arXiv:2411.11941v2 [cs.CV] 6 Oct 2025

# TimeFormer: Capturing Temporal Relationships of Deformable 3D Gaussians for Robust Reconstruction

Dadong Jiang[1,3] Zhihui Ke[1] Xiaobo Zhou[1†] Zhi Hou[2†] Xianghui Yang[3]
Wenbo Hu[4] Qiu Tie[1] Chunchao Guo[3]
[1] Tianjin University [2] Shanghai Artificial Intelligence Laboratory
[3] Tencent Hunyuan [4] Tencent AI Lab


Figure 1. We propose **TimeFormer**, a Transformer module that implicitly models the motion pattern via Temporal Attention from a learning perspective (right). TimeFormer is plug-and-play to existing deformable 3D Gaussian reconstruction methods [27, 61, 67] and enhances reconstruction results (left and middle) without affecting their original inference speed.

## Abstract

*Dynamic scene reconstruction is a long-term challenge in 3D vision. Recent methods extend 3D Gaussian Splatting to dynamic scenes via additional deformation fields and apply explicit constraints like motion flow to guide the deformation. However, they learn motion changes from individual timestamps independently, making it challenging to reconstruct complex scenes, particularly when dealing with violent movement, extreme-shaped geometries, or reflective surfaces. To address the above issue, we design a plug-and-play module called TimeFormer to enable existing deformable 3D Gaussians reconstruction methods with the ability to implicitly model motion patterns from a learning perspective. Specifically, TimeFormer includes a Cross-Temporal Transformer Encoder, which adaptively learns the temporal relationships of deformable 3D Gaussians. Furthermore, we propose a two-stream optimization strategy that transfers the motion knowledge learned from TimeFormer to the base stream during the training phase. This allows us to remove TimeFormer during inference, thereby preserving the original rendering speed. Extensive experiments in the multi-view and monocular dynamic scenes validate qualitative and quantitative improvement brought by TimeFormer. Project Page: https://patrickddj.github.io/TimeFormer/*

† Corresponding Author

## 1. Introduction

High-quality reconstruction of dynamic scenes is significantly challenging in computer vision and graphics, yet has a wide range of potential applications in movie production, virtual reality, and augmented reality. The difficulty stems from factors like occlusions, translucent materials, specular surfaces, and changing topology, all of which are prevalent in dynamic scenes.

Inspired by the success of neural radiance field (NeRF)

on static scenes [3, 36, 37], extending NeRF to dynamic scenes [15, 24, 26, 30, 38, 39, 51, 72] has been explored. However, these NeRF-based methods are limited by computationally intensive volume rendering [7], which makes real-time rendering almost impossible. Recently, 3D Gaussian Splatting (3DGS) [22] represents the scene with anisotropic 3D Gaussians and develops a rasterization-based rendering algorithm, which allows real-time rendering through directly projecting 3D Gaussians onto the image plane. Following the revolutionized 3DGS, dynamic Gaussian Splatting methods [9, 27, 49, 68] have been introduced for reconstructing dynamic scenes. These methods typically construct a canonical 3DGS and utilize a deformation field to deform it based on individual timestamps [27, 61, 67].

However, temporal relationships of 3D Gaussians have been poorly investigated. Previous studies have primarily modeled motion patterns using various types of the deformation field, for example, MLPs [2, 19, 33, 34, 44, 54, 67, 71], spatial-temporal planes [10, 32, 61], polynomial functions [27], Fourier series [21], and combinations of these methods [28]. These methods learn motion patterns from independent time input in a vanilla way, neglecting internal cross-time relationships. Some studies introduce the motion flow regularization [23] to explicitly learn motion patterns from neighboring frames [32, 34]. Although these methods promote similar motion between adjacent timestamps, they adopt a local perspective on time series during optimization. This constraint makes it challenging to reconstruct scenes with more complex motion patterns, such as sudden appearances, violent movements, or reflective surfaces.

In this paper, we introduce TimeFormer, a transformer module designed to implicitly learn motion patterns across multiple timestamps. By modeling temporal relationships within a time batch, we aim to provide a global view of the entire time series, enabling the deformation field backbones themselves to capture motion patterns from a learning perspective, as shown in Fig. 1. Specifically, TimeFormer utilizes a Cross-Temporal Encoder to capture the implicit motion patterns of 3D Gaussians across multiple sampled timestamps using a self-attention mechanism. Moreover, to avoid additional computational costs of TimeFormer during inference, we present a two-stream optimization strategy. By sharing the weights of two deformation fields, we transfer the motion pattern learning from TimeFormer stream to the base stream during training. Therefore, we can eliminate TimeFormer during inference and maintain the original rendering speed. Notably, TimeFormer does not require any prior information and extracts motion patterns solely from RGB supervision, which can be seamlessly adapted to previous deformable 3D Gaussian methods in a plug-and-play manner. Furthermore, TimeFormer promotes faster gradient descent and guides a more efficient canonical space, ultimately increasing FPS during inference, as demonstrated in Fig. 2.

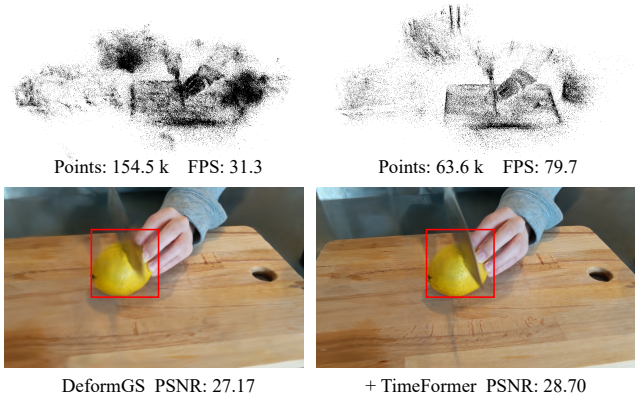



Figure 2. TimeFormer guides towards more efficiently distributed canonical space, showing higher FPS and better quality. The results are from "cut lemon" in the HyperNeRF Dataset [39].

To sum up, our main contributions are as follows:

- We propose TimeFormer, a transformer module that enhances current deformable 3D Gaussians reconstruction methods in a plug-and-play manner from an automatic learning perspective.
- The two-stream optimization strategy allows the exclusion of TimeFormer during inference while maintaining and even improving rendering speed.
- Extensive experiments on real-world datasets validate the effectiveness of TimeFormer, achieving state-of-the-art rendering quality.

## 2. Related Works

### 2.1. Dynamic Scene Reconstruction

Dynamic Scene Reconstruction has been extensively researched over many years, with a wealth of studies [20, 39, 66, 67, 74] contributing to current progress. The pioneering neural radiance field [3, 12, 36, 48, 69] has demonstrated photorealistic rendering for novel view synthesis from calibrated multiview images, inspiring extensive approaches [8, 11, 38, 39, 42, 52, 63] to extend NeRF to 4D space-time field for dynamic scene reconstruction. There are majorly two lines of NeRF-based methods for dynamic scene reconstruction: 1) deformation-based methods [11, 38, 39, 42, 52] model deformation changes by using a deformation field to map the queried positions in different timestamps to a canonical space; 2) compact 4D space-time fields [5, 8, 25, 63] take the timestamps, positions, and directions as input to predict color and density. However, NeRF-based approaches are limited to training and rendering speed. Thus, a large number of methods [1, 4, 13, 24, 29, 31, 43, 47, 56–58, 62, 64] have been proposed to accelerate NeRF-based methods.

