GAUSSIANFLOW: SPLATTING GAUSSIAN DYNAMICS FOR 4D CONTENT CREATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Creating 4D fields of Gaussian Splatting from images or videos is a challenging task due to its under-constrained nature. While the optimization can draw photometric reference from the input videos or be regulated by generative models, directly supervising Gaussian motions remains underexplored. In this paper, we introduce a novel concept, Gaussian flow, which connects the dynamics of 3D Gaussians and pixel velocities between consecutive frames. The Gaussian flow can be efficiently obtained by splatting Gaussian dynamics into the image space. This differentiable process enables direct dynamic supervision from optical flow. Our method significantly benefits 4D dynamic content generation and 4D novel view synthesis with Gaussian Splatting, especially for contents with rich motions that are hard to be handled by existing methods. The common color drifting issue that happens in 4D generation is also resolved with improved Guassian dynamics. Superior visual quality on extensive experiments demonstrates our method's effectiveness. As shown in our evaluation, Gaussian Flow can drastically improve both quantitative and qualitative results for 4D Generation and 4D novel view synthesis.

1 INTRODUCTION

4D dynamic content creation from monocular or multi-view videos has garnered significant attention from academia and industry due to its wide applicability in virtual reality/augmented reality, digital games, and movie industry. Studies (Li et al., 2022; Pumarola et al., 2021; Park et al., 2021a;b) model 4D scenes by 4D dynamic Neural Radiance Fields (NeRFs) and optimize them based on input multi-view or monocular videos. Once optimized, the 4D field can be viewed from novel camera poses at preferred time steps through volumetric rendering. A more challenging task is generating 360 degree 4D content based on uncalibrated monocular videos or synthetic videos generated by text-to-video or image-to-video models. Since the monocular input cannot provide enough multi-view cues and unobserved regions are not supervised due to occlusions, studies (Singer et al., 2023; Jiang et al., 2023; Zhao et al., 2023) optimizes 4D dynamic NeRFs by leveraging generative models to create plausible and temporally consistent 3D structures and appearance. The optimization of 4D NeRFs requires volumetric rendering which makes the process time-consuming. And real-time rendering of optimized 4D NeRFs is also hardly achieved without special designs. A more efficient alternative is to model 4D Radiance Fields by 4D Gaussian Splatting (GS) (Wu et al., 2023; Luiten et al., 2023), which extends 3D Gaussian Splatting (Kerbl et al., 2023) with a temporal dimension. Leveraging the efficient rendering of 3D GS, the lengthy training time of a 4D Radiance Field can be drastically reduced (Yang et al., 2023c; Ren et al., 2023) and rendering can achieve real-time speed during inference.

The optimization of 4D Gaussian fields takes photometric loss as major supervision. As a result, the scene dynamics are usually under-constraint. Similarly to 4D NeRFs (Li et al., 2023; Park et al., 2021a; Pumarola et al., 2021), the radiance properties and the time-varying spatial properties (location, scales, and orientations) of Gaussians are both optimized to reduce the photometric Mean Squared Error (MSE) between the rendered frames and the input video frames. The ambiguities of appearance, geometry, and dynamics have been introduced in the process and become prominent with sparse-view or monocular video input. Per-frame Score Distillation Sampling (SDS) (Tang et al., 2023) reduces the appearance-geometry ambiguity to some extent by involving multi-view supervision in latent domain. However, both monocular photometric supervision and SDS supervision do not directly supervise scene dynamics.

To avoid temporal inconsistency brought by fast motions, Consistent4D (Jiang et al., 2023) leverages a video interpolation block, which imposes a photometric consistency between the interpolated frame and generated frame, at a cost of involving more frames as pseudo ground truth for fitting. Similarly, AYG (Ling et al., 2023) uses text-to-video diffusion model to balance motion magnitude and temporal consistency with a pre-set frame rate. 4D NeRF model (Li et al., 2023) has proven that optical flows on reference videos are strong motion cues and can significantly benefit scene dynamics. However, for 4D GS, connecting 4D Gaussian motions with optical flows has following two challenges. First, a Gaussian's motion is in 3D space, but it is its 2D splat that contributes to rendered pixels. Second, multiple 3D Gaussians might contribute to the same pixel in rendering, and each pixel's flow does not equal to any one Gaussian's motion.

