# DN-4DGS: Denoised Deformable Network with Temporal-Spatial Aggregation for Dynamic Scene Rendering

**Jiahao Lu**[1] **Jiacheng Deng**[1] **Ruijie Zhu**[1] **Yanzhe Liang**[1]
**Wenfei Yang**[1,2] **Tianzhu Zhang**[1*] **Xu Zhou**[3]

[1]University of Science and Technology of China/Deep Space Exploration Lab,
[2]Jianghuai Advance Technology Center, [3]Sangfor Technologies Inc.
{lujiahao, dengjc, ruijiezhu, yzliang}@mail.ustc.edu.cn,
{yangwf, tzzhang}@ustc.edu.cn, zhouxu@sangfor.com.cn

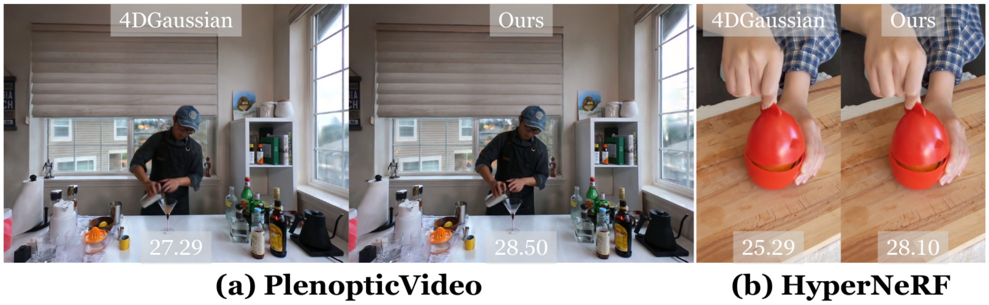

Figure 1: (a) The visualization results on PlenopticVideo [1] dataset. (b) The visualization results on HyperNeRF [2] dataset. The numbers below the images represent PSNR.

## Abstract

Dynamic scenes rendering is an intriguing yet challenging problem. Although current methods based on NeRF have achieved satisfactory performance, they still can not reach real-time levels. Recently, 3D Gaussian Splatting (3DGS) has garnered researchers' attention due to their outstanding rendering quality and real-time speed. Therefore, a new paradigm has been proposed: defining a canonical 3D gaussians and deforming it to individual frames in deformable fields. However, since the coordinates of canonical 3D gaussians are filled with noise, which can transfer noise into the deformable fields, and there is currently no method that adequately considers the aggregation of 4D information. Therefore, we propose Denoised Deformable Network with Temporal-Spatial Aggregation for Dynamic Scene Rendering (DN-4DGS). Specifically, a Noise Suppression Strategy is introduced to change the distribution of the coordinates of the canonical 3D gaussians and suppress noise. Additionally, a Decoupled Temporal-Spatial Aggregation Module is designed to aggregate information from adjacent points and frames. Extensive experiments on various real-world datasets demonstrate that our method achieves state-of-the-art rendering quality under a real-time level. Code is available at https://github.com/peoplelu/DN-4DGS.

[*]Corresponding author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

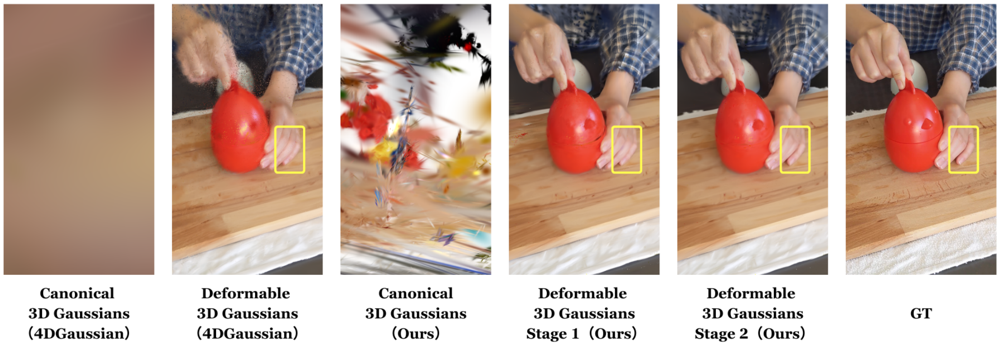
Canonical 3D Gaussians (4DGaussian)
Deformable 3D Gaussians (4DGaussian)
Canonical 3D Gaussians (Ours)
Deformable 3D Gaussians Stage 1 (Ours)
Deformable 3D Gaussians Stage 2 (Ours)
GT

Figure 2: **Comparison of our render visualization with 4DGaussian [11].** The results are rendered on HyperNeRF [2] dataset and use the point cloud provided by HyperNeRF for Gaussian initialization ($Sparse\ Init$). Image 1: canonical 3D gaussians generated by 4DGaussian. Image 2: deformable 3D gaussians generated by 4DGaussian. Image 3: canonical 3D gaussians generated by our method. Image 4: deformable 3D gaussians after the first stage. Image 5: deformable 3D gaussians after the second stage. Image 6: ground truth. The yellow box emphasizes that through a two-stage deformation process, our method can produce higher-quality rendering results.

## 1 Introduction

Dynamic scene reconstruction from single or multi-view videos is a crucial task in computer vision, with applications such as VR/AR [3, 4], 3D perception [5, 6, 7], movie production [8], etc. Neural Radiance Fields (NeRF) [9] offer a promising approach by representing scenes with implicit functions derived from multi-view inputs. By incorporating time as an additional input [10, 1], NeRF enables dynamic scene rendering. However, the original NeRF model suffers from significant training and rendering costs, attributed to the high number of points sampled per camera ray and volume rendering.

Recently, the emerging 3D Gaussian Splatting (3DGS) [12] has significantly increased rendering speed to a real-time level compared to NeRFs by employing a differentiable rasterizer for 3D Gaussian primitives. 3DGS directly optimizes the parameters of 3D gaussians (position, opacity, anisotropic covariance, and spherical harmonics (SH) coefficients) and renders them through projection and $\alpha$-blending. Given the explicit expression nature of 3DGS, recent studies [11, 13, 14] represent dynamic scenes by defining a canonical 3D gaussians and deforming it to individual frames in deformable fields. Specifically, during the execution of the deformation field, the coordinates $xyz$ along with time $t$ are used as input, and the output corresponds to the changes in Gaussian properties. It is noteworthy that the existing methods, when designing deformable networks, either directly map the 4D coordinates of each input point to the latent space using MLPs [13, 14], or use HexPlane [15] to interpolate a series of learnable embeddings to obtain the latent of each point [11].

Both of these approaches have drawbacks that can not be ignored. 1) Canonical 3D gaussians are synthesized from multi-frame images of dynamic scenes. Due to the presence of dynamic regions and the specific design of A (canonical 3D gaussians) + B (deformable network), canonical 3D gaussians exhibit significant noise, as illustrated in Figure 2. This noise is inevitably transferred to the deformable field after the input $xyz$ is passed through the deformable network. To elaborate on the "Noise": In the canonical + deformable design, we input the canonical Gaussian coordinates $xyz$ and time $t$ into the deformable network. The deformable network essentially performs basic operations (addition, subtraction, multiplication, division) on the coordinates $xyz$ and time $t$. Since the point-to-point relationships within the canonical Gaussians are chaotic and erroneous, as shown in Figure 2, it is predictable that feeding these erroneous coordinates into the deformable network will transfer this error into the deformation field, introducing inaccuracies in the final deformations $\Delta x, \Delta y, \Delta z$. 2) There is a lack of feature aggregation for spatial-temporal information, yet due to the presence of noise in canonical 3D gaussians' $xyz$, direct feature aggregation for spatial information would further amplify noise, affecting the learning of the deformable field. Therefore, spatial aggregation after denoising is very crucial.

To address the aforementioned issues, we propose Denoised Deformable Network with Temporal-Spatial Aggregation for Dynamic Scene Rendering (DN-4DGS), primarily consisting of two components: the Noise Suppression Strategy (NSS) and the Decoupled Temporal-Spatial Aggregation Module (DTS). To address the initial issue, the design of NSS incorporates two deformation operations. The first deformation operation is a standard deformation. It takes the coordinates $xyz$ of the canonical 3D gaussians and time $t$ as input and outputs corresponding coordinate deformations $\Delta x, \Delta y, \Delta z$. The second deformation builds upon the first by adding $\Delta x, \Delta y, \Delta z$ to the original $xyz$, creating a modified set of coordinates that is then input into a new feature extraction network. The entire process is illustrated by image 3, 4 and 5 in Figure 2. This strategy achieves a successful alteration of the distribution of coordinates $xyz$ through the initial coordinate deformation, resulting in noise reduction and the generation of a more accurate deformation field. It's worth noting that during the early stage of training, we only perform the first deformation operation. Only after achieving acceptable results with the first deformation operation do we proceed to the second deformation operation to further enhance accuracy. To address the second problem, we design the DTS. The reason we decouple spatial-temporal aggregation is due to the presence of noise in the coordinates of the canonical 3D gaussians. If we directly perform spatial aggregation on the coordinates of the canonical 3D gaussians, the noise information is inevitably amplified after a series of aggregation operations such as k-nearest neighbors (KNN) [16], significantly affecting the results of the deformation field. Therefore, based on the first design NSS, we conduct spatial aggregation during the second deformation operation. Considering that temporal information is unrelated to the canonical 3D gaussians, temporal aggregation can be directly incorporated into the first deformation operation to enhance feature extraction capabilities. In order to reduce computational overhead and considering that temporal information has already been effectively extracted in the first deformation operation, we do not perform temporal aggregation in the second deformation operation. In conclusion, our main contributions are outlined as follows:

(i) We introduce a novel representation called Denoised Deformable Network with Temporal-Spatial Aggregation for high-fidelity and efficient dynamic scene rendering.