Recent 3DGS [22] achieves real-time rendering and dy-

namic 3DGS approaches [27, 67, 68] emerge. In details, deformation-based methods [2, 10, 19, 21, 27, 28, 32–34, 44, 54, 61, 67, 71] employ a deformation field to predict per-Gaussian offsets and deform a canonical 3DGS according to different timestamps. Meanwhile, 4DGS [68] and 4D-Rotor [9] add the time dimension to 3D Gaussian, forming a 4D Gaussian representation. TimeFormer focuses on modeling the temporal relationship and is plug-and-play to deformation-based methods [67] and 4D space-time representations [27, 68]. There are also a few attempts [28, 44, 49] in leveraging the nearby motion or flow for dynamic Gaussian reconstruction. For example, 3DGStream [49] proposes a per-frame optimization strategy, which predicts current Gaussian attributes based on previous Gaussian attributes. Meanwhile, Gaussian-Flow [28] presents explicitly model time-dependent residual of each attribute. To model a long video sequence, SWinGS [44] splits a video sequence into multiple sliding windows based on optical flow and uses a deformation-based method to model each sliding window.

### 2.2. Motion Modeling

Motion prediction in dynamic scene reconstruction from multi-view videos or monocular videos is an ill-posed problem. Early dynamic NeRF [15, 38, 39, 52] directly learns motion patterns from individual timestamps , and several later works propose motion flow regularization terms [15, 26, 30, 35, 40, 45, 46, 51, 55, 70] to promote the learning of cross-time motion patterns. These methods typically utilize 2D prior information (*e.g.*, optical flow) from pre-trained networks [50] to supervise the scene flows within neighboring frames. PREF [46] uses motion predictor to infer current motion based on the previous four frames and propose a self-supervision strategy without optical flow prior. KFD-NeRF [70] models motion patterns by Kalman Filter based on the previous two frames. CT-NeRF [35] employs a cross-attention mechanism from transformer [53] to learn the correlation of conservative frames.

Dynamic 3D Gaussian Splatting (3DGS) methods [16, 17, 59, 73] also utilize optical flow supervision to enhance reconstruction performance. Techniques such as MD-Splatting [10], D3DG [34], and ST-4DGS [23] focus on minimizing both forward and backward Gaussian flow to promote temporal smoothness in Gaussian motions. DN-4DGS [32] incorporates information from previous, current, and next timestamps into the deformation field to capture cross-time motion patterns. However, these methods primarily establish temporal correlations with neighboring timestamps , which limits their ability to address long-term motion changes. Additionally, they introduce extra computational costs during inference, which can reduce rendering speed. In contrast, TimeFormer captures the temporal relationships of Gaussians from a global perspective across the entire time series. Notably, TimeFormer is employed only during the training phase, without any additional computational costs during inference.

## 3. Preliminary: Deformable 3D Gaussians

3D Gaussians [22] represent the scene with a set of 3D Gaussians, each of which has unique opacity $o \in [0, 1]$, center position $\mu \in \mathbb{R}^{3\times1}$, and covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$. For a position $x \in \mathbb{R}^{3\times1}$ in 3D space, the corresponding contribution of a 3D Gaussian on it can be formulated as:

$$G(x) = o \cdot e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}. \tag{1}$$

The covariance matrix $\Sigma$ can be decomposed into a scaling matrix $\mathbf{S}$ and a rotation matrix $\mathbf{R}$: $\Sigma = \mathbf{RSS}^T\mathbf{R}^T$, where $\mathbf{S} = \text{diag}([s_x, s_y, s_z])$ and $\mathbf{R}$ can be transformed from a quaternion $[r_w, r_x, r_y, r_z]$. Then the 3D Gaussians can be splatted to a 2D camera plane through differential Gaussian splatting. To model the appearance of 3D Gaussians, spherical harmonics (SH) are introduced to define the color $c$. Finally, for each pixel, the rendering results of 3DGS can be derived by calculating the color contribution of all the related Gaussians. This process is known as $\alpha$-blending:

$$C = \sum_{i}^{N} c_i \alpha_i \prod_{j=1}^{i-1}(1 - \alpha_j), \tag{2}$$

where $c_i$, $\alpha_i$ represent the color and density computed from the $i$-th 3D Gaussian.

Recent methods [27, 61, 67] utilize deformation fields that extend 3DGS to 4D space, inspired from NeRF-based methods such as D-NeRF [42]. The structure of the deformation field $\mathcal{D}$ can vary among MLP [67], K-Plane [61] and Polynominal [27], while the deformation process can be summarized as follows:

$$(\Delta\mu, \Delta r, \Delta s) = \mathcal{D}(\mu, t), \tag{3}$$

where timestamp $t \in \mathcal{T}$, $\mathcal{T} \in \mathbb{R}^{T\times1}$ contains $T$ linear time inputs, $\mu, r, s$ are the center position, rotation quaternion, and scaling factors of 3D Gaussians, and $\Delta\mu, \Delta r, \Delta s$ are their residuals, respectively. The inputs and outputs of Eq. 3 can vary among different methods, while they share the same framework.

## 4. Method

In this section, we first provide an overview of the enhanced deformable 3D Gaussians reconstruction with the proposed TimeFormer (Sec. 4.1). We then introduce the TimeFormer in detail (Sec. 4.2) which consists of a Cross-Temporal Encoder and shared deformation fields (Sec. 4.3). We also discuss the insights behind TimeFormer from the view of gradient flow.

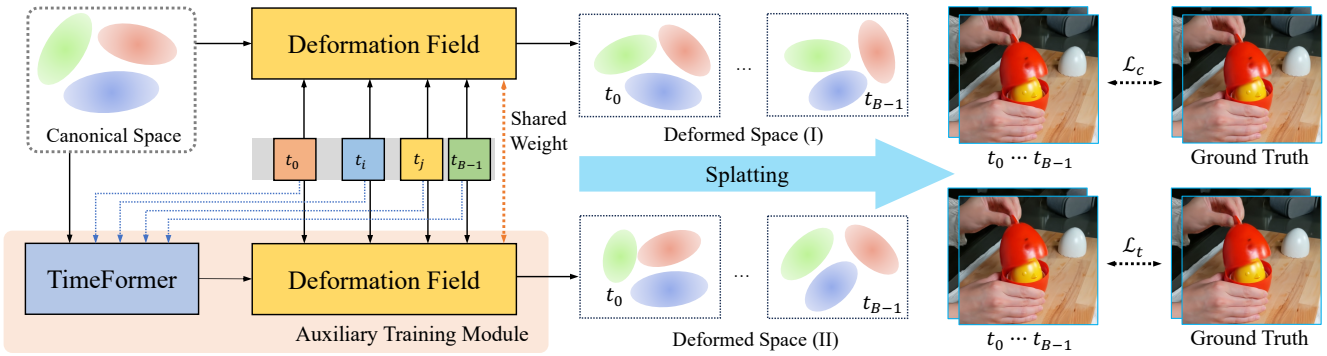



Figure 3. The Framework of Deformable 3D Gaussians Reconstruction with TimeFormer. Existing deformable 3D Gaussians framework usually includes the canonical space and the deformation field (first row), we incorporate TimeFormer to capture cross-time relationships and explore motion patterns implicitly (second row). We share weights of two deformation fields to transfer the learned motion knowledge. This allows us to **exclude this Auxiliary Training Module** during inference.

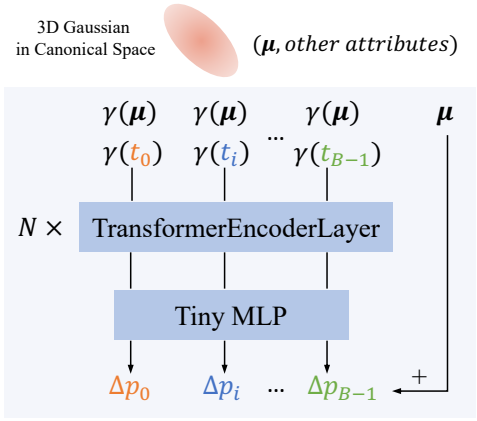



Figure 4. The Structure of Cross-Temporal Encoder. We concatenate randomly sampled timestamps to position $\mu$, treating them as special *tokens* in a sequence. This module is designed to model multi-temporal relationships and produce distinct time-variant position offsets $\Delta p_0, \ldots, \Delta p_{B-1}$.