To overcome these challenges, we introduce a novel concept, Gaussian flow, bridging the dynamics of 3D Gaussians and pixel velocities between consecutive frames. Specifically, we assume the optical flow of each pixel in image space is influenced by the Gaussians that cover it. The Gaussian flow of each pixel is considered to be the weighted sum of these Gaussian motions in 2D. To obtain the Gaussian flow value on each pixel without losing the speed advantage of Gaussian Splatting, we splat 3D Gaussian dynamics, including scaling, rotation, and translation in 3D space, onto the image plane along with its radiance properties. As the whole process is end-to-end differentiable, the 3D Gaussian dynamics can be directly supervised by matching Gaussian flow with optical flow on input video frames. We apply such flow supervision to both 4D content generation and 4D novel view synthesis to showcase the benefit of our proposed method, especially for contents with rich motions that are hard to be handled by existing methods. The flow-guided Guassian dynamics also resolve the color drifting artifacts that are commonly observed in 4D Generation. We summarize our contributions as follows:

- We introduce a novel concept, Gaussian flow, that first time bridges the 3D Gaussian dynamics to resulting pixel velocities, enabling flow supervision for Gaussian Splatting based-representations. Matching Gaussian flows with optical flows, 3D Gaussian dynamics can be directly supervised.
- The Gaussian flow can be obtained by splatting Gaussian dynamics into the image space. Following the tile-based design by original 3D Gaussian Splatting, we implement the dynamics splatting in CUDA with minimal overhead. The operation to generate dense Gaussian flow from 3D Gaussian dynamics is highly efficient and end-to-end differentiable.
- With Gaussian flow to optical flow matching, our model drastically improves over existing Gaussian Splatting based-methods, especially on scene sequences of fast motions. Color drifting is also resolved with our improved Gaussian dynamics.

2 Related Works

3D Generation. 3D generation has drawn tremendous attention with the progress of various 2D or 3D-aware diffusion models (Liu et al., 2023b; Rombach et al., 2022; Shi et al., 2023b; Liu et al., 2023c) and large vision models Radford et al. (2021); Jun & Nichol (2023); Nichol et al. (2022). Thanks to the availability of large-scale multi-view image datasets (Deitke et al., 2023; Yu et al., 2023; Downs et al., 2022), object-level multi-view cues can be encoded in generative models and are used for generation purpose. Pioneered by DreamFusion (Poole et al., 2022) that firstly proposes Score Distillation Sampling (SDS) loss to lift realistic contents from 2D to 3D via NeRFs, 3D content creation from text or image input has flourished. This progress includes approaches based on online optimization (Tang et al., 2023; Lin et al., 2023; Wang et al., 2024; Raj et al., 2023) and feedforward methods (Hong et al., 2023; Liu et al., 2023a; 2024; Xu et al., 2023; Wang et al., 2023c) with different representations such as NeRFs Mildenhall et al. (2021), triplane (Chan et al., 2022; Chen et al., 2022; Gao et al., 2023) and 3D Gaussian Splatting (Kerbl et al., 2023). 3D generation becomes more multi-view consistent by involving multi-view constraints (Shi et al., 2023b) and 3D-aware diffusion models (Liu et al., 2023b) as SDS supervision. Not limited to high quality rendering, studies (Sun et al., 2023; Long et al., 2023) also explore enhancing the quality of generated 3D geometry by incorporating normal cues.

4D Novel View Synthesis and Reconstruction. By adding timestamp as an additional variable, recent 4D methods with different dynamic representations such as dynamic NeRF (Park et al.,