(ii) We promose the Noise Suppression Strategy, which can change the distribution of the coordinates of the canonical 3D gaussians, suppress noise and generate a more precise deformation field.

(iii) We promose the Decoupled Temporal-Spatial Aggregation Module to aggregate information from adjacent points and frames.

(iv) Extensive experiments on various real-world datasets demonstrate that our method achieves state-of-the-art rendering quality under a real-time level.

## 2 Related Work

**Dynamic NeRF.** Novel view synthesis has been a hot topic in academia for several years. NeRF [9] models static scenes implicitly using MLPs, and numerous studies [17, 18, 19, 2, 10, 20, 21] have extended its application to dynamic scenes through a canonical 3D grid structure and a deformation field. HyperNeRF [2] models object topology deformation using higher-dimensional inputs, while DyNeRF [1] employs time-conditioned NeRF to represent a 4D scene. However, these approaches, based on vanilla NeRF, suffer from high computational costs due to ray point sampling and volume rendering. To address this issue, several acceleration methods [22, 23, 24, 25, 26, 27, 15, 28, 29, 30, 31, 32] have been proposed for rendering dynamic scenes. DeVRF [22] introduces a grid representation, while IBR-based methods [24, 25] utilize multi-camera information for improved quality and efficiency. TensorRF [33] adopts multiple planes as explicit representations for direct dynamic scene modeling. Recent approaches such as K-Planes [34], Tensor4D [30], and HexPlane [15] have also been proposed. NeRFPlayer [35] introduces a unified streaming representation for both grid-based and plane-based methods, utilizing separate models to differentiate static and dynamic scene components, albeit at the cost of slow rendering times. HyperReel [36] suggests a flexible sampling network coupled with two planes for dynamic scene representation. Despite the improvements in training and rendering speed achieved by these methods, they still fall short of meeting real-time requirements.

**Dynamic Gaussian Splatting.** Recently, 3D Gaussian Splatting (3DGS) [12] has garnered increasing attention from researchers due to its superior rendering quality and real-time rendering speed. The method employs a soft point representation with attributes including position, rotation, density, and radiance, and utilizes differentiable point-based rendering for scene optimization. Soon after, several concurrent works [11, 13, 14, 37] have adapted 3D Gaussians for dynamic scenes. These methods represent dynamic scenes by establishing a canonical 3DGS and deforming it to individual frames using deformable fields. Yang et al. [13] predict per-Gaussian offsets using an additional MLP on canonical 3D gaussians, while Wu et al. [11] substitute the MLP with multi-resolution HexPlanes [15] and a lightweight MLP. Our work introduces the Noise Suppression Strategy to change the distribution of the coordinates of the canonical 3D gaussians and generate a more precise

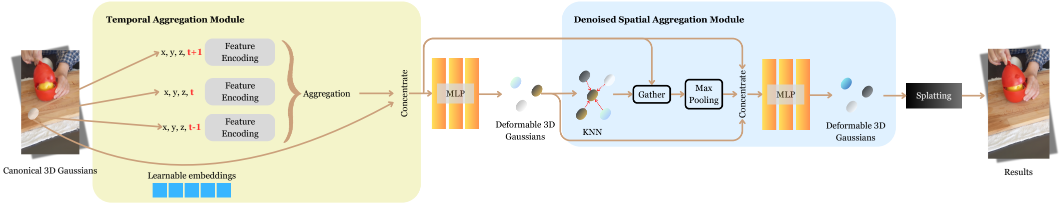

Figure 3: **The overall framework of our method DN-4DGS.** Our approach employs a two-stage deformation process. In the first deformation, the well-designed Temporal Aggregation Module is utilized to aggregate temporal information. After the first deformation, the coordinate distribution of 3D gaussians is altered, and noise is suppressed. Subsequently, we proceed with the second deformation, utilizing the Denoised Spatial Aggregation Module to aggregate spatial information.

deformation field. Additionally, for better aggregation of temporal-spatial information, we propose the Decoupled Temporal-Spatial Aggregation Module to consolidate information from adjacent points and frames.

# 3 Preliminary: 3D Gaussian Splatting

Given images at multiple known viewpoints and timesteps, 3D Gaussian Splatting (3DGS) [12] optimizes a set of attributes (position, opacity, anisotropic covariance and spherical harmonics) via differentiable rasterization. 3DGS can realize high-fidelity rendering of static 3D scenes in real-time.

Suppose a 3D Gaussian $G(i)$ has the following attributes: position $\mu_i$, opacity $\sigma_i$, covariance matrix $\Sigma_i$ and spherical harmonics $h_i$. The covariance matrix $\Sigma_i$ is decomposed as $\Sigma_i = \mathcal{R}\mathcal{S}\mathcal{S}^T\mathcal{R}^T$ for optimization, with $\mathcal{R}$ as a rotation matrix represented by a quaternion $q \in \mathbf{SO}(3)$, and $\mathcal{S}$ as a scaling matrix represented by a 3D vector $s$. Each Gaussian has an opacity value $\sigma_i$ to adjust its influence in rendering and is associated with sphere harmonic (SH) coefficients $h_i$ for view-dependent appearance. The final opacity of a 3D gaussian at any spatial point x can be represented as:

$$\alpha_i = \sigma_i e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)}. \tag{1}$$

To render a 2D image, 3D gaussians are projected to 2D space and aggregating them using fast $\alpha$-blending. The 2D covariance matrix and center are $\Sigma_i^{2D} = JW\Sigma W^T J^T$ and $\mu_i^{2D} = JW\mu_i$. The color $\mathbf{C}(u)$ of a pixel $u$ is rendered using the fast $\alpha$-blending operation:

$$\mathbf{C}(u) = \sum_{i \in N} T_i \alpha_i \mathbf{SH}(h_i, v_i), \tag{2}$$

where $T_i = \prod_{j=1}^{i-1}(1-\alpha_j)$, **SH** is the spherical harmonic function and $v_i$ is the view direction.

# 4 Method

## 4.1 Overview

Our goal is to reconstruct dynamic 3D scenes from single/multi-view videos. Following previous works [11, 13], we represent the geometry and appearance of the dynamic scene using canonical 3D gaussians and model the motion through the deformation fields. An overview of our method is shown in Figure 3. We first describe the details of the Noise Suppression Strategy (NSS) in Section 4.2. Then in Section 4.3, we present the design of the Decoupled Temporal-Spatial Aggregation Module (DTS). Section 4.4 details our optimization process.

## 4.2 Noise Suppression Strategy

In this section, we attempt to mitigate the terrible noise of the canonical 3D gaussians, as shown in Figure 5. Specifically, the Noise Suppression Strategy comprises two deformation operations. The first deformation operation is a standard deformation. It takes the coordinates $x, y, z$ of the canonical 3D gaussians and time $t$ as input and outputs corresponding coordinate deformations $\Delta x, \Delta y, \Delta z$. To simplify, here we only list the attributes $x, y, z$, while other attributes are detailed in the subsequent Section 4.3,

$$\Delta x, \Delta y, \Delta z = \Psi(x, y, z, T_n), \tag{3}$$

where $\Psi$ represents the first deformation operation, $T_n$ represents the set of $t$'s neighbors. The details of $\Psi$ are introduced in Section 4.3.1. Next, after obtaining $\Delta x, \Delta y, \Delta z$, we add them to the original $x, y, z$.

$$x', y', z' = x + \Delta x, y + \Delta y, z + \Delta z. \tag{4}$$

Following this, the second deformation operation is carried out.

$$\Delta x', \Delta y', \Delta z' = \Psi'(P_k, t), \tag{5}$$

where $\Psi'$ represents the second deformation operation, $P_k$ represents the set of $k$ neighbors of $(x', y', z')$. The details of $\Psi'$ are introduced in Section 4.3.2. In this deformation, due to the successful alteration of the input coordinate distribution during the first deformation stage, we can obtain more accurate Gaussian positions compared to the canonical Gaussian. As a result, the noise in the input is attenuated.

Overall, Noise Suppression Strategy (NSS) is a strategy that uses two stages of deformation to reduce the impact of noise on the deformable network. During this process, we have two training phases. In the early training phase, we only supervise the first deformation. Once the Gaussian coordinates obtained from the first deformation are relatively accurate, we add the second deformation and shift the supervision to it.

### 4.3 Decoupled Temporal-Spatial Aggregation Module

Local feature aggregation is very important for 3D point clouds, which can effectively extract local structure information. PointNet++ [16] introduces the set abstraction layer to aggregate information from spatially adjacent points, which has become a fundamental operation in various point cloud tasks [38, 39, 40, 41, 42]. Therefore, to enhance the accuracy of the deformation fields, an intuitive approach is to perform neighbor aggregation for each gaussian.

For 4D gaussians, there are four dimensions of information $x, y, z, t$ available for aggregation. As discussed in the introduction, performing local aggregation on noisy coordinates would further amplify the noise. Therefore, for $x, y, z$, spatial aggregation is conducted during the second deformation operation. Since temporal information is unrelated to the canonical 3D gaussians, temporal aggregation can be directly integrated into the first deformation operation to enhance feature extraction capabilities. To reduce computational overhead, and considering that temporal information has already been effectively extracted in the first deformation operation, we omit temporal aggregation in the second deformation operation. The entire process is referred to as decoupled temporal-spatial aggregation.