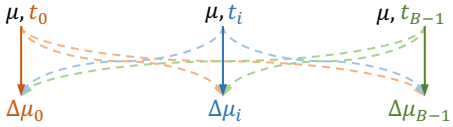


Figure 5. Data Flow Changes in the Deformation Field. Dashed lines represent new data flow among time samples $t_0, \ldots, t_{B-1}$.

### 4.1. Overview

Previous methods model motion patterns by explicitly learning temporal relationships on individual or neighboring timestamps, failing on those complex scenes containing violent movement or dynamic reflective surfaces. In contrast, we present TimeFormer to enable the deformable 3D Gaussian backbones themselves to model cross-time relationships from an implicit learning perspective. The main framework with the proposed TimeFormer is shown in Fig. 3. Our approach retains standard reconstruction modules, which include (1) 3D Gaussians in the canonical space and (2) a deformation field that applies time-variant transformation. Additionally, TimeFormer is introduced before the deformation field to extract implicit cross-time motion features for each Gaussian through a self-attention mechanism along the time dimension. Moreover, we share the weights of two deformation fields to transfer the motion knowledge from TimeFormer to mitigate the gap between the original branch and TimeFormer branch, which supports real-time rendering without TimeFormer during inference.

### 4.2. TimeFormer

**Cross-Temporal Encoder** BatchFormer [18] demonstrates that the attention mechanism helps learn sample relationships from batch dimension, rather than channel and spatial dimentions [6, 53], inspiring us that different timestamps can also be considered as a special time batch. Let $\mathcal{T}_s \subset \mathcal{T}, \mathcal{T}_s = \{t_i\}_{i=0}^{B-1}$ denotes randomly sampled timestamps, and let $\mathcal{G} \in \mathbb{R}^{N\times(3+C)}$ denotes Gaussians in the canonical space, where $B$ is the size of time batch, $N$ is the number of Gaussians, $3+C$ means each Gaussian has 3 position channels and $C$ additional channels. As in Fig. 4, all Gaussians's postions are made into $B$ copies, expanded into $\mathcal{G}_c \in \mathbb{R}^{B\times N\times 3}$, and sampled timestamps are made into $N$ copies, expanded into $\mathcal{T}' \in \mathbb{R}^{B\times N\times 1}$. Then, we composite $\mathcal{G}_c$ and $\mathcal{T}'$ together and apply position encoding function $\gamma$ to extract high frequency information, as in Eq. 4:

$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots \right. \\ \left. \sin\left(2^{L-1}\pi p\right), \cos\left(2^{L-1}\pi p\right)\right) \tag{4}$$

In our experiments, we set $L = 6$ for both positions $x$ and time $t$. We treat $[\gamma(\mathcal{G}_c, \gamma(\mathcal{T}')] \in \mathbb{R}^{B\times N\times(3\times 2L+2L)}$ as the original input $F_0$ to TimeFormer. It can be considered as $N$ sequences of the length $B$, containing $8L$ feature channels. With $M$ transformer encoder layers, for $m^{th}$ layer, intermediate features are calculated through multi-head self-attention(MSA) and MLP blocks. In the final stage, we use a tiny MLP to transform the last encoded features $F_{M-1}$

into offset $\mathcal{O} \in \mathbb{R}^{B\times N\times 3}$ in the linear space:

$$\mathcal{O} = MLP(F_{M-1}), \quad \mathcal{G}_t = \mathcal{G}_c + \mathcal{O} \tag{5}$$

We consider the output from the Cross-Temporal Encoder as a fixing residual term to the original positions to encourage a gradual, steady learning process on motion patterns. Implicit cross-time relationship learning in TimeFormer enables the automatic aggregation of Gaussians with similar variations during optimization, promoting more efficient spatial distribution and accelerating rendering speed.

**Gradient Analysis** Let $\mathcal{D}$ and $\mathcal{P}$ be the deformation field and TimeFormer respectively, the output of the deformation field $\Delta\mu_i$ and partial derivative for $\mu$ in time $t_i \in \mathcal{T}_s$ are formulated as follows:

$$\Delta\mu_i = \mathcal{D}(\mu, t_i), \quad \frac{\partial \Delta\mu_i}{\partial \mu} = \frac{\partial \mathcal{D}(\mu, t_i)}{\partial \mu} \tag{6}$$

With TimeFormer applied on time batch $\mathcal{T}_s$, we reformulate as Eq. 7 and data flow changes as in Fig. 5.

$$\Delta\mu_i = \mathcal{D}(\mu + \mathcal{P}(\mu, \mathcal{T}_s), t_i) \tag{7}$$

To calculate the partial derivative for $\mu$, we apply the chain rule. Let $a = \mu + \mathcal{P}(\mu, \mathcal{T}_s)$, then:

$$\frac{\partial \Delta\mu_i}{\partial \mu} = \frac{\partial \mathcal{D}(a, t_i)}{\partial a} \cdot \frac{\partial a}{\partial \mu}, \quad \frac{\partial a}{\partial \mu} = 1 + \frac{\partial \mathcal{P}(\mu, \mathcal{T}_s)}{\partial \mu} \tag{8}$$

$$\frac{\partial \Delta\mu_i}{\partial \mu} = \frac{\partial \mathcal{D}(a, t_i)}{\partial a} \cdot \left(1 + \frac{\partial \mathcal{P}(\mu, \mathcal{T}_s)}{\partial \mu}\right) \tag{9}$$

Eq. 6 has a weaker dynamic nature, as it only considers the current timestamp $t_i$. In contrast, Eq. 9 incorporates an additional gradient term, $\frac{\partial \mathcal{P}(\mu, \mathcal{T}_s)}{\partial \mu}$, enabling the current state to be influenced by any past or future states. In other words, $\Delta\mu_i$ also optimizes the models according to other timestamps $t_j \in \mathcal{T}_s (j \neq i)$, which is a significant difference compared to the backward process without TimeFormer. Such design accounts for cross-time attention and allows the models to capture more challenging motion patterns from a global view of the entire time series.

### 4.3. Shared Deformation Fields

Accounting for additional computation costs of TimeFormer, which can significantly decrease rendering speed, we force the original deformation field and the auxiliary deformation field to share weights for knowledge transferring. Let $\mathcal{V}$ be the camera viewpoints, we apply the shared deformation fields to predict deformed space from both Gaussians in the canonical space and Gaussians from TimeFormer. Then, we apply the splatting algorithm to these two groups of deformed space. We calculate the losses between rendered images and ground truth $\mathcal{I}_{gt}$ as follows:

$$\mathcal{L}_c = \|Splatting(\mathcal{D}(\mathcal{G}_c, \mathcal{T}), \mathcal{V}) - \mathcal{I}_{gt}\|_1 \tag{10}$$

$$\mathcal{L}_t = \|Splatting(\mathcal{D}(\mathcal{G}_t, \mathcal{T}), \mathcal{V}) - \mathcal{I}_{gt}\|_1 \tag{11}$$

$$\mathcal{L} = \lambda_c \mathcal{L}_c + \lambda_t \mathcal{L}_t \tag{12}$$

, where $\mathcal{L}_c, \mathcal{L}_t$ represent losses of original branch and TimeFormer branch with $\lambda_c > \lambda_t$. We use a relatively smaller $\lambda_t$ because we find it easy to overfit on the second branch with TimeFormer, causing a degradation in inference quality.

## 5. Experiment

### 5.1. Implementation Details

Our implementation is tested on a single A100 GPU. We use $M = 4$ transformer encoder layers in TimeFormer and the number of time samples is $B = 4$. We set $\lambda_c = 1$ and $\lambda_t = 0.8$ to prevent overfitting on TimeFormer. We assess our experimental results using image quality metrics, including peak-signal-to-noise ratio (PSNR) and structural similarity index (SSIM [60]).