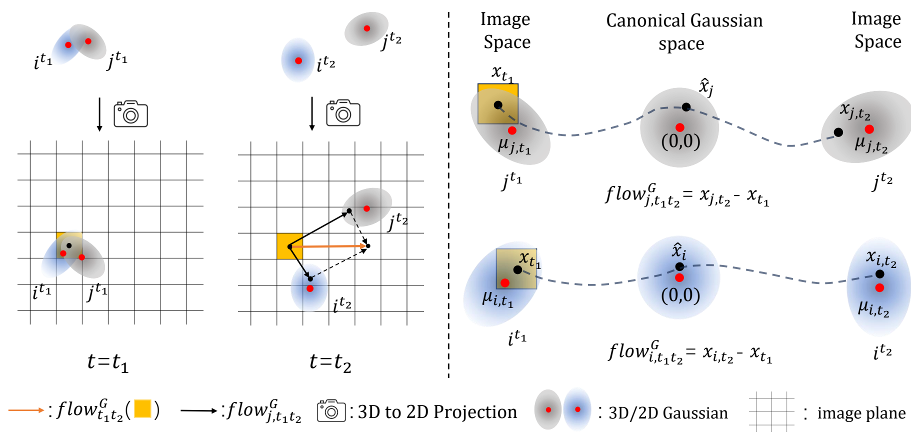


Figure 1: Between two consecutive frames, a pixel x_{t_1} will be pushed towards $x_{t_1} \rightarrow x_{i,t_2}$ by the 2D Gaussian i's motion $i^{t_1} \rightarrow i^{t_2}$. We can track x_{t_1} in Gaussian i by normalizing it to canonical Gaussian space as $\hat{x}_i$ and unnormalize it to image space to obtain x_{i,t_2}. Here, we denote this shift contribution from Gaussian i as $flow^G_{i,t_1,t_2}$. The Gaussian flow $flow^G_{t_1,t_2}(x_{t_1})$ on pixel x_{t_1} is defined as the weighted sum of the shift contributions from all Gaussians covering the pixel (i and j in our example). The weighting factor utilizes alpha composition weights. The Gaussian flow of the entire image can be obtained efficiently by splatting 3D Gaussian dynamics and rendering with alpha composition, which is implemented along with the CUDA pipeline of the original 3DGS Kerbl et al. (2023).

2021a;b; Li et al., 2021; Wang et al., 2023a; Li et al., 2022; Tretschk et al., 2021; Gao et al., 2021), dynamic triplane Fridovich-Keil et al. (2023); Cao & Johnson (2023); Shao et al. (2023) and 4D Gaussian Splatting Wu et al. (2023); Yang et al. (2023c); Lin et al. (2024) are proposed to achieve high quality 4D motions and scene contents reconstruction from either calibrated multi-view or uncalibrated RGB monocular video inputs. There are also some works (Newcombe et al., 2011; 2015; Zollhöfer et al., 2014) reconstruct rigid and non-rigid scene contents with RGB-D sensors, which help to resolve 3D ambiguities by involving depth cues. Different from static 3D reconstruction and novel view synthesis, 4D novel view synthesis consisting of both rigid and non-rigid deformations is notoriously challenging and ill-posed with only RGB monocular inputs. Some progress (Li et al., 2021; Gao et al., 2021; Tretschk et al., 2021; Wang et al., 2021) involve temporal priors and motion cues (e.g. optical flow) to better regularize temporal photometric consistency and 4D motions. One of recent works (Wang et al., 2023a) provides an analytical solution for flow supervision on deformable NeRF without inverting the backward deformation function from world coordinate to canonical coordinate. Several works (Yang et al., 2021a;b; 2023a;b) explore object-level mesh recovery from monocular videos with optical flow.

4D Generation. Similar to 3D generation from text prompts or single images, 4D generation from text prompts or monocular videos also relies on frame-by-frame multi-view cues from pre-trained diffusion models. Besides, 4D generation methods yet always rely on either video diffusion models or video interpolation block to ensure the temporal consistency. Animate124 (Zhao et al., 2023), 4D-fy (Bahmani et al., 2023) and one of the earliest works Singer et al. (2023) use dynamic NeRFs as 4D representations and achieve temporal consistency with text-to-video diffusion models, which can generate videos with controlled frame rates. Instead of using dynamic NeRF, Align Your Gaussians (Ling et al., 2023) DreamGaussian4D (Ren et al., 2023) and L4GM Ren et al. (2024) generate vivid 4D contents with 3D Gaussian Splatting, but again, relying on text-to-video diffusion model for free frame rate control. Without the use of text-to-video diffusion models, Consistent4D (Jiang et al., 2023) achieves coherent 4D generation with an off-the-shelf video interpolation model (Huang et al., 2022). Our method benefits 4D Gaussian representations by involving flow supervision and without the need of specialized temporal consistency networks.

3 METHODOLOGY

To better illustrate the relationship between Gaussian motions and corresponding pixel flow in image space, we first recap the rendering process of 3D Gaussian Splatting and then investigate its 4D case.

3.1 PRELIMINARY

3D Gaussian Splatting. From a set of initialized 3D Gaussian primitives, 3D Gaussian Splatting aims to recover the 3D scene by minimizing photometric loss between input m images $\{I\}_m$ and rendered images $\{I_r\}_m$. For each pixel, its rendered color C is the weighted sum of multiple Gaussians' colors c_i in depth order along the ray by point-based α-blending as in Eq. 1,

$$C = \sum_{i=1}^{N} T_i \alpha_i c_i, \tag{1}$$

with weights specifying as

$$\alpha_i = o_i e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)} \quad \text{and} \quad T_i = \sum_{j=1}^{i-1}(1-\alpha_i). \tag{2}$$

where $o_i \in [0,1]$, $\boldsymbol{\mu}_i \in \mathbb{R}^{2\times1}$, and $\boldsymbol{\Sigma}_i \in \mathbb{R}^{2\times2}$ are the opacity, 2D mean, and 2D covariance matrix of i-th 2D Gaussian projected from 3D space, respectively. And $\mathbf{x}$ is the intersection between a pixel ray and i-th Gaussian. As shown in Eq. 1, the relationship between a rendered pixel and 3D Gaussians is not bijective.