#### 4.3.1 Temporal Aggregation Moudle

For each gaussian $G_t(i)$, we first input $x, y, z, t$ into the Feature Encoding. Regarding Feature Encoding, we can utilize the MLPs from D3DGS [13] or the HexPlanes from 4DGaussian [11]. After that, we acquire $F_t(i)$ at time $t$. Next is a critical step, where we repeat the above-mentioned process to obtain the features $F_{t-1}(i)$ for time $t-1$ and the features $F_{t+1}(i)$ for time $t+1$. It's worth noting that here, the "1" represents one timestep. Next, similar to PointNet++ [16], we perform the aggregation operation. As illustrated in Figure 4, we merge $F_{t-1}(i)$, $F_t(i)$, and $F_{t+1}(i)$ together to form $F(i) \in \mathbb{R}^{3\times 1\times C_1}$ and input into a lightweight MLP for channel change. Then, we perform MaxPooling along the first dimension of $F(i)$ to generate $F_{\text{max}}(i) \in \mathbb{R}^{1\times C_2}$. Additionally, we introduce a new attribute $\mathcal{Y}_i$, which is a learnable embedding. This attribute can provide information independent of coordinates without interference due to adjacency. We have also observed a similar design in a recent work E-D3DGS [43]. Finally, $F_{\text{max}}(i)$, $F_t(i)$ and $\mathcal{Y}_i$ are concatenated together to generate deformation:

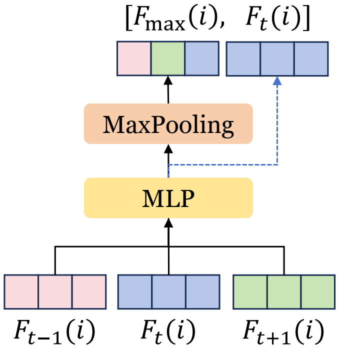

Figure 4: **The structure of aggregation operation.**

$$F_t(i)' = [F_{\text{max}}(i), F_t(i), \mathcal{Y}_i], \tag{6}$$

$$\mathcal{F}_\theta : F_t(i)' \rightarrow (\Delta x, \Delta y, \Delta z, \Delta r, \Delta s, \Delta \sigma, \Delta h), \tag{7}$$

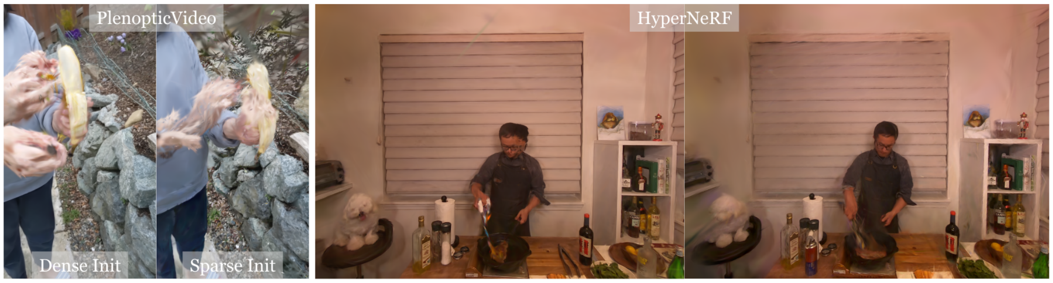

Figure 5: **More rendering images of canonical 3D gaussians.** Here, $Sparse\ Init$ refers to using the point cloud provided by the HyperNeRF [2] dataset ($\mathrm{COLMAP}_{\mathrm{SFM}}$ [44]) for Gaussian initialization, while $Dense\ Init$ denotes generating a denser point cloud via $\mathrm{COLMAP}_{\mathrm{MVS}}$ [44]. In fact, $Dense\ Init$ can produce better rendering quality, but due to the need for regenerating, it consumes more computational resources.

where $\mathcal{F}_\theta$ is the deformation MLP head, $r$ is a rotation quaternion, $s$ is a vector for scaling, $\sigma$ is an opacity, and $h$ is $\mathbf{SH}$ coefficients for modeling view-dependent color.

### 4.3.2 Denoised Spatial Aggregation Moudle

After obtaining the new $x', y', z'$, we input them into the denoised spatial aggregation module for spatial aggregation. Concretely, we calculate the k-nearest neighbors for each gaussian based on $x', y', z'$. Then, we aggregate the features of the k-nearest neighbors to obtain $F_n(i) \in \mathbb{R}^{1\times K\times C_3}$. MaxPooling is then performed on $F_n(i)$ to get $F_{nm}(i)$. Finally, $F_n(i)$, $F_{nm}(i)$ and $x', y', z'$ are concatenated together to generate the second deformation:

$$F_n(i)' = [F_n(i), F_{nm}(i), x', y', z'], \tag{8}$$

$$\mathcal{F}'_\theta : F_n(i)' \to (\Delta x, \Delta y, \Delta z, \Delta r, \Delta s, \Delta \sigma, \Delta h), \tag{9}$$

where $\mathcal{F}_\theta$ is the deformation MLP head in the second deformation operation.

Overall, Decoupled Temporal-Spatial Aggregation Module (DTS) is a specific feature aggregation method we propose. Unlike 4DGaussian, D3DGS lacks explicit spatiotemporal aggregation, so we designed DTS to aggregate spatiotemporal information. Considering that inaccurate coordinate relationships would be amplified through spatial aggregation (KNN), we only perform spatial aggregation in the second stage of NSS, which is DSAM. We name the spatial aggregation module "Denoised Spatial Aggregation Module" (DSAM) because the Gaussian coordinates input into DSAM are more accurate (denoised), as shown in the fourth column of Figure 9. Therefore, we prefix the Spatial Aggregation Module with "Denoised". DSAM itself does not have denoising capabilities; it solely performs spatial feature aggregation.

### 4.4 Optimization

The parameters to be optimized include the deformable network and the attributes of each 3D gaussian $G(i)$: $\mu_i$, $\sigma_i$, $\Sigma_i$, $h_i$ and $\mathcal{Y}_i$. Following 4DGaussian [11], we use the reconstruction loss $\mathcal{L}_1$ and gird-based TV loss [15, 45, 34, 46] $\mathcal{L}_{tv}$ to supervise the training process. Additionally, we add a D-SSIM term $\mathcal{L}_{ssim}$ to improve structural similarity:

$$\mathcal{L} = \lambda \mathcal{L}_1 + (1-\lambda)\mathcal{L}_{ssim} + \mathcal{L}_{tv}, \tag{10}$$

where $\lambda$ is the hyperparameter. It is worth noting that we employ a two-stage training strategy. During the early stages of training, we exclusively execute the first deformation operation. Once satisfactory results are attained with the initial deformation operation, we then proceed to implement the second deformation operation to further refine accuracy. The reason for this strategy is that only the deformation $\Delta x, \Delta y, \Delta z$ in the first stage is sufficiently precise to remove a large amount of noise, thereby positively impacting the deformation in the second stage.

## 5 Experiment

### 5.1 Experimental Setup

**Dataset and Metrics. PlenopticVideo** [1] dataset includes 20 multi-view videos, with each scene consisting of either 300 frames, except for the flame salmon scene, which comprises 1200 frames.

| Method | PSNR(↑) | SSIM(↑) | LPIPS(↓) | Time(↓) | FPS(↑) | Storage(MB)(↓) |
|---|---|---|---|---|---|---|
| DyNeRF [1] | 29.58 | - | 0.099 | 1344 hours | 0.01 | 28 |
| NeRFPlayer [35] | 30.69 | 0.909 | 0.111 | 6 hours | 0.045 | - |
| HyperReel [36] | 31.10 | 0.921 | 0.096 | 9 hours | 2.0 | 360 |
| HexPlane-all* [15] | 31.70 | 0.984 | 0.075 | 12 hours | 0.2 | 250 |
| KPlanes [34] | 31.63 | 0.964 | - | 1.8 hours | 0.3 | 309 |
| 4DGS [51] | 31.19 | 0.940 | 0.051 | 9.5 hours | 19.5 | 8700 |
| E-D3DGS [43] | 31.31 | 0.945 | 0.037 | 2 hours | 43.1 | 35 |
| 4DGaussian [11] | 31.15 | 0.940 | 0.049 | 40 mins | 30 | 90 |
| Ours | 32.02 | 0.944 | 0.043 | 50 mins | 15 | 112 |

Table 1: **Quantitative comparison on PlenopticVideo dataset.** We display the average PSNR/SSIM/LPIPS (Alex) metrics for novel view synthesis on dynamic scenes, with each cell colored to indicate the best, second best, and third best.

These scenes encompass a relatively long duration and various movements, with some featuring multiple objects in motion. We utilized PlenopticVideo dataset to observe the capability to capture dynamic areas. Total six scenes (coffee martini, cook spinach, cut roasted beef, flame salmon, flame steak, sear steak) are utilized to train and render. Rendering resolution is set to 1352 × 1014 **HyperNeRF** [2] dataset includes videos using two Pixel 3 phones rigidly mounted on a handheld capture rig. We train and render on four scenes (3D Printer, Banana, Broom, Chicken) at a resolution downsampled by a factor of two to 540 × 960. **NeRF-DS** [47] dataset consists of seven captured videos (Sieve, Press, Plate, Cup, As, Bell, Basin) with camera pose estimated using colmap [44]. The dataset involves a variety of rigid and non-rigid deformation of various objects. We train and render on the seven scenes. Rendering resolution is set to 480 × 270.

We report the quality of rendered images using PSNR, SSIM [48], MS-SSIM and LPIPS [49]. Higher PSNR, SSIM and MS-SSIM values and lower LPIPS values indicate better visual quality. To PlenopticVideo dataset, we report PSNR, SSIM and LPIPS (Alex). To HyperNeRF dataset, we report PSNR, SSIM and MS-SSIM. To NeRF-DS dataset, we report PSNR, MS-SSIM and LPIPS (VGG).

