liu. A revisit of shape editing techniques: From the geometric to the neural viewpoint. *Journal of Computer Science and Technology*, 36(3):520–554, 2021. 1, 3

[81] Yu-Jie Yuan, Yu-Kun Lai, Tong Wu, Shihong Xia, and Lin Gao. Data-driven weight optimization for real-time mesh deformation. *Graphical Models*, 104:101037, 2019. 3

[82] Jiakai Zhang, Xinhang Liu, Xinyi Ye, Fuqiang Zhao, Yanshun Zhang, Minye Wu, Yingliang Zhang, Lan Xu, and Jingyi Yu. Editable free-viewpoint video using a layered neural representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–18, 2021. 2

[83] K. Zhang, G. Riegler, Noah Snavely, and V. Koltun. NeRF++: Analyzing and improving neural radiance fields. *ArXiv*, abs/2010.07492, 2020. 2

[84] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 6

[85] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (TOG)*, 40(6):1–18, 2021. 2, 8

[86] Yuzhe Zhang, Jianmin Zheng, and Yiyu Cai. Proxy-driven free-form deformation by topology-adjustable control lattice. *Computers & Graphics*, 89:167–177, 2020. 3

[87] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018. 2