3D Gaussian Splatting in 4D. Modeling 4D motions with 3D Gaussian Splatting can be done frame-by-frame via either directly multi-view fitting (Luiten et al., 2023) or moving 3D Gaussians with a time-variant deformation field (Ling et al., 2023; Ren et al., 2023) or parameterize 3D Gaussians with time (Yang et al., 2023c). While with monocular inputs, Gaussian motions are under-constrained because different Gaussian motions can lead to the same rendered color, and thus long-term persistent tracks are lost (Luiten et al., 2023). Though Local Rigidity Loss (Luiten et al., 2023; Ling et al., 2023) is proposed to reduce global freedom of Gaussian motions, it sometimes brings severe problems due to poor or challenging initialization and lack of multi-view supervision. As shown in Fig. 6, 3D Gaussians initialized with the skull mouth closed are hard to be split when the mouth open with Local Rigidity Loss.

3.2 GAUSSIANFLOW

We consider the full freedom of each Gaussian motion in a 4D field, including 1) scaling, 2) rotation, and 3) translation at each time step. As the time changes, Gaussians covering the queried pixel at $t = t_1$ will move to other places at $t = t_2$, as shown in Fig. 1. To specify new pixel location $\mathbf{x}_{t_2}$ at $t = t_2$, we first project all the 3D Gaussians into 2D image plane as 2D Gaussians and calculate their motion's influence on pixel shifts.

Flow from Single Gaussian. To track pixel shifts (flow) contributed by Gaussian motions, we let the relative position of a pixel in a deforming 2D Gaussian stay the same. This setting preserves the mahalanobis distance between the pixel locations under two consecutive time steps and the 2D Gaussian unchanged. According to Eq. 2, this preservation will grant the pixel with the same radiance and α contribution from the 2D Gaussian, albeit the 2D Gaussian is deformed.

The pixel shift (flow) is the image space distance of the same pixel at two time steps. We first calculate the pixel shift influenced by a single 2D Gaussian that covers the pixel. We can find a pixel $\mathbf{x}$'s location at t_2 by normalizing its image location at t_1 to canonical Gaussian space and unnormalizing it to image space at t_2:

1) *normalize*. A pixel $\mathbf{x}_{t_1}$ following i-th 2D Gaussian distribution can be written as $\mathbf{x}_{t_1} \sim N(\boldsymbol{\mu}_{i,t_1}\boldsymbol{\Sigma}_{i,t_1})$. And in i-th Gaussian coordinate system with 2D mean $\boldsymbol{\mu}_{i,t_1} \in \mathbb{R}^{2\times1}$ and 2D covariance matrix $\boldsymbol{\Sigma}_{i,t_1} \in \mathbb{R}^{2\times2}$. After normalizing the i-th Gaussian into the standard normal distribution, we denote the pixel location in canonical Gaussian space as

$$\hat{\mathbf{x}}_{t_1} = \mathbf{B}_{i,t_1}^{-1}(\mathbf{x}_{t_1} - \boldsymbol{\mu}_{i,t_1}), \tag{3}$$

which follows $\boldsymbol{\Sigma}_{i,t_1} = \mathbf{B}_{i,t_1}\mathbf{B}_{i,t_1}^T$, $\hat{\mathbf{x}}_{t_1} \sim N(\mathbf{0}, \mathbf{I})$ and $\mathbf{I} \in \mathbb{R}^{2\times2}$ is identity matrix.

2) *unnormalize*. When $t = t_2$, the new location along with the Gaussian motion denotes $\mathbf{x}_{i,t_2}$ on the image plane.