**Baselines & Datasets.** We apply TimeFormer to 4DGS [68], STGS [27] on multi-view videos, *e.g.*, N3DV Dataset [25], and evaluate TimeFormer on 4DGS [68] and DeformGS [67] on monocular dynamic videos, *e.g.*, HyperNeRF Dataset [39], respectively. We also utilize NeRF-DS Dataset [65] to demonstrate the robustness of TimeFormer on moving objects with specular surfaces.

### 5.2. Experimental Comparisons

**Multi-View Video Dataset.** In Tab. 1, TimeFormer improves the reconstruction quality (*e.g.*, PSNR) of original 4DGS [61] and STGS [27] by 0.74 and 0.61, outperforming all baselines methods. We extract the frames at an initial time for scene *Cook Spanish* in Fig. 6, and TimeFormer can reconstruct a clearer geometry and specular effects compared to the original results.

To have a more comprehensive overview of TimeFormer's capacity on different temporal stages, we collect PSNR for 300 frames of all six scenes from N3DV Dataset [25] and calculate the average per-frame PSNR. As in Fig. 7, the reconstruction quality increases more significantly at the beginning and end, and such balanced reconstruction results are attributed to TimeFormer's cross-temporal attention mechanism on the whole time series.

**Monocular Video Dataset.** Tab. 3 shows the enhancements achieved by TimeFormer on the HyperNeRF Dataset. It's observed that TimeFormer increases the PSNR by 0.89 for 4DGS [61] and 0.94 for DeformGS [67], and the SSIM by 0.019 and 0.028, respectively. Fig. 9 additionally provides visualization of the improvement in the image quality: on the left, TimeFormer generates a clearer outline between fingers, while on the right the texture on the front of the broom is clearer. TimeFormer also eliminates artifacts and reveals a clear structure of the broom.

**Dynamic Reflective Video Dataset.** Fig. 10 additionally demonstrates that TimeFormer can work well on scenes

| Method | Sear Steak | | Flame Salmon | | Cut Beef | | Flame Steak | | Cook Spinach | | Coffee Martini | | Mean | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| K-Plane [14] | 32.52 | 0.971 | 30.44 | 0.942 | 31.82 | 0.965 | 32.38 | 0.970 | 30.60 | 0.968 | 29.99 | 0.943 | 31.63 | 0.960 |
| MixVovels [56] | 31.21 | 0.971 | 29.92 | 0.945 | 31.30 | 0.965 | 31.43 | 0.970 | 31.61 | 0.965 | 29.36 | 0.946 | 30.80 | 0.960 |
| GS4D [68] | 32.92 | 0.953 | 26.39 | 0.897 | 33.08 | 0.959 | 33.81 | 0.967 | 32.77 | 0.956 | 25.23 | 0.884 | 30.07 | 0.936 |
| 4D-Rotor [9] | 32.86 | 0.956 | 28.25 | 0.913 | 33.14 | 0.952 | 31.61 | 0.953 | 32.56 | 0.949 | 27.95 | 0.908 | 31.06 | 0.938 |
| 4DGS [61] | 32.49 | 0.949 | 28.92 | 0.917 | 32.90 | 0.956 | 32.51 | 0.954 | 32.46 | 0.948 | 27.34 | 0.903 | 31.10 | 0.938 |
| +TimeFormer | **33.38** | **0.955** | **29.33** | **0.924** | **33.11** | **0.957** | **33.25** | **0.953** | **33.03** | **0.949** | **28.93** | **0.910** | **31.84** | **0.941** |
| STGS [27] | 33.71 | 0.962 | 28.21 | 0.921 | 33.52 | 0.958 | 33.46 | 0.963 | 33.13 | 0.955 | 27.71 | 0.915 | 31.62 | 0.946 |
| +TimeFormer | **34.34** | **0.965** | **29.13** | **0.924** | **33.57** | **0.958** | **34.04** | **0.964** | **33.45** | **0.956** | **28.83** | **0.917** | **32.23** | **0.947** |

Table 1. Quantitative Comparisons on N3DV Dataset [25]. We use **bold font** to indicate the improvement, statistics of K-Plane [14] and MixVoxels [56] are from their original paper, while we calculate metrics of all Gaussian-based methods by running their official codes.


Figure 6. Visualization of Comparisons on N3DV Dataset [25].

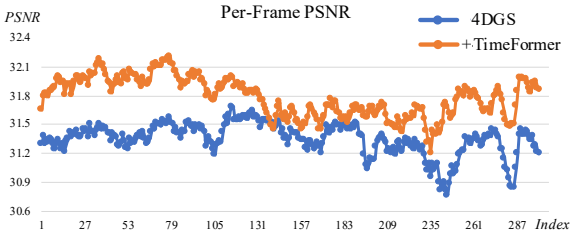



Figure 7. Per-Frame PSNR on N3DV Dataset [25]. TimeFormer improves the reconstruction quality, especially at the beginning and end time series.

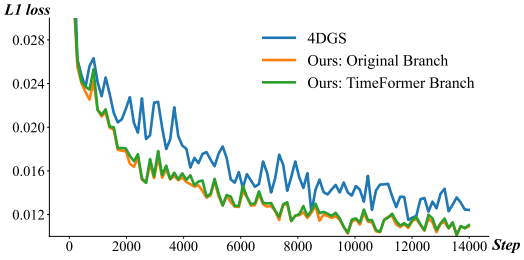



Figure 8. Comparions of Convergence Speed. We use the same batch size for 4DGS [61] and calculate the loss of both branches in our method.

with dynamic specular objects on NeRF-DS Dataset [65]. TimeFormer can reconstruct a cleaner appearance on reflective material (*e.g.*, a metallic cup, a glass bottle), and this improvement is attributed to the attention mechanism which enables the deformation field to detect changes on specular surfaces from a learning perspective automatically.

**Analysis of FPS & Canonical Space.** We also find that TimeFormer achieves more efficient representation with much fewer points in the final canonical space, thereby achieving a higher FPS in Tab. 2. We attribute the phenomenon to the reason that TimeFormer guides faster gradient descent and promotes the spatial layout of Gaussian points during optimization. Interestingly, in Fig. 8, the L1 loss of both branches in our method decreases faster and shows almost the same convergence trend during training, exceeding the optimization speed of the original method. This eliminates many redundant points and guides the canonical space toward a more efficient distribution, as in Fig. 2. Moreover, Fig. 8 also proves that the cross-time relationship learned in TimeFormer has been successfully transferred to the base branch, as introduced in Sec. 4.3.

**Analysis of Motion Patterns.** We argue that TimeFormer achieves more robust learning of motion patterns. To take a deeper insight, we visualize the motion changes as fol-

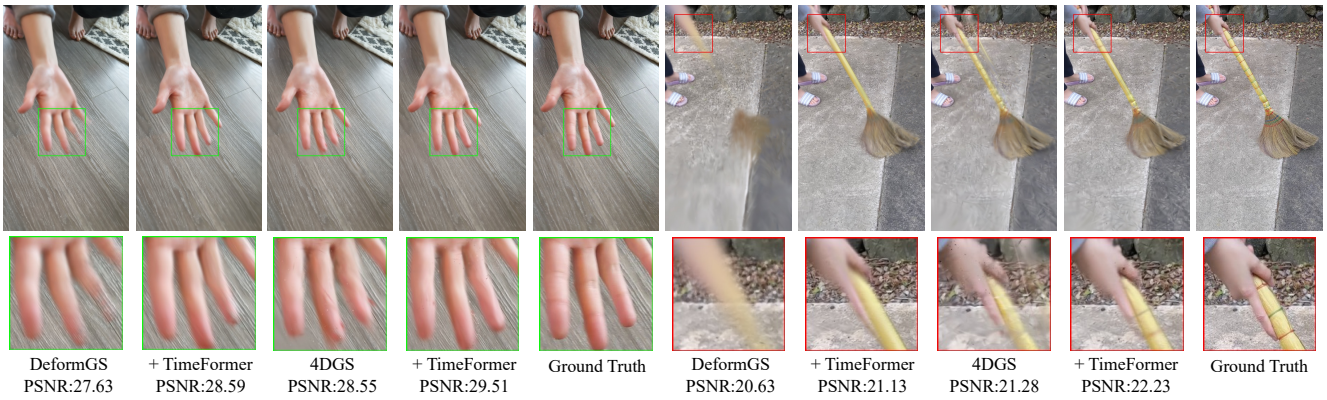



Figure 9. Visualization of Comparisons on HyperNeRF Dataset [39].