**Implementation Details.** We train our model on a single RTX3090. The optimizer we utilize is Adam [50]. The learning rate is initially set at 1.6e-4, gradually decreasing exponentially to 1.6e-6 by the end of the training process. The learning rate for the voxel grid is initialized at 1.6e-3 and exponentially decays to 1.6e-5. For hyperparameters, we tune $K, \lambda$ as 16, 0.9 respectively. More details will be shown in the Appendix.

### 5.2 Comparison with existing methods.

**Results on PlenopticVideo.** Table 1 reports the results on PlenopticVideo dataset. Refer to the Appendix for per-scene details. Due to the incorporation of the Noise Suppression Strategy, which alters the distribution of canonical 3D gaussians coordinates to suppress noise, as well as the utilization of the Decoupled Temporal-Spatial Aggregation Module for feature aggregation, our approach demonstrates superior reconstruction quality across all metrics compared to the baseline (4DGaussian [11]). In fact, PSNR and LPIPS are currently the state-of-the-art metrics. As the table shows, despite our method involving two stages of deformation, resulting in a slight weakening in training time and FPS, it overall meets the requirements for rapid training and real-time demands. Regarding storage, due to the presence of a new attribute $\mathcal{Y}_i$, it may slightly exceed the baseline. To vividly illustrate the differences between our method and others, we visualize the qualitative results in Figure 6. From the regions highlighted in red boxes, it is evident that our method can render higher-quality images.

**Results on HyperNeRF.** Table 2 reports the results on HyperNeRF dataset. In this dataset, we present results for both $Sparse\ Init$ and $Dense\ Init$. $Sparse\ Init$ refers to using the point cloud provided by the HyperNeRF dataset (COLMAP$_{\text{SFM}}$ [44]) for Gaussian initialization, while $Dense\ Init$ denotes generating a denser point cloud via COLMAP$_{\text{MVS}}$ [44]. From Table 2, it can be observed that our method outperforms other Gaussian-based methods under both $Sparse\ Init$ and $Dense\ Init$ settings. Moreover, under the $Dense\ Init$ setting, our method achieves the current state-of-the-art performance. More importantly, we find that 4DGaussian is highly sensitive to the sparsity or density of Gaussian initialization. In contrast, our approach benefits from noise suppression and feature

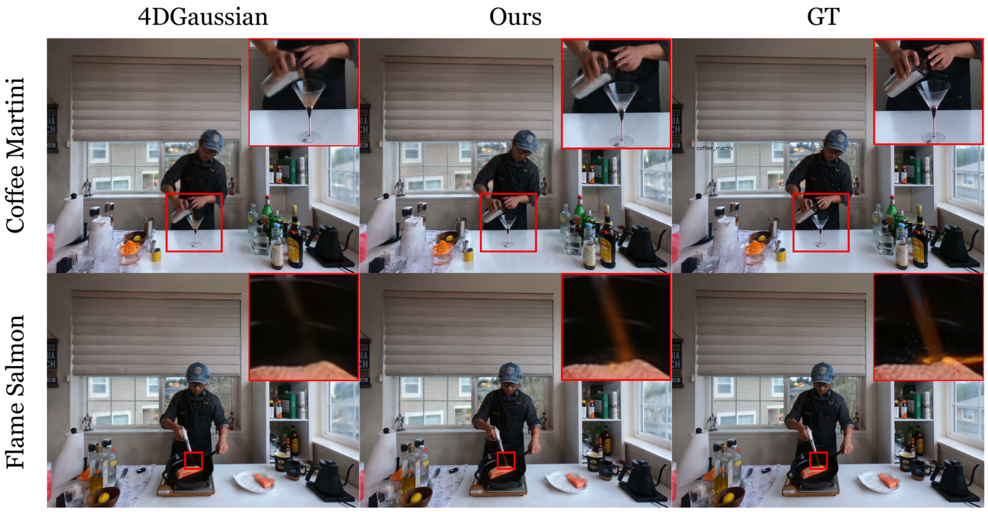

Figure 6: **Qualitative comparisons on PlenopticVideo Dataset.**

| Method | PSNR(↑) | SSIM(↑) | MS-SSIM(↑) | Time(↓) | FPS(↑) | Storage(MB)(↓) |
|---|---|---|---|---|---|---|
| Nerfies [19] | 22.23 | - | 0.803 | $\sim$ hours | <1 | - |
| HyperNeRF DS [2] | 22.2 | 0.598 | 0.811 | 32 hours | <1 | - |
| TiNeuVox-B [45] | 24.30 | 0.616 | 0.837 | 30 mins | 1 | 48 |
| D3DGS‡ [13] | 21.50 | - | - | 2 hours | 10 | 18 |
| 4DGaussian‡ [11] | 21.80 | 0.573 | 0.710 | 50 mins | 38 | 11 |
| Ours‡ | 23.31 (+1.51) | 0.618 (+0.045) | 0.768 (+0.058) | 1.1 hours | 24 | 12 |
| D3DGS* | 23.43 | - | - | 3.5 hours | 7 | 88 |
| 4DGaussian* | 25.20 | 0.682 | 0.845 | 1 hour | 34 | 61 |
| Ours* | 25.59 (+0.39) | 0.691 (+0.009) | 0.863 (+0.018) | 1.2 hours | 20 | 68 |

Table 2: **Quantitative comparison on HyperNeRF dataset.** Here, ‡ represents that we train the model based on $Sparse\ Init$. * represents that we train the model based on $Dense\ Init$.

aggregation, resulting in a more pronounced performance improvement under the sparse setting. The qualitative results can be observed from Figure 7. From the regions highlighted in red boxes, our method can render higher-quality images, as further supported by PSNR in gray cells.

**Results on NeRF-DS.** Table 4 presents the results on NeRF-DS dataset. Our proposed method achieves better performance compared to previous methods, demonstrating the effectiveness and generalization of our method. More qualitative results are shown in the Appendix.

**Results on D-NeRF.** As illustrated in the Figure 8, we have visualized the canonical results for both 4DGaussian and D3DGS. The results show that D3DGS has less noise compared to 4DGaussian, but noise is still present in moving areas. The quantitative results on D-NeRF dataset, as shown in the Table 3, indicate that our method improves performance on both 4DGaussian and D3DGS, with enhancements on 4DGaussian surpassing those on D3DGS. Regarding the parameters of the deformation networks, in 4DGaussian: Most of the parameters are composed of the HexPlane and deformation head. Our method introduces an additional deformation operation, resulting in a slight increase in the number of parameters compared to the baseline, with an increase of only 0.03M parameters. In D3DGS: The network primarily consists of an MLP-based deformation network. To reduce the computational load, our method halves the number of MLP layers, resulting in fewer overall parameters compared to the original D3DGS.

### 5.3 Ablation Studies

**Evaluation of the model with different designs.** To evaluate the effectiveness of proposed components, we conduct an ablation study in Table 5 on PlenopticVideo dataset. Here, NSS, TAM, DSAM represents the Noise Suppression Strategy, Temporal Aggregation Moudle and Denoised

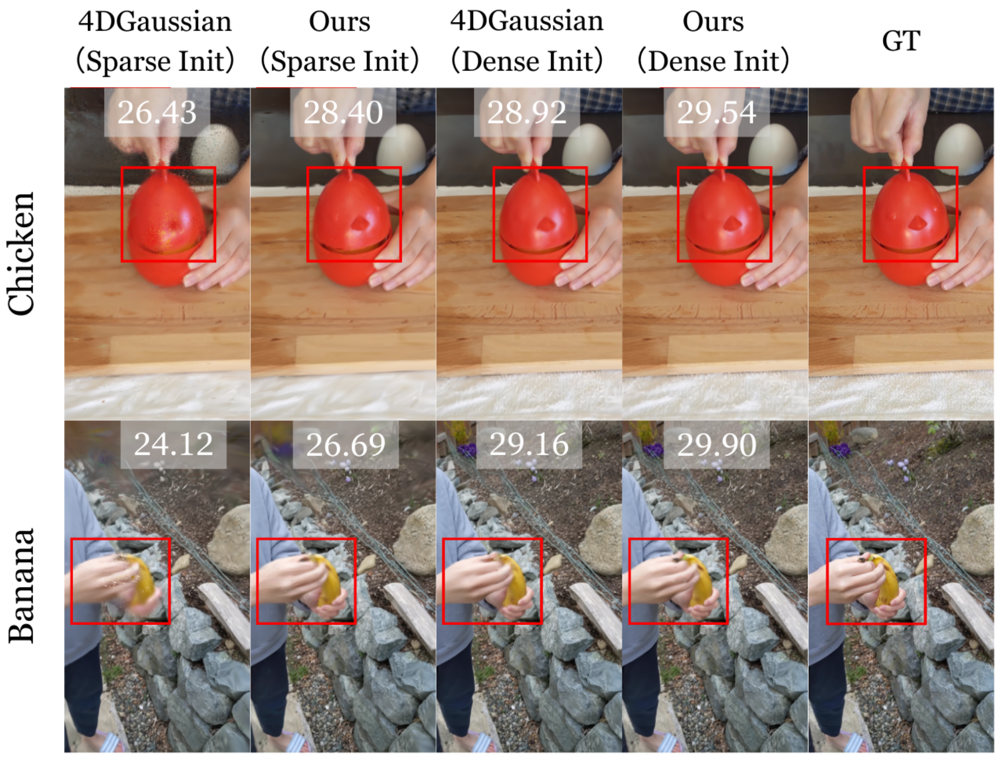

Figure 7: **Qualitative comparisons on HyperNeRF Dataset.** In the gray cells, the numbers represent PSNR.

| Method | PSNR(↑) | SSIM(↑) | Parameter (M)(↓) |
|---|---|---|---|
| 4DGaussian [11] | 34.05 | 0.9787 | 3.38 |
| 4DGaussian+Ours | 34.53(+0.48) | 0.9811(+0.0024) | 3.41(+0.03) |
| D3DGS [13] | 39.51 | 0.9902 | 0.52 |
| D3DGS+Ours | 39.87(+0.36) | 0.9922(+0020) | 0.39(-0.13) |

Table 3: **Quantitative comparison on D-NeRF dataset.** Here, parameter refers to the parameters of the deformation networks corresponding to different baselines.