$$\mathbf{x}_{i,t_2} = \mathbf{B}_{i,t_2}\hat{\mathbf{x}}_{t_1} + \boldsymbol{\mu}_{i,t_2}, \tag{4}$$

and $\mathbf{\Sigma}_{i,t_2} = \mathbf{B}_{i,t_2}\mathbf{B}_{i,t_2}^T$, $\mathbf{x}_{t_2} \sim N(\boldsymbol{\mu}_{i,t_2}, \mathbf{\Sigma}_{i,t_2})$. Eq. 3 and Eq. 4 preserve Mahalanobis distance between the tracked pixel and the 2D Gaussian leading to consistent α value (see Eq.2) for this pixel across consecutive time steps. The pixel shift contribution from each Gaussian therefore can be calculated as:

$$flow^G_{i,t_1t_2} = \mathbf{x}_{i,t_2} - \mathbf{x}_{t_1} \tag{5}$$

Flow Composition. In the original 3D Gaussian Splatting, a pixel's color is the weighted sum of the 2D Gaussians' radiance contribution. Similarly, we define the Gaussian flow value at a pixel as the weighted sum of the 2D Gaussians' contributions to its pixel shift, following alpha composition. With Eq. 3 and Eq. 4, the Gaussian flow value at pixel $\mathbf{x}_{t_1}$ from $t = t_{t_1}$ to $t = t_{t_2}$ is

$$\begin{aligned} flow^G_{t_1t_2} &= \sum_{i=1}^{K} w_i flow^G_{i,t_1t_2} && (6)\\ &= \sum_{i=1}^{K} w_i \left[\mathbf{B}_{i,t_2}\mathbf{B}^{-1}_{i,t_1}(\mathbf{x}_{t_1} - \boldsymbol{\mu}_{i,t_1}) + \boldsymbol{\mu}_{i,t_2} - \mathbf{x}_{t_1})\right], && (7) \end{aligned}$$

where K is the number of Gaussians along each camera ray sorted in depth order and each Gaussian has weight $w_i = \frac{T_i\alpha_i}{\Sigma_i T_i\alpha_i}$ according to Eq. 1, but normalized to [0,1] along each pixel ray. The intuition behind the using of the same weight as α-blending is that, if a pixel color is contributed by a weighted sum of a set of Gaussians, then its corresponding pixel shift i.e. pixel-wised optical flow should also be contributed by the same set of Gaussians with the same weights by nature, since optical flow is calculated based on the pixel-wised correspondences as well.

In some cases Ling et al. (2023); Keetha et al. (2023); Yugay et al. (2023); Matsuki et al. (2023), each Gaussian is assumed to be isotropic, and its scaling matrix $\mathbf{S} = \sigma\mathbf{I}$, where σ is the scaling factor. And its 3D covariance matrix $\mathbf{RSS}^T\mathbf{R}^T = \sigma^2\mathbf{I}$. If the scaling factor of each Gaussian doesn't change too much across time, $\mathbf{B}_{i,t_2}\mathbf{B}^{-1}_{i,t_1} \approx \mathbf{I}$. Therefore, to pair with this line of work, the formulation of our Gaussian flow as in Eq. 7 can be simplified as

$$flow^G_{t_1t_2} = \sum_{i=1}^{K} w_i(\boldsymbol{\mu}_{i,t_2} - \boldsymbol{\mu}_{i,t_1}). \tag{8}$$

In other words, for isotropic Gaussian fields, Gaussian flow between two different time steps can be approximated as the weighted sum of individual translation of 2D Gaussian.

Following either Eq. 7 or Eq. 8, the Gaussian flow can be densely calculated at each pixel. The flow supervision at pixel $\mathbf{x}_{t_1}$ from $t = t_1$ to $t = t_2$ can then be specified as

$$\mathcal{L}_{flow} = ||flow^o_{t_1t_2}(\mathbf{x}_{t_1}) - flow^G_{t_1t_2}||, \tag{9}$$

where optical flow $flow^o_{t_1t_2}$ can be calculated by off-the-shelf methods as pseudo ground-truth. Our method also allows for camera motions, please refer to the our experiments on NeRF-DS dataset (Yan et al., 2023) and the supplementary material D for more details.

3.3 4D Content Generation

As shown in Fig. 2, 4D content generation with Gaussian representation takes an uncalibrated monocular video either by real capturing or generating from text-to-video or image-to-video models as input and output a 4D Gaussian field. 3D Gaussians are initialized from the first video frame with photometric supervision between rendered image and input image and a 3D-aware diffusion model (Liu et al., 2023b) for multi-view SDS supervision. In our method, 3D Gaussian initialization can be done by One-2-3-45 (Liu et al., 2024) or DreamGaussian (Tang et al., 2023). After initialization, 4D Gaussian field is optimized with per-frame photometric supervision, per-frame SDS supervision, and our flow supervision as in Eq. 9. The loss function for 4D Gaussian field optimization can be written as:

$$\mathcal{L} = \mathcal{L}_{photometric} + \lambda_1\mathcal{L}_{flow} + \lambda_2\mathcal{L}_{sds} + \lambda_3\mathcal{L}_{other}, \tag{10}$$

where λ_1, λ_2 and λ_3 are hyperparameters. $\mathcal{L}_{other}$ is optional and method-dependent. Though not used in our method, we leave it for completeness.