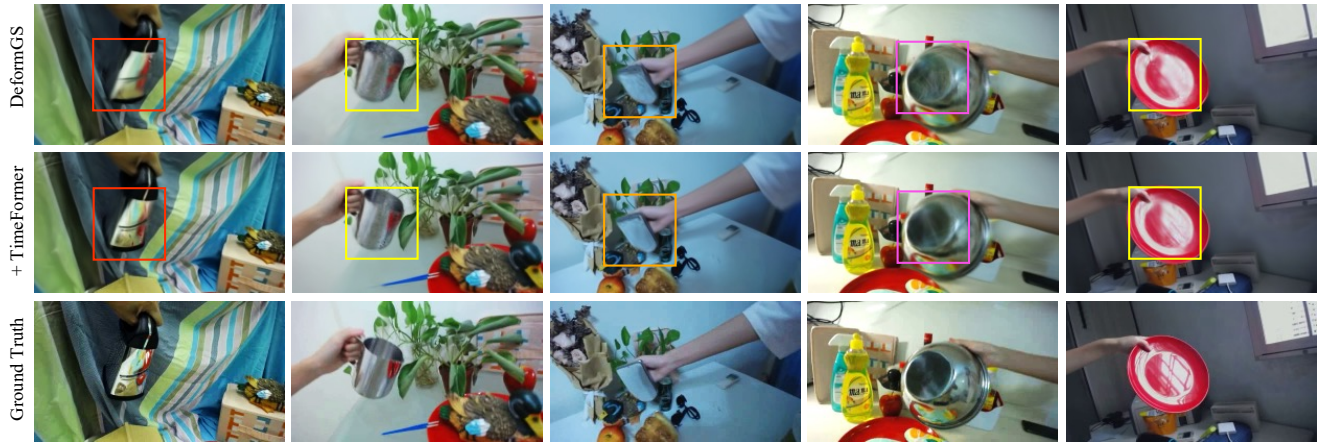



Figure 10. Visualization of Comparisons on NeRF-DS Dataset [65]. Compared with original results from DeformGS [67], TimeFormer presents a clearer visual effect on dynamic objects with specular surfaces.

| Method | $N\downarrow$ | Training $\downarrow$ | FPS $\uparrow$ |
|---|---|---|---|
| STGS [27] | 172.1 k | **41 min** | 82.1 |
| +TimeFormer | **148.9 k** | 78 min | **88.9** |
| 4DGS [61] | 145.9 k | **52 min** | 31.7 |
| +TimeFormer | **113.1 k** | 94 min | **37.9** |

(a) Quantitative comparisons on N3DV Dataset [25].

| Method | $N\downarrow$ | Training $\downarrow$ | FPS $\uparrow$ |
|---|---|---|---|
| DeformGS [67] | 169.6 k | **25 min** | 30.1 |
| +TimeFormer | **82.9 k** | 35 min | **58.9** |
| 4DGS [61] | 172.3 k | **32 min** | 35.7 |
| +TimeFormer | **135.5 k** | 48 min | **40.9** |

(b) Quantitative comparisons on HyperNeRF Dataset [39].

Table 2. Comparisons of Gaussian Number, Training Time & FPS.

lows: for each Gaussian, we calculate its bias $(\Delta_x, \Delta_y, \Delta_z)$ towards canonical space in time $t$ and replace the color attributes $(r, g, b)$ with the absolute value of this bias: $(r, g, b) \leftarrow (|\Delta_x|, |\Delta_y|, |\Delta_z|)$. We use the same splatting algorithm to render the accumulated motion bias, transferring the rendered result as a heatmap, as in Fig. 11.

In Fig. 11a, TimeFormer identifies the blade's motion as a rigid deformation, showing a sharper outline and consistent motion, especially as it contacts the initially static lemon. This demonstrates TimeFormer 's ability to distinguish between moving and static objects. In Fig. 11b, TimeFormer captures subtle motion on the spray gun and clear flame flickering on the beef. Fig. 11c shows that TimeFormer also detects the violent movement of the elongated broom, where 4DGS [61] fails, proving its robustness in handling extremely geometries.

### 5.3. Ablation Studies

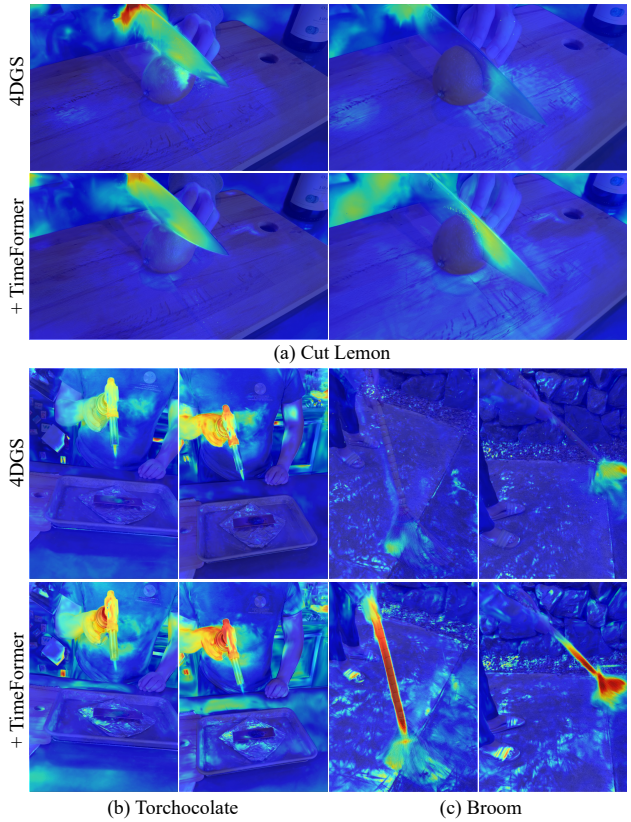
**Cross-Temporal Encoder.** Tab. 5 demonstrates that the performance of TimeFormer is not sensitive to time batch $B$ and transformer encoder layer $M$. In Tab. 5b, we explore two sampling methods: random sampling and continuous sampling of timestamps. Both strategies demonstrate improvements in PSNR and other metrics compared to the baseline. Notably, the random sampling method yields

| Method | Broom | | Chicken | | Cut Lemon | | Torchco | | Peel Banana | | Hand | | Mean | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| DeformGS [67] | 20.74 | 0.322 | 26.32 | 0.786 | 27.94 | 0.714 | 27.4 | 0.877 | 26.38 | 0.836 | 27.79 | 0.78 | 26.09 | 0.719 |
| +TimeFormer | **21.01** | **0.324** | **26.55** | **0.792** | **30.01** | **0.780** | **27.55** | **0.883** | **27.24** | **0.852** | **29.79** | **0.848** | **27.03** | **0.747** |
| 4DGS[61] | 21.53 | 0.351 | 26.82 | 0.797 | 29.72 | 0.763 | 27.45 | 0.883 | 27.82 | 0.844 | 29.52 | 0.841 | 27.14 | 0.747 |
| +TimeFormer | **22.84** | **0.401** | **27.06** | **0.801** | **30.23** | **0.778** | **29.48** | **0.902** | **28.21** | **0.853** | **30.38** | **0.862** | **28.03** | **0.766** |