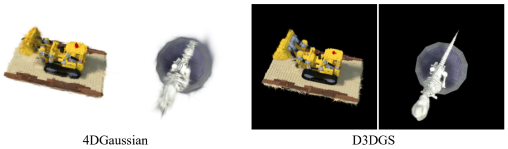

Figure 8: **The canonical results for both 4DGaussian and D3DGS.**

Spatial Aggregation Moudle respectively. Specifically, the second row shows that with the use of two-stage deformation operations, our model can acquire a certain degree of improvement in quality. This highlights the meaning of noise suppression. The third row demonstrates that with the help of temporal aggregation, a performance gain of 0.41, 0.02, 0.003 has been achieved in PSNR, SSIM and LPIPS. The fourth row demonstrates the effective collaboration between NSS and TAM, resulting in performance improvement. The fifth row indicates that if we do not utilize two-stage training, but instead only perform spatial aggregation on canonical 3D gaussians, it not only fails to bring about an improvement in quality but also leads to a decrease. The sixth row indicates that if we replace TAM with an ordinary deformation network, there is a slight drop in performance compared to the last row. The last row indicates that if we combine all components together, the performance can reach its optimal level.

**Effectiveness of the two-stage deformation operations.** To validate the significance of two-stage deformation operations, we conducted visual experiments. As shown in Figure 9, canonical 3D aussians exhibit a significant amount of noise, which severely affects the accuracy of the deformation field. To alleviate this issue, we employed two-stage deformation. As depicted in the fourth column of the figure, after the first deformation, there is a significant change in the distribution of coordinates $xyz$, effectively suppressing the noise. Moreover, due to the design of temporal aggregation, the corresponding PSNR value is even higher than that of 4DGaussian. Finally, by performing the second deformation operation on the basis of the first-stage deformation, the performance is further improved. This is attributed to the cleaner coordinates $xyz$ and the design of spatial aggregation.

| Method | PSNR(↑) | SSIM(↑) | LPIPS(↓) |
|---|---|---|---|
| TiNeuVox [45] | 21.61 | 0.823 | 0.277 |
| HyperNeRF [2] | 23.45 | 0.849 | 0.199 |
| NeRF-DS [47] | 23.60 | 0.849 | 0.182 |
| 3D-GS [12] | 20.29 | 0.782 | 0.292 |
| D3DGS [13] | 23.92 | 0.847 | 0.184 |
| Ours | 24.36 | 0.865 | 0.171 |

Table 4: **Quantitative comparison on NeRF-DS dataset.**

| NSS | TAM | DSAM | PSNR(↑) | SSIM(↑) | LPIPS(↓) |
|---|---|---|---|---|---|
| ✗ | ✗ | ✗ | 31.15 | 0.940 | 0.049 |
| ✓ | ✗ | ✗ | 31.31 | 0.941 | 0.047 |
| ✗ | ✓ | ✗ | 31.56 | 0.942 | 0.046 |
| ✓ | ✓ | ✗ | 31.69 | 0.943 | 0.045 |
| ✗ | ✓ | ✓ | 31.31 | 0.936 | 0.052 |
| ✓ | ✗ | ✓ | 31.72 | 0.943 | 0.045 |
| ✓ | ✓ | ✓ | **32.02** | **0.944** | **0.043** |

Table 5: **Evaluation of the model with different designs on PlenopticVideo dataset.**

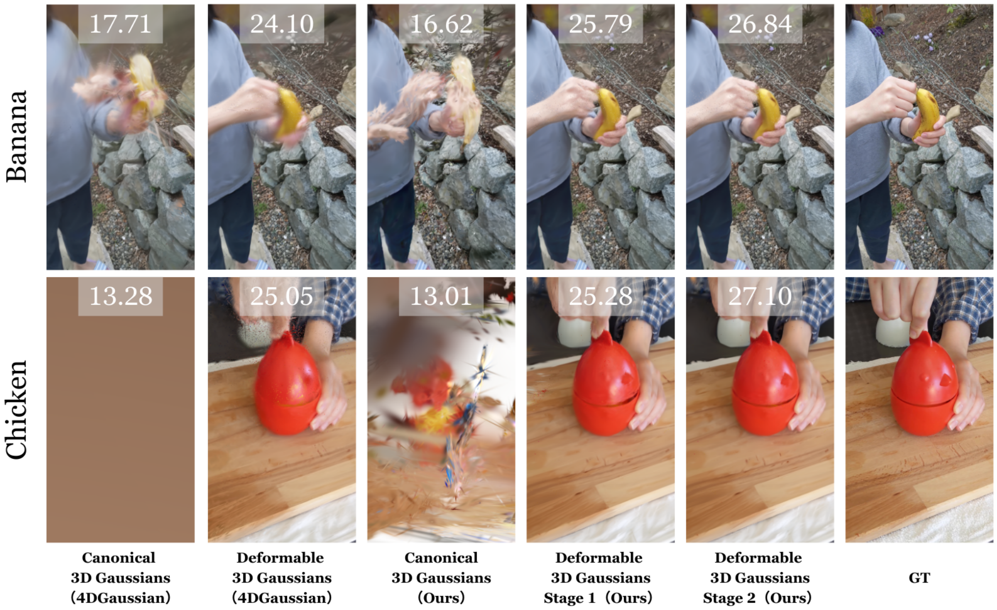
Banana
Chicken
17.71
24.10
16.62
25.79
26.84
13.28
25.05
13.01
25.28
27.10
Canonical 3D Gaussians (4DGaussian)
Deformable 3D Gaussians (4DGaussian)
Canonical 3D Gaussians (Ours)
Deformable 3D Gaussians Stage 1 (Ours)
Deformable 3D Gaussians Stage 2 (Ours)
GT

Figure 9: **Effectiveness of the two-stage deformation operations.** In the gray cells, the numbers represent PSNR.

### 5.4 Limitations and Future Work

Although two-stage deformation can alter the coordinate distribution of canonical 3D gaussians and reduce the noise introduced into the deformation field, the lack of simultaneous supervision for both stages [52, 53, 41] poses a challenge. Consequently, during the second stage, due to the lack of supervision in the first-stage deformation, the direction of coordinate deformation becomes uncontrollable to some extent. This, in turn, affects the spatial feature aggregation in the second stage. To address this issue, future work should explore the direction of simultaneous supervision.

## 6 Conclusion

In this paper, we introduce a novel representation called Denoised Deformable Network with Temporal-Spatial Aggregation for Dynamic Scene Rendering. We promose the Noise Suppression Strategy, which can change the distribution of the coordinates of the canonical 3D gaussians, suppress noise and generate a more precise deformation field. To aggregate information from adjacent points and frames, we promose the Decoupled Temporal-Spatial Aggregation Module. Extensive experiments on various real-world datasets demonstrate that our method achieves state-of-the-art rendering quality under a real-time level.

## 7 Acknowledgements

This work was partially supported by the National Nature Science Foundation of China (12150007, 62306294), Dreams Foundation of Jianghuai Advance Technology Center (No.2023-ZM01Z019), and the Youth Innovation Promotion Association.

# References

[1] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022.

[2] 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. *arXiv preprint arXiv:2106.13228*, 2021.

[3] Hansung Kim, Luca Remaggi, Philip JB Jackson, and Adrian Hilton. Immersive spatial audio reproduction for vr/ar using room acoustic modelling from 360 images. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 120–126. IEEE, 2019.

[4] Rafael Pagés, Konstantinos Amplianitis, David Monaghan, Jan Ondřej, and Aljosa Smolić. Affordable content creation for free-viewpoint video and vr/ar applications. *Journal of Visual Communication and Image Representation*, 53:192–201, 2018.

[5] Jiahao Lu, Jiacheng Deng, and Tianzhu Zhang. Bsnet: Box-supervised simulation-assisted mean teacher for 3d instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20374–20384, 2024.

[6] Zhuoyuan Li, Yubo Ai, Jiahao Lu, ChuXin Wang, Jiacheng Deng, Hanzhi Chang, Yanzhe Liang, Wenfei Yang, Shifeng Zhang, and Tianzhu Zhang. Mamba24/8d: Enhancing global interaction in point clouds via state space model. *arXiv preprint arXiv:2406.17442*, 2024.

[7] Jiacheng Deng, Jiahao Lu, and Tianzhu Zhang. Diff3detr: Agent-based diffusion model for semi-supervised 3d object detection. *arXiv preprint arXiv:2408.00286*, 2024.

[8] Kensuke Hisatomi, Kimihiro Tomiyama, Miwa Katayama, Yuichi Iwadate, Koji Matsunaga, Yoshiyuki Ito, and Wataru Ishihara. A method of video production using dynamic 3d models and its application to making scenes of a crowd. *SMPTE motion imaging journal*, 118(7):29–36, 2009.

[9] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[10] 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.

[11] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023.

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):1–14, 2023.

[13] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023.

[14] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. *arXiv preprint arXiv:2312.14937*, 2023.

[15] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 130–141, 2023.