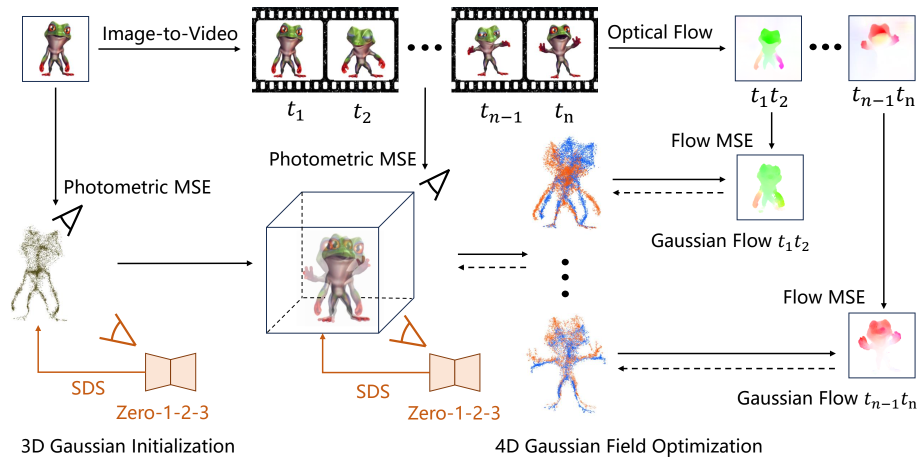


Figure 2: Overview of our 4D content generation pipeline. An uncalibrated monocular video or video generated from an image is taken as the input. We optimize a 3D Gaussian field initialized by the first frame with both photometric and SDS supervision (Liu et al., 2023b) (for 4D generation only). Then, we optimize the dynamics of the 3D Gaussians with the same two losses for each frame. Most importantly, we calculate Gaussian flows with our novel design on reference view for each consecutive two time steps and match it with a pre-computed optical flow of the input video. The gradients from the flow matching will propagate back through dynamics splatting and rendering process, resulting in a 4D Gaussian field with natural and smooth motions.

3.4 4D Novel View Synthesis

Unlike 4D content generation that has multi-view object-level prior from 3D-aware diffusion model, 4D novel view synthesis takes only multi-view or monocular input video frames for photometric supervision without any scene-level prior. 3D Gaussians are usually initialized by sfm (Snavely et al., 2006; Schonberger & Frahm, 2016) from input videos. After initialization, 4D Gaussian field is then optimized with per-frame photometric supervision and our flow supervision. We adopt the 4D Gaussian Fields from (Yang et al., 2023c). The loss function for 4D Gaussian field optimization can be written as:

$$\mathcal{L} = \mathcal{L}_{photometric} + \lambda_1 \mathcal{L}_{flow} + \lambda_2 \mathcal{L}_{other}, \tag{11}$$

where $\mathcal{L}_{other}$ is optional and method-dependent (please refer to Yang et al. (2023c)).

4 Experiments

In this section, we first provide implementation details of the proposed method and then valid our method on 4D Gaussian representations with (1) 4D novel view synthesis and (2) 4D generation. We test on the Plenoptic Video Datasets (Li et al., 2022) and the Consistent4D Dataset (Jiang et al., 2023) for both quantitative and qualitative evaluation. Our method achieves state-of-the-art results on both tasks. To obtain dense Gaussian flow, we efficient splatting the Gaussian dynamics along with the original 3DGS(Kerbl et al., 2023) CUDA pipeline. Please refer to our supplemental materials for implementation details.

4.1 Dataset

Plenoptic Video Dataset. A high-quality real-world dataset consists of 6 scenes with 30FPS and 2028 × 2704 resolution. There are 15 to 20 camera views per scene for training and 1 camera view for testing. The cameras are distributed to face the frontal part of scenes from different angles.

NeRF-DS Dataset. This dataset (Yan et al., 2023) consists of 8 scenes in everyday environments with various types of moving or deforming specular objects. Each scene contains two videos captured by two forward-facing cameras rigidly mounted together.

Consistent4D Dataset. This dataset (Jiang et al., 2023) includes 14 synthetic and 12 in-the-wild monocular videos. All the videos have only one moving object with a white background. 7 of the synthetic videos are provided with multi-view ground-truth for quantitative evaluation. Each input monocular video with a static camera is set at an azimuth angle of 0°. Ground-truth images include four distinct views at azimuth angles of -75°, 15°, 105°, and 195°, respectively, while keeping elevation, radius, and other camera parameters the same with input camera.