Table 3. Quantitative Comparisons on HyperNeRF Dataset [39].

| Method | Press | | Plate | | Basin | | Sieve | | Bell | | Cup | | Mean | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
| HyperNeRF [39] | 25.4 | 0.873 | 18.1 | 0.714 | 20.2 | 0.829 | 25.0 | 0.909 | 24.0 | 0.884 | 24.1 | 0.896 | 22.8 | 0.851 |
| NeRF-DS [65] | 26.4 | 0.911 | 20.8 | 0.867 | 20.3 | 0.868 | 26.1 | 0.935 | 23.3 | 0.872 | 24.5 | 0.916 | 23.57 | 0.895 |
| DeformGS[67] | 25.68 | 0.866 | 20.82 | 0.812 | 19.87 | 0.804 | 25.71 | 0.881 | 24.92 | 0.854 | 24.52 | 0.897 | 23.59 | 0.852 |
| +TimeFormer | **26.29** | **0.867** | **20.90** | **0.817** | **19.93** | **0.805** | **26.26** | **0.891** | **25.90** | **0.873** | **25.16** | **0.903** | **24.10** | **0.859** |

Table 4. Quantitative Comparisons on NeRF-DS Dataset [65].




(a) Cut Lemon

(b) Torchocolate (c) Broom

Figure 11. Motion Visualization on HyperNeRF Dataset [39].

more significant enhancements. We hypothesize that this phenomenon occurs because the random sampling method allows for more effective propagation of dynamic behavior modeling across the entire time series.

**Shared Deformation Fields.** In Tab. 5b, the image quality during inference suffers a considerable decrease if we train two deformation fields without sharing their weights. Fig. 8 also demonstrates the effectiveness of the two-stream strategy and TimeFormer's ability of re-balancing.

| Setting | Random Sampling | | | Continuous Sampling | | |
|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ |
| $B$=2 | 25.89 | 0.875 | 0.117 | 25.73 | 0.871 | 0.124 |
| $B$=4 | 26.06 | 0.871 | 0.119 | 25.78 | 0.874 | 0.117 |
| $B$=6 | 25.99 | 0.87 | 0.123 | 25.73 | 0.865 | 0.125 |

(a) Ablation Studies on Batch Size and Sampling Strategies.

| Setting | PSNR↑ | SSIM↑ | LPIPS↓ |
|---|---|---|---|
| Baseline | 25.44 | 0.867 | 0.125 |
| M=1 | 25.96 | 0.874 | 0.117 |
| M=2 | 25.90 | 0.874 | 0.120 |
| M=3 | 26.05 | 0.876 | 0.112 |
| M=4 | 26.04 | 0.873 | 0.119 |
| w/o Shared | 24.81 | 0.852 | 0.132 |

(b) Ablation Results. $M$ is the number of encoder layers, "w/o Shared" means not using shared weights in two deformation fields.

Table 5. Ablation Results. The results are from three scenes *press*, *sieve* and *bell* on NeRF-DS Dataset [65].

## 6. Conclusion

We propose TimeFormer, a Transformer module that is plug-and-play to existing deformable 3D Gaussians methods and enhances reconstruction results without additional computational budget. TimeFormer enables deformation fields to implicitly model complex motion patterns from a learning perspective. In addition, we design a two-stream optimization strategy to transfer the learned motion knowledge from TimeFormer to the original deformation branch. This allows us to remove TimeFormer during inference and thus maintain the same inference speed as the original meth-

ods. Extensive experiments demonstrate TimeFormer effectively facilitates the reconstruction of three state-of-the-art deformable 3D Gaussians Splatting methods among three datasets, and illustrate the improvement on reconstructing complex scenes containing violent movement, extreme-shaped geometries, or reflective surfaces.

**Limitations** TimeFormer may produce overly smooth textures for objects with intricate details.

## References


[1] 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. 2

[2] 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. 2, 3

[3] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5855–5864, 2021. 2

[4] 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. 2

[5] 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. 2

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 4

[7] Robert A Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *ACM Siggraph Computer Graphics*, 22(4): 65–74, 1988. 2

[8] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 2

[9] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d-rotor gaussian splatting: Towards efficient novel view synthesis for dynamic scenes. In *Proc. SIGGRAPH*, 2024. 2, 3, 6

[10] Bardienus P Duisterhof, Zhao Mandi, Yunchao Yao, Jia-Wei Liu, Mike Zheng Shou, Shuran Song, and Jeffrey Ichnowski. Md-splatting: Learning metric deformation from 4d gaussians in highly deformable scenes. *arXiv preprint arXiv:2312.00583*, 2023. 2, 3

[11] 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. 2

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 2

[13] 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. 2

[14] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, 2023. 6

[15] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 2, 3

[16] Quankai Gao, Qiangeng Xu, Zhe Cao, Ben Mildenhall, Wenchao Ma, Le Chen, Danhang Tang, and Ulrich Neumann. Gaussianflow: Splatting gaussian dynamics for 4d content creation. *arXiv preprint arXiv:2403.12365*, 2024. 3

[17] Zhiyang Guo, Wengang Zhou, Li Li, Min Wang, and Houqiang Li. Motion-aware 3d gaussian splatting for efficient dynamic scene reconstruction. *arXiv preprint arXiv:2403.11447*, 2024. 3

[18] Zhi Hou, Baosheng Yu, and Dacheng Tao. Batchformer: Learning to explore sample relationships for robust representation learning. In *CVPR*, 2022. 4

[19] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4220–4230, 2024. 2, 3

[20] Takeo Kanade, Peter Rander, and PJ Narayanan. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE multimedia*, 4(1):34–47, 1997. 2

[21] Kai Katsumata, Duc Minh Vo, and Hideki Nakayama. A compact dynamic 3d gaussian representation for real-time dynamic view synthesis. In *Computer Vision – ECCV 2024*, pages 394–412. Springer Nature Switzerland, 2025. 2, 3

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 2, 3

[23] Deqi Li, Shi-Sheng Huang, Zhiyuan Lu, Xinran Duan, and Hua Huang. St-4dgs: Spatial-temporally consistent 4d gaussian splatting for efficient dynamic scene rendering. In *ACM SIGGRAPH 2024 Conference Papers*, New York, NY, USA, 2024. Association for Computing Machinery. 2, 3

[24] Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping Tan. Streaming radiance fields for 3d video synthesis. *Advances in Neural Information Processing Systems*, 35: 13485–13498, 2022. 2

[25] 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, 5, 6, 7

[26] 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, 3

[27] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8508–8520, 2024. 1, 2, 3, 5, 6, 7

[28] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21136–21145, 2024. 2, 3

[29] 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. 2

[30] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13–23, 2023. 2, 3

[31] Ange Lou, Benjamin Planche, Zhongpai Gao, Yamin Li, Tianyu Luan, Hao Ding, Terrence Chen, Jack Noble, and Ziyan Wu. Darenerf: Direction-aware representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5031–5042, 2024. 2

[32] Jiahao Lu, Jiacheng Deng, Ruijie Zhu, Yanzhe Liang, Wenfei Yang, Tianzhu Zhang, and Xu Zhou. Dn-4dgs: Denoised deformable network with temporal-spatial aggregation for dynamic scene rendering. *arXiv preprint arXiv:2410.13607*, 2024. 2, 3

[33] Zhicheng Lu, Xiang Guo, Le Hui, Tianrui Chen, Min Yang, Xiao Tang, Feng Zhu, and Yuchao Dai. 3d geometry-aware deformable gaussian splatting for dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8900–8910, 2024. 2