[16] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[17] Xiang Guo, Jiadai Sun, Yuchao Dai, Guanying Chen, Xiaoqing Ye, Xiao Tan, Errui Ding, Yumeng Zhang, and Jingdong Wang. Forward flow for novel view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16022–16033, 2023.

[18] 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.

[19] 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.

[20] 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 dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021.

[21] 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.

[22] Jia-Wei Liu, Yan-Pei Cao, Weijia Mao, Wenqiao Zhang, David Junhao Zhang, Jussi Keppo, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Devrf: Fast deformable voxel radiance fields for dynamic scenes. *Advances in Neural Information Processing Systems*, 35:36762–36775, 2022.

[23] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023.

[24] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[25] Haotong Lin, Sida Peng, Zhen Xu, Tao Xie, Xingyi He, Hujun Bao, and Xiaowei Zhou. Im4d: High-fidelity and real-time novel view synthesis for dynamic scenes. *arXiv preprint arXiv:2310.08585*, 2023.

[26] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics (ToG)*, 40(4):1–13, 2021.

[27] Sida Peng, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Representing volumetric videos as dynamic mlp maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4252–4262, 2023.

[28] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[29] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023.

[30] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16632–16642, 2023.

[31] Feng Wang, Sinan Tan, Xinghang Li, Zeyue Tian, Yafei Song, and Huaping Liu. Mixed neural voxels for fast multi-view video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19706–19716, 2023.

[32] Liao Wang, Qiang Hu, Qihan He, Ziyu Wang, Jingyi Yu, Tinne Tuytelaars, Lan Xu, and Minye Wu. Neural residual radiance fields for streamably free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 76–87, 2023.

[33] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022.

[34] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023.

[35] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, 2023.

[36] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhoefer, Johannes Kopf, Matthew O'Toole, and Changil Kim. Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16610–16620, 2023.

[37] Ruijie Zhu, Yanzhe Liang, Hanzhi Chang, Jiacheng Deng, Jiahao Lu, Wenfei Yang, Tianzhu Zhang, and Yongdong Zhang. Motiongs: Exploring explicit motion guidance for deformable 3d gaussian splatting. *arXiv preprint arXiv:2410.07707*, 2024.

[38] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer, 2022.

[39] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

[40] Weiguang Zhao, Yuyao Yan, Chaolong Yang, Jianan Ye, Xi Yang, and Kaizhu Huang. Divide and conquer: 3d point cloud instance segmentation with point-wise binarization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 562–571, 2023.

[41] Jiahao Lu, Jiacheng Deng, Chuxin Wang, Jianfeng He, and Tianzhu Zhang. Query refinement transformer for 3d instance segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18516–18526, 2023.

[42] Jiacheng Deng, Jiahao Lu, and Tianzhu Zhang. Unsupervised template-assisted point cloud shape correspondence network. *arXiv preprint arXiv:2403.16412*, 2024.

[43] Jeongmin Bae, Seoha Kim, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Per-gaussian embedding-based deformation for deformable 3d gaussian splatting. *arXiv preprint arXiv:2404.03613*, 2024.

[44] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[45] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[46] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.

[47] Zhiwen Yan, Chen Li, and Gim Hee Lee. Nerf-ds: Neural radiance fields for dynamic specular objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8285–8295, 2023.

[48] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[49] 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 conference on computer vision and pattern recognition*, pages 586–595, 2018.

[50] P Kingma Diederik. Adam: A method for stochastic optimization. *(No Title)*, 2014.

[51] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023.

[52] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 213–229. Springer, 2020.

[53] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[54] Xiang Guo, Jiadai Sun, Yuchao Dai, Guanying Chen, Xiaoqing Ye, Xiao Tan, Errui Ding, Yumeng Zhang, and Jingdong Wang. Forward flow for novel view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16022–16033, 2023.

# A Appendix / supplemental material

## A.1 Overview

This supplementary material provides more model and experimental details to understand our proposed method. After that, we present more experiments to demonstrate the effectiveness of our methods. Finally, we show a rich visualization of our modules.

## A.2 More Model Details

**Feature Encoding.** As illustrated in Section 4.3.1, we can utilize the MLPs from D3DGS [13] or the HexPlanes from 4DGaussian [11] as Feature Encoding. Specifically, for both PlenopticVideo [1] and HyperNeRF [2], we use the HexPlanes to encode per-gaussian's feature. The complete details can be referred to 4DGaussian's main text. For NeRF-DS [47], we utilize the MLPs for feature encoding. The complete details can be referred to D3DGS's main text.

## A.3 More Implementation Details

For both PlenopticVideo and HyperNeRF, the total training comprises 14,000 iterations, with the first stage encompassing 5,000 iterations. For NeRF-DS, the total training comprises 40,000 iterations, with the first stage encompassing 15,000 iterations. The dimension of attribute $\mathcal{Y}$ is set as 16.

## A.4 Comparison with 3DGStream

**Similarities:** Our method and 3DGStream both perform deformations on 3D Gaussians (3DGs) where absolute positions and relative positional relationships are more accurately maintained. For each timestep $i$, 3DGStream uses the 3DGs from the previous timestep $i - 1$ as initialization. In contrast, our method uses canonical Gaussians (which are time-independent) as the initialization. To achieve accurate relative positional relationships and minimize noise interference from the canonical Gaussians, we employ a two-stage deformation strategy. The first stage obtains accurate 3DGs, and the second stage further deforms these 3DGs to achieve preciser rendering results.

**Advantages:** 1. *Flexibility:* Unlike 3DGStream, our method does not require the results from the previous timestep. We can render at any arbitrary time without needing the previous timestep's 3DGs. On the other hand, 3DGStream relies on the 3DGs from the previous timestep for rendering the next. 2. *Robustness:* 3DGStream heavily depends on the reconstruction quality at timestep 0. If the initial reconstruction is poor, subsequent reconstructions will be negatively affected, and these errors can accumulate over time. Our method, however, starts with noisy canonical Gaussians and improves the reconstruction quality through a two-stage deformation process, resulting in progressively better reconstructions.

**Disadvantages:** *Training Efficiency:* 3DGStream employs an online reconstruction approach, leading to shorter training time and faster rendering speed. In contrast, our method involves offline training, which results in relatively longer training time.

## A.5 Detailed Results

In Table 6, Table 7 and Table 8, we provide the results for individual scenes associated with Section 5.2 of the main text. It can be observed that our method achieved superior metrics in almost every scene compared to previous methods, demonstrating the effectiveness and generalization of our method under various scenes.

## A.6 More Ablation Studies

**The effectiveness of $\mathcal{Y}$.** As depicted in Table 9, we conduct an ablation study on the flame steak scene of PlenopticVideo dataset to examine the impact of $\mathcal{Y}$. From the table, it is evident that $\mathcal{Y}$ can enhance rendering quality and $\mathcal{Y}$=16 is the best setting.

**Ablation study of timestep in Temporal Aggregation Moudle.** From Table 10, it is evident that when timestep is set as 1×, the performance is the best. We speculate that the reason for the lowest

| Method | Cut Beef | | Cook Spinach | | Sear Steak | |
|---|---|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| NeRFPlayer [35] | 31.83 | 0.928 | 32.06 | 0.930 | 32.31 | 0.940 |
| HexPlane [15] | 32.71 | 0.985 | 31.86 | 0.983 | 32.09 | 0.986 |
| KPlanes [34] | 31.82 | 0.966 | 32.60 | 0.966 | 32.52 | 0.974 |
| MixVoxels [31] | 31.30 | 0.965 | 31.65 | 0.965 | 31.43 | 0.971 |
| 4DGaussian [11] | 32.90 | 0.957 | 32.46 | 0.949 | 32.49 | 0.957 |
| Ours | 33.49 | 0.960 | 32.91 | 0.951 | 33.98 | 0.959 |
| Method | Flame Steak | | Flame Salmon | | Coffee Martini | |
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| NeRFPlayer [35] | 27.36 | 0.867 | 26.14 | 0.849 | 32.05 | 0.938 |
| HexPlane [15] | 31.92 | 0.988 | 29.26 | 0.980 | - | - |
| KPlanes [34] | 32.39 | 0.970 | 30.44 | 0.953 | 29.99 | 0.953 |
| MixVoxels [31] | 31.21 | 0.970 | 29.92 | 0.945 | 29.36 | 0.946 |
| 4DGaussian [11] | 32.51 | 0.954 | 29.20 | 0.917 | 27.34 | 0.905 |
| Ours | 33.51 | 0.958 | 29.19 | 0.921 | 29.04 | 0.915 |

Table 6: **Per-scene results of PlenopticVideo dataset.**

| Method | 3D Printer | | Chicken | | Broom | | Banana | |
|---|---|---|---|---|---|---|---|---|
| | PSNR | MS-SSIM | PSNR | MS-SSIM | PSNR | MS-SSIM | PSNR | MS-SSIM |
| Nerfies [19] | 20.6 | 0.83 | 26.7 | 0.94 | 19.2 | 0.56 | 22.4 | 0.87 |
| HyperNeRF [2] | 20.0 | 0.59 | 26.9 | 0.94 | 19.3 | 0.59 | 23.3 | 0.90 |
| TiNeuVox-B [45] | 22.8 | 0.84 | 28.3 | 0.95 | 21.5 | 0.69 | 24.4 | 0.87 |
| FFDNeRF [54] | 22.8 | 0.84 | 28.0 | 0.94 | 21.9 | 0.71 | 24.3 | 0.86 |
| 3D-GS [12] | 18.3 | 0.60 | 19.7 | 0.70 | 20.6 | 0.63 | 20.4 | 0.80 |
| 4DGaussian [11]‡ | 20.9 | 0.75 | 24.1 | 0.82 | 20.0 | 0.53 | 22.2 | 0.74 |
| Ours‡ | 21.6 | 0.78 | 26.2 | 0.89 | 20.8 | 0.57 | 24.7 | 0.84 |
| 4DGaussian* | 22.1 | 0.81 | 28.7 | 0.93 | 22.0 | 0.70 | 28.0 | 0.94 |
| Ours* | 22.1 | 0.81 | 29.2 | 0.95 | 22.3 | 0.74 | 28.7 | 0.95 |