4.2 Results and Analysis

4D Novel View Synthesis. We visualize rendered images and depth maps of a very recent state-of-the-art 4D Gaussian method RT-4DGS (Yang et al., 2023c) with (yellow) and without (red) our flow supervision in Fig. 3. According to zoom-in comparisons, our method can consistently model realistic motions and correct structures. These regions are known to be challenging (Verbin et al., 2022; Liu et al., 2023d) for most methods, even under adequate multi-view supervision. Our method can reduce ambiguities in photometric supervision by involving motion cues and is shown to be consistently effective across frames. By using an off-the-shelf optical flow algorithm (Shi et al., 2023a), we found that only a small portion of image pixels from Plenoptic Video Dataset have optical flow values larger than one pixel. Since our method benefits 4D Gaussian-based methods more on the regions with large motions, we report PSNR numbers on both full scene reconstruction and dynamic regions (optical flow value > 1) in Tab. 1. With the proposed flow supervision, our method shows improved performance on all scenes and the gains are prominent on dynamic regions. Consequently, our 4D novel view synthesis results achieves state-of-the art quality. More comparisons are shown in the Fig. 11-13 and the video of the supplemental material.

Both qualitative and quantitative comparisons on NeRF-DS dataset in Fig. 4 and Tab. 2 show the effectiveness of the proposed method on scenes with complex camera motions, where we refer to our supplementary material D for more details in terms of implementations.

Table 1: Quantitative evaluation between ours and other methods on the DyNeRF dataset Li et al. (2022). We report PSNR numbers on both full-scene novel view synthesis and dynamic regions where the ground-truth optical flow value is larger than one pixel. "Ours" denotes RT-4DGS with the proposed flow supervision. We also achieve the best results on D-SSIM and LPIPS (see the Tab. 5 and 4 in the supplemental material).

Method	Coffee Martini	Spinach	Cut Beef	Flame Salmon	Flame Steak	Sear Steak	Mean
HexPlane Cao & Johnson (2023)	-	32.04	32.55	29.47	32.08	32.39	31.70
K-Planes Fridovich-Keil et al. (2023)	**29.99**	32.60	31.82	30.44	32.38	32.52	31.63
MixVoxels Wang et al. (2023b)	29.36	31.61	31.30	29.92	31.21	31.43	30.80
NeRFPlayer Song et al. (2023)	31.53	30.56	29.35	**31.65**	31.93	29.12	30.69
HyperReel Attal et al. (2023)	28.37	32.30	32.92	28.26	32.20	32.57	31.10
4DGS Wu et al. (2023)	27.34	32.46	32.90	29.20	32.51	32.49	31.15
RT-4DGS Yang et al. (2023c)	28.33	32.93	33.85	29.38	34.03	33.51	32.01
Ours	28.42	**33.68**	**34.19**	29.37	**34.22**	**34.06**	**32.32**
Dynamic Region Only							
RT-4DGS Yang et al. (2023c)	27.36	27.47	34.48	23.16	26.04	29.52	28.00
Ours	**28.02**	**28.71**	**35.18**	**23.36**	**27.53**	**31.14**	**28.99**

Table 2: Quantitative comparisons on NeRF-DS dataset. Note that our method is effective and robust under both complex camera motions and object motions.

	PSNR ↑	SSIM ↑	LPIPS↓
3DGS (Kerbl et al., 2023)	20.79	0.78	0.29
TiNeuVo (Fang et al., 2022)	21.60	0.83	0.30
HyperNeRF (Park et al., 2021b)	23.45	0.85	0.19
NeRF-DS (Yan et al., 2023)	23.40	0.84	0.18
Deformable-3DGS (Yang et al., 2024)	23.61	0.83	0.21
Deformable-3DGS (with flow)	**24.12**	**0.86**	**0.17**

4D Generation. We evaluate and compare DreamGaussian4D (Ren et al., 2023), which is a recent 4D Gaussian-based state-of-the-art generative model with open-sourced code, and dynamic NeRF-based methods in Tab. 3 on Consistent4D dataset with ours. Scores on individual videos are

Figure 3: Qualitative comparisons on DyNeRF dataset (Li et al., 2022). The left column shows the novel view rendered images and depth maps of RT-4DGS (Yang et al., 2023c), which suffers from artifacts in the dynamic regions. The right column shows the results of RT-4DGS optimized with our flow supervision during training. We refer to our supplementary material (Fig. 11-13, including the video) for more visual comparisons.