[34] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *2024 International Conference on 3D Vision (3DV)*, pages 800–809. IEEE, 2024. 2, 3

[35] Xingyu Miao, Yang Bai, Haoran Duan, Fan Wan, Yawen Huang, Yang Long, and Yefeng Zheng. Ctnerf: Cross-time transformer for dynamic neural radiance field from monocular video. *Pattern Recognition*, 156:110729, 2024. 3

[36] 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. 2

[37] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2

[38] 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, 3

[39] 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 Trans. Graph.*, 40(6), 2021. 2, 3, 5, 7, 8, 13

[40] Sungheon Park, Minjung Son, Seokhwan Jang, Young Chun Ahn, Ji-Yeon Kim, and Nahyup Kang. Temporal interpolation is all you need for dynamic neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4212–4221, 2023. 3

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019. 12

[42] 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. 2, 3

[43] 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. 2

[44] Richard Shaw, Michal Nazarczuk, Jifei Song, Arthur Moreau, Sibi Catley-Chandar, Helisa Dhamo, and Eduardo Pérez-Pellitero. Swings: sliding windows for dynamic 3d gaussian splatting. In *European Conference on Computer Vision*. Springer, 2024. 2, 3

[45] Nagabhushan Somraj, Kapil Choudhary, Sai Harsha Mupparaju, and Rajiv Soundararajan. Factorized motion fields for fast sparse input dynamic view synthesis. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12, 2024. 3

[46] Liangchen Song, Xuan Gong, Benjamin Planche, Meng Zheng, David Doermann, Junsong Yuan, Terrence Chen, and

Ziyan Wu. Pref: Predictability regularized neural motion fields. In *European Conference on Computer Vision*, pages 664–681. Springer, 2022. 3

[47] 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. 2

[48] 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. 2

[49] Jiakai Sun, Han Jiao, Guangyuan Li, Zhanjie Zhang, Lei Zhao, and Wei Xing. 3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20675–20685, 2024. 2, 3

[50] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 3

[51] Fengrui Tian, Shaoyi Du, and Yueqi Duan. Mononerf: Learning a generalizable dynamic radiance field from monocular videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17903–17913, 2023. 2, 3

[52] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Nonrigid 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, 3

[53] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. 3, 4

[54] Diwen Wan, Ruijie Lu, and Gang Zeng. Superpoint gaussian splatting for real-time high-fidelity dynamic scene reconstruction. In *Forty-first International Conference on Machine Learning*, 2024. 2, 3

[55] Chaoyang Wang, Lachlan Ewen MacDonald, Laszlo A Jeni, and Simon Lucey. Flow supervision for deformable nerf. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21128–21137, 2023. 3

[56] 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. 2, 6

[57] Feng Wang, Zilong Chen, Guokang Wang, Yafei Song, and Huaping Liu. Masked space-time hash encoding for efficient dynamic scene reconstruction. *Advances in Neural Information Processing Systems*, 36, 2024.

[58] 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. 2

[59] Shizun Wang, Xingyi Yang, Qiuhong Shen, Zhenxiang Jiang, and Xinchao Wang. Gflow: Recovering 4d world from monocular video. *arXiv preprint arXiv:2405.18426*, 2024. 3

[60] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. 5

[61] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20310–20320, 2024. 1, 2, 3, 5, 6, 7, 8

[62] Minye Wu, Zehao Wang, Georgios Kouros, and Tinne Tuytelaars. Tetrirf: Temporal tri-plane radiance fields for efficient free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6487–6496, 2024. 2

[63] 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

[64] Zhen Xu, Sida Peng, Haotong Lin, Guangzhao He, Jiaming Sun, Yujun Shen, Hujun Bao, and Xiaowei Zhou. 4k4d: Real-time 4d view synthesis at 4k resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20029–20040, 2024. 2

[65] 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. 5, 6, 7, 8, 13, 14

[66] Jason C Yang, Matthew Everett, Chris Buehler, and Leonard McMillan. A real-time distributed light field camera. *Rendering Techniques*, 2002(77-86):2, 2002. 2

[67] 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. 1, 2, 3, 5, 7, 8

[68] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *International Conference on Learning Representations (ICLR)*, 2024. 2, 3, 5, 6

[69] 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. 2

[70] Yifan Zhan, Zhuoxiao Li, Muyao Niu, Zhihang Zhong, Shohei Nobuhara, Ko Nishino, and Yinqiang Zheng. Kfd-nerf: Rethinking dynamic nerf with kalman filter. *arXiv preprint arXiv:2407.13185*, 2024. 3

[71] Boming Zhao, Yuan Li, Ziyu Sun, Lin Zeng, Yujun Shen, Rui Ma, Yinda Zhang, Hujun Bao, and Zhaopeng Cui. Gaussianprediction: Dynamic 3d gaussian prediction for motion

**Algorithm 1:** Implementation of TimeFormer.

```
class TimeFormer (nn.module):
    # d_in:  here is 4, (x, y, z, t)
    # L: frequency of Position Encoding (PE) γ
    def __init__(self, d_in, L, nhead, d_hidden,
     n_layer):
        # PE: PE function, PE_ch:  d_in×2L
        self.PE, PE_ch = get_PE(L=L, d_in=d_in)
        # define Cross-Temporal Encoder
        layer = nn.TransformerEncoderLayer(PE_ch,
         nhead, d_hidden,
         activation=nn.functional.tanh)
        self.encoder =
         nn.TransformerEncoder(layer, n_layer)
        # define Tiny MLP
        self.mlp = nn.Linear(PE_ch, 3)

    # x:  [seq_len, seq_batch, channel]
    def forward(self, x):
        PE_x = self.PE(x)
        h = self.encoder(PE_x)
        h = self.mlp(h)
        return h
```

**Algorithm 2:** Two-Stream Optimization Strategy.

```
# timeformer:  TimeFormer
# deform:  Shared Deformation Field

# N: number of Gaussians
# GS: Gaussians in the canonical space
# B: size of time batch
B = 4
lambda_t = 0.8
# Vs, Ts:  sampled cameras and timestamps
# images_gt:  sampled GT images
Vs, Ts, images_gt = random.sample(Dataset, B)

# construct input to TimeFormer
# [N, 3] => [B, N, 3]
G_expanded = GS.xyz.unsqueeze(0).expand(B, -1, -1)
# [B] => [B, N, 1]
T_expanded = Ts.unsqueeze(1).expand(-1,
 N).unsqueeze(2)
# src:  [B(seq_len), N(seq_batch), 4(channel)]
src = torch.cat((G_expanded, T_expanded), dim=2)
# offset_t:  [B, N, 3]
offset_t = timeformer(src)

# use iterations to save cuda memory
loss = 0.0
for i in range(B):
    # original branch
    # simplified:  (xyz, t) => d_xyz
    d_xyz = deform(GS.xyz, Ts[i])
    image = splatting(GS, Vs[i], d_xyz=d_xyz)
    loss += L1(image, images_gt[i])

    # TimeFormer branch:  use offset_t[i,:,:]
    d_xyz_t = deform(GS.xyz+offset_t[i,:,:], Ts[i])
    image_t = splatting(GS, Vs[i], d_xyz=d_xyz_t)
    loss += lambda_t * L1(image_t, images_gt[i])

# Optimize shared deformation field, TimeFormer
 and Gaussians in the canonical space
loss.backward()
deform.optimizer.step()
timeformer.optimizer.step()
GS.optimizer.step()
```


extrapolation and free view synthesis. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12, 2024. 2, 3

[72] Kaichen Zhou, Jia-Xing Zhong, Sangyun Shin, Kai Lu, Yiyuan Yang, Andrew Markham, and Niki Trigoni. Dynpoint: Dynamic neural point for view synthesis. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[73] 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. 3

[74] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM transactions on graphics (TOG)*, 23(3):600–608, 2004. 2


## A. Implementation

In this section, we provide the Pytorch [41] code of TimeFormer in Alg. 1 and two-stream optimization strategy in Alg. 2, to clarify the framework in Fig. 3.