Table 7: **Perscene results of HyperNeRF dataset by different models.** Here, ‡ represents that we train the model based on $Sparse\ Init$. * represents that we train the model based on $Dense\ Init$.

| Method | PSNR | | | | | | |
|---|---|---|---|---|---|---|---|
| | Sieve | Plate | Bell | Press | Cup | As | Basin |
| 3D-GS [12] | 23.16 | 16.14 | 21.01 | 22.89 | 21.71 | 22.69 | 18.42 |
| TiNeuVox [45] | 21.49 | 20.58 | 23.08 | 24.47 | 19.71 | 21.26 | 20.66 |
| HyperNeRF [2] | 25.43 | 18.93 | 23.06 | 26.15 | 24.59 | 25.58 | 20.41 |
| NeRF-DS [47] | 25.78 | 20.54 | 23.19 | 25.72 | 24.91 | 25.13 | 19.96 |
| D3DGS [13] | 25.72 | 20.40 | 25.24 | 25.70 | 24.35 | 26.35 | 19.70 |
| Ours | 26.60 | 20.92 | 25.48 | 26.24 | 24.95 | 26.54 | 19.79 |

Table 8: **Perscene results of NeRF-DS dataset by different models.**

performance at 0.5× might be attributed to the absence of images corresponding to ±0.5 timestep in the dataset. Consequently, the lack of supervision at this timestep may induce significant feature biases, resulting in relatively poorer performance.

**Ablation study of $K$ in Denoised Spatial Aggregation Moudle.** From Table 11, it is evident that setting $K = 16$ yields the best results.

**Ablation study of the iteration rounds of the first stage.** From Table 12, it is evident that the two-stage training strategy is meaningful. If we train both deformation operations simultaneously, the performance is poor, as indicated in the first row. Only by first training the first deformation network and then proceeding to train the second deformation network after the first deformation network

| Setting | Dim | PSNR | SSIM |
|---|---|---|---|
| W/o $\mathcal{Y}$ | - | 33.13 | 0.956 |
| W $\mathcal{Y}$ | 4 | 33.33 | 0.957 |
| W $\mathcal{Y}$ | 16 | **33.51** | **0.958** |
| W $\mathcal{Y}$ | 32 | 33.40 | 0.958 |
| W $\mathcal{Y}$ | 64 | 33.29 | 0.957 |

Table 9: **The effectiveness of $\mathcal{Y}$.** We compare the results on the flame steak scene of PlenopticVideo dataset.

| Setting | PSNR | SSIM |
|---|---|---|
| 0.5× | 33.40 | 0.957 |
| 1× | **33.51** | **0.958** |
| 2× | 33.45 | 0.957 |

Table 10: **Ablation study of timestep in Temporal Aggregation Moudle.** We compare the results on the flame steak scene of PlenopticVideo dataset.

| $K$ | PSNR | SSIM |
|---|---|---|
| 4 | 33.39 | 0.957 |
| 16 | **33.51** | **0.958** |
| 32 | 33.35 | 0.957 |

Table 11: **Ablation study of $K$ in Denoised Spatial Aggregation Moudle.** We compare the results on the flame steak scene of PlenopticVideo dataset.

| Rounds | PSNR | SSIM |
|---|---|---|
| 0 | 32.89 | 0.951 |
| 4000 | 33.40 | 0.957 |
| 6000 | 33.38 | 0.957 |
| 8000 | **33.51** | **0.958** |
| 10000 | 33.37 | 0.957 |

Table 12: **Ablation study of the iteration rounds of the first stage.** We compare the results on the flame steak scene of PlenopticVideo dataset.

is well-trained can we achieve optimal performance. Additionally, the optimal number of iteration roudns is 8000.

**Ablation study of the iteration rounds of the first stage.**

### A.7 Assets Availability

The datasets that support the findings of this study are available in the following repositories: HyperNeRF [2] at `https://github.com/google/hypernerf/releases/tag/v0.1` under Apache-2.0 license. NeRF-DS [47] at `https://github.com/JokerYan/NeRF-DS/releases/tag/v0.1-pre-release` under Apache-2.0 license. PlenopticVideo [1] at `https://github.com/facebookresearch/Neural_3D_Video?tab=License-1-ov-file` under CC BY-NC 4.0 license. The code of our baseline [11, 13] is available at `https://github.com/ingra14m/Deformable-3D-Gaussians` under MIT license and `https://github.com/hustvl/4DGaussians` under Gaussian-Splatting license.

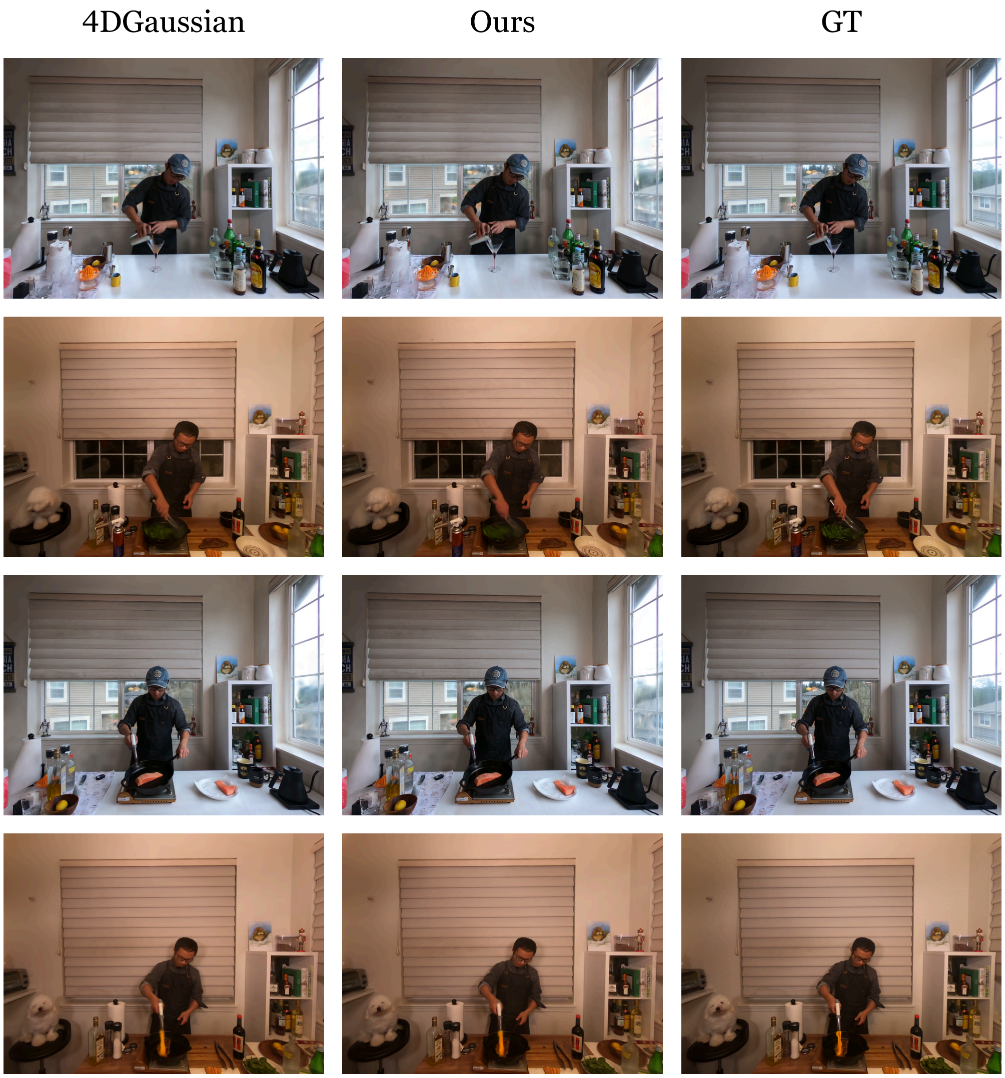

Figure 10: **Qualitative comparisons on PlenopticVideo Dataset.**

### A.8 More Visual Comparison

Figure 10 shows more visual comparisons on PlenopticVideo Dataset. We compare the results of 4DGaussian and our model.

Figure 11 and Figure 12 shows more visual comparisons on HyperNeRF Dataset. We compare the results of 4DGaussian and our model.

Figure 13 shows more visual comparisons on NeRF-DS Dataset. We compare the results of 4DGaussian and our model.

The above visual comparisons demonstrate that our method preserves better rendering quality while containing fewer artifacts.

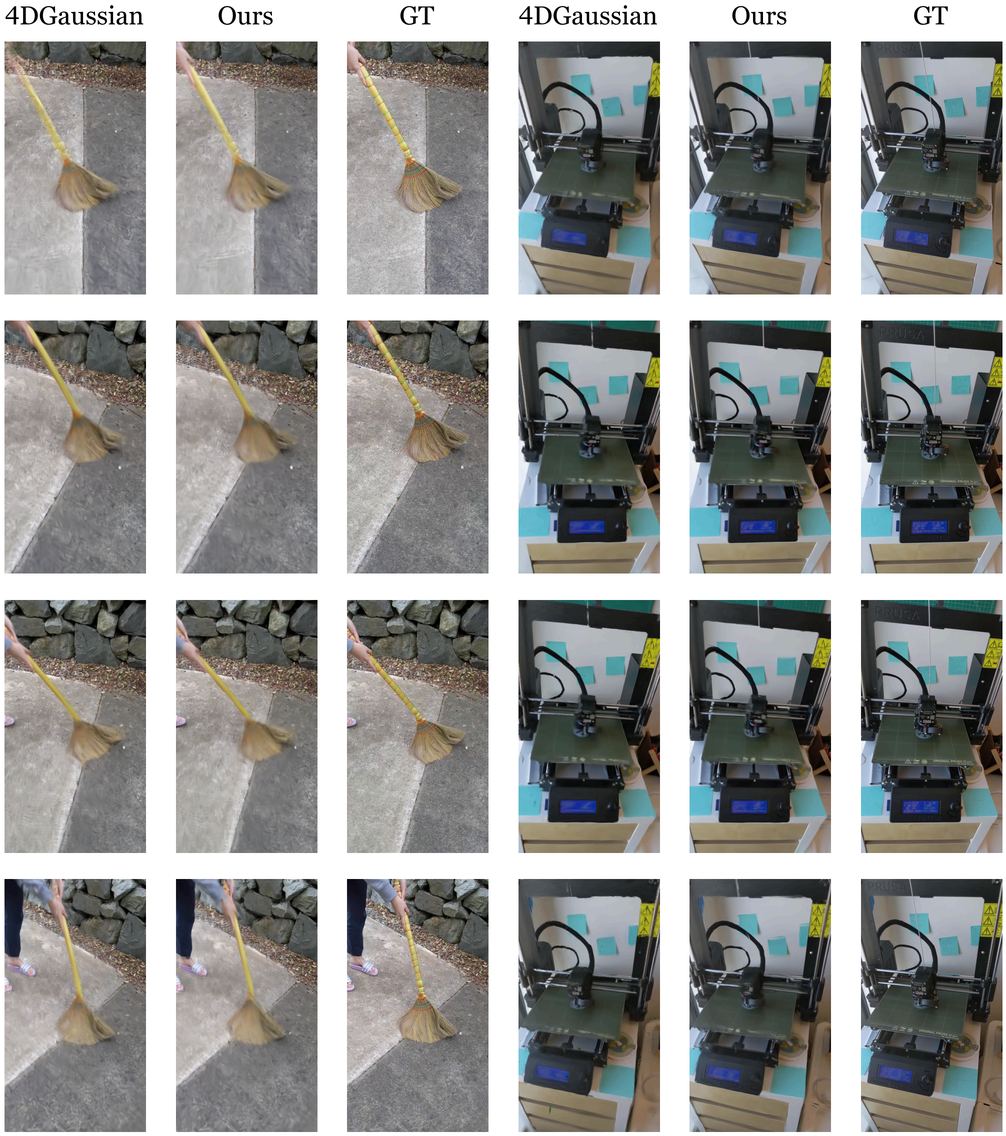

Figure 11: **Qualitative comparisons on HyperNeRF Dataset.**

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We accurately reflect the contributions and scope in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.

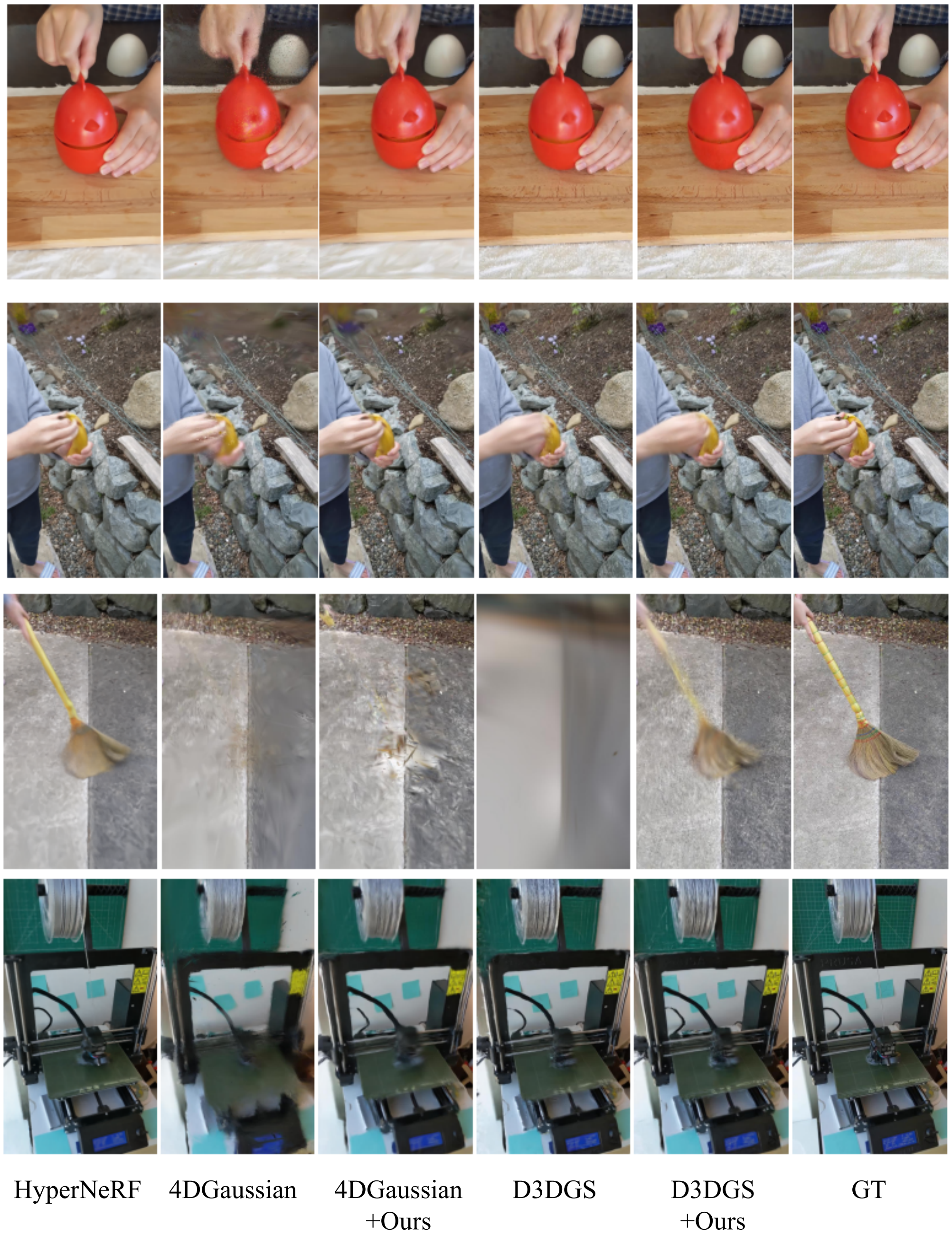

Figure 12: **Qualitative comparisons on HyperNeRF Dataset.**

- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to Section 5.4 in the main text.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.

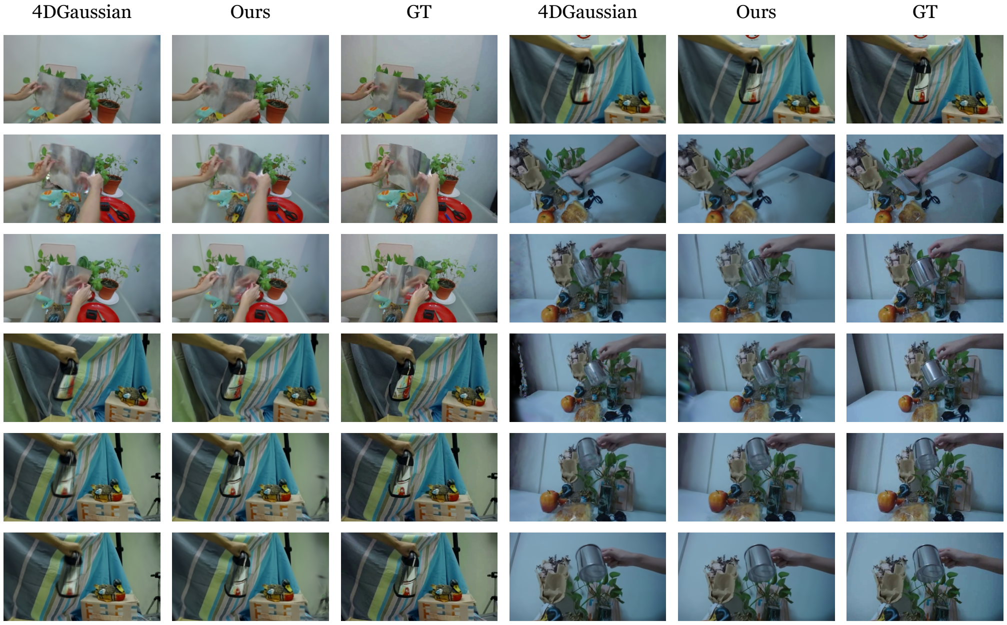

Figure 13: **Qualitative comparisons on NeRF-DS Dataset.**

- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Please refer to Section 3.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.

- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 outlines the structure, while Section 5 delves into the specific experimental settings. For additional details, please refer to the supplemental material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The data are public datasets and we are going to open-source our code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the specific experimental settings in Section 5. More details can be acquired in supplemental material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We use the same datasets and evaluation as previous methods, making this a fair comparison.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide them in Section 5.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our work is solely intended for academic research purposes.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly credit the related assets and explicitly mention the license and terms of use. Please refer to Section A.7 in the supplemental material.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.