Alg. 1 is an additional explanation for Fig. 4, including three parts: 1) Position Encoding (PE), 2) Definition of Transformer encoder, 3) Definition of tiny MLP. Note that we use a shared TimeFormer on all Gaussians in the canonical space.

The Transformer Encoder receives input structured as [seq_len, seq_batch, channel], and we input x structured as [$B$, $N$, 4], where $B$ is the size of time batch, $N$ is the number of Gaussians and 4 means 3 position channel and 1 channel for the timestamp. Alg. 2 shows how we construct input to time.

In Alg. 2, we introduce the process in a time batch in detail. We first construct input to TimeFormer by concatenating Gaussian positions and sampled time stamps, as in Sec. 4.2. Besides the original branch where the deformation function is directly performed on the canonical space, we add another TimeFormer branch. TimeFormer calculates prior offsets "offset_t" via cross-time relationships before the deformation field. These two branches are optimized at the same time during training, while the TimeFormer branch can be removed during inference.

## B. Analysis on Canonical Space & FPS

Apart from Fig. 2, we provide more results in Fig. 12 and Fig. 13, as a further illustration on TimeFormer's capability to reduce Gaussians in the canonical space and improve inference speed. TimeFormer promotes more efficient spatial distribution of Gaussians in the canonical space, leading to improvements in reconstruction quality while simultaneously eliminating a substantial number of redundant Gaussians compared to baseline methods.

## C. Slider Window Demo

We provide additional comparison results of baseline methods with TimeFormer. We strongly recommend opening the following website demos in folder *website_demos* and us-

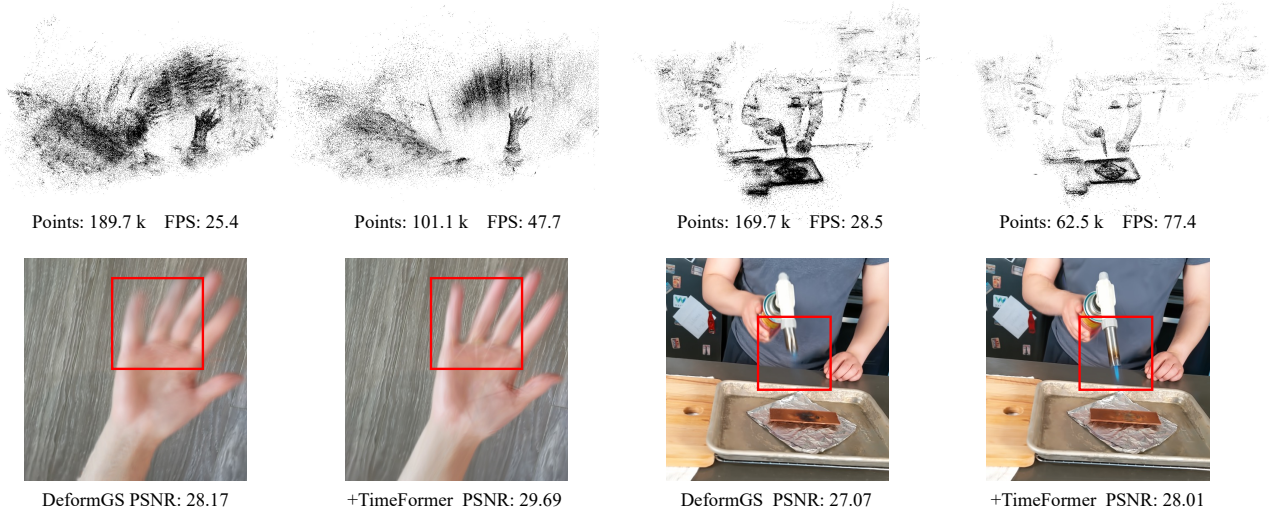



Figure 12. Comparisons of Canonical Space, FPS on Hypernerf Dataset [39].

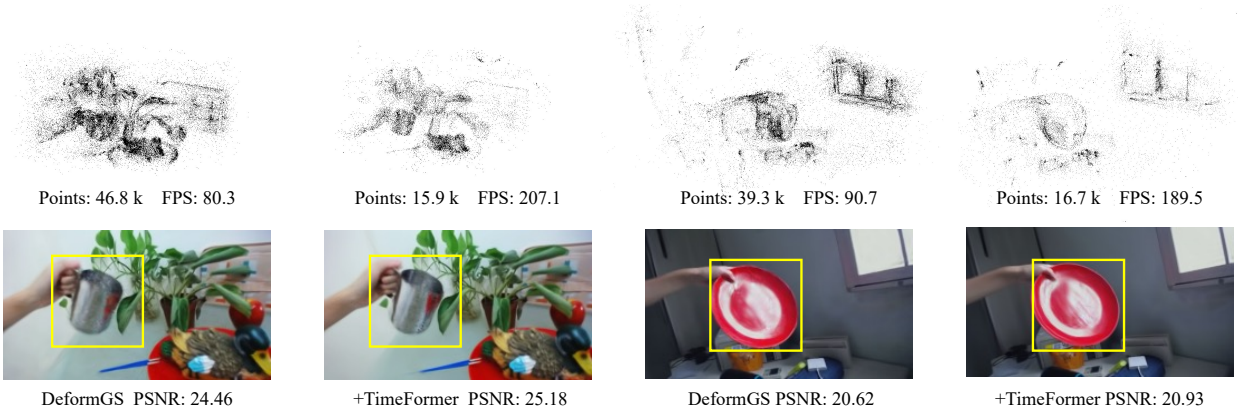



Figure 13. Comparisons of Canonical Space, FPS on NeRF-DS Dataset [65].

| Setting | Bell | | | Press | | | Sieve | | | Mean | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ |
| Baseline | 24.92 | 0.854 | 0.126 | 25.68 | 0.866 | 0.141 | 25.71 | 0.881 | 0.108 | 25.44 | 0.867 | 0.125 |
| M=1 | 25.67 | 0.872 | 0.097 | 26.17 | 0.867 | 0.141 | 26.04 | 0.884 | 0.113 | 25.96 | 0.874 | 0.117 |
| M=2 | 25.46 | 0.872 | 0.104 | 26.01 | 0.866 | 0.139 | 26.22 | 0.883 | 0.116 | 25.90 | 0.874 | 0.120 |
| M=3 | 25.87 | 0.875 | 0.095 | 26.29 | 0.865 | 0.138 | 26.00 | 0.887 | 0.104 | 26.05 | 0.876 | 0.112 |
| M=4 | 25.71 | 0.870 | 0.103 | 26.09 | 0.865 | 0.14 | 26.33 | 0.885 | 0.115 | 26.04 | 0.873 | 0.119 |
| w/o Shared | 24.12 | 0.832 | 0.137 | 25.01 | 0.854 | 0.146 | 25.29 | 0.869 | 0.113 | 24.81 | 0.852 | 0.132 |

Table 6. Ablation Results of three scenes *press*, *sieve* and *bell* on NeRF-DS Dataset [65].

ing the "slider window" to see the improvements brought by TimeFormer more clearly.

- Overview: index.html
- 4DGS+TimeFormer on HyperNeRF Dataset: index_4DGS_hypernerf.html
- DeformGS+TimeFormer on HyperNeRF Dataset: index_deformGS_hypernerf.html
- DeformGS+TimeFormer on NeRF-DS Dataset: index_deformGS_nerfds.html

## D. Ablation Studies

We provide more detailed ablation results on three scenes on NeRF-DS Dataset [65], as in Tab. 6. This further illustrates that TimeFormer is not sensitive to the number of transformer encoder layers $M$. However, we observe a significant decrease in reconstruction quality on all scenes without shared deformation fields.