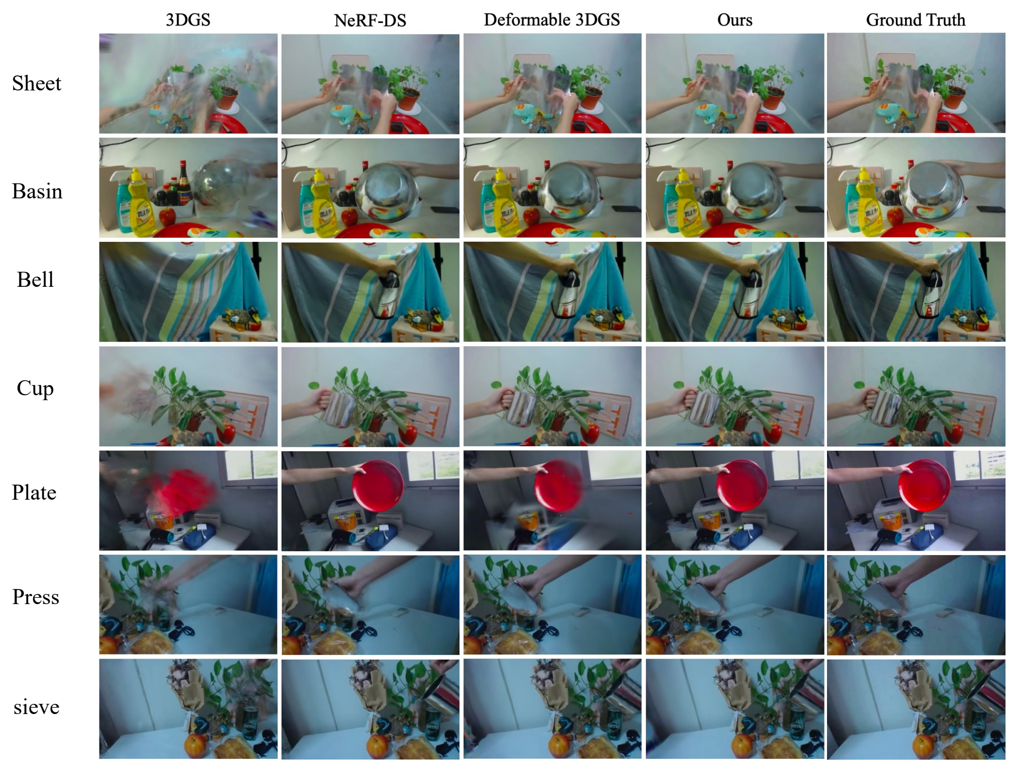


Figure 4: Qualitative comparisons on NeRF-DS dataset.

calculated and averaged over four novel views mentioned above. Note that flow supervision is effective and helps with 4D generative Gaussian representation. Compared to DreamGaussian4D, our method shows better quality as shown in Fig. 6 after the same number of training iterations. For the two hard dynamic scenes shown in Fig. 6, our method benefit from flow supervision and generate desirable motions, while DG4D shows prominent artifacts on the novel views. Additionally, flow supervision helps our method avoid color drifting, compared with dynamic NeRF-based method Consistent4D(Jiang et al., 2023) (Fig. 5). Our results are more consistent in terms of texture and geometry. We also show more generation results in the Fig. 8 of the supplemental material.

Table 3: Quantitative comparisons between ours and others on Consistent4D dataset.

Method	Pistol		Guppie		Crocodile		Monster		Skull		Trump		Aurorus		Mean	
	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑	LPIPS↓	CLIP↑
D-NeRF Pumarola et al. (2021)	0.52	0.66	0.32	0.76	0.54	0.61	0.52	0.79	0.53	0.72	0.55	0.60	0.56	0.66	0.51	0.68
K-planes Fridovich-Keil et al. (2023)	0.40	0.74	0.29	0.75	0.19	0.75	0.47	0.73	0.41	0.72	0.51	0.66	0.37	0.67	0.38	0.72
Consistent4D Jiang et al. (2023)	**0.10**	0.90	0.12	0.90	0.12	0.82	0.18	0.90	**0.17**	0.88	0.23	**0.85**	0.17	0.85	0.16	0.87
DG4D Ren et al. (2023)	0.12	0.92	0.12	0.91	0.12	0.88	0.19	0.90	0.18	0.90	0.22	0.83	0.17	0.86	0.16	0.87
Ours	**0.10**	**0.94**	**0.10**	**0.93**	**0.10**	**0.90**	**0.17**	**0.92**	**0.17**	**0.92**	**0.20**	**0.85**	**0.15**	**0.89**	**0.14**	**0.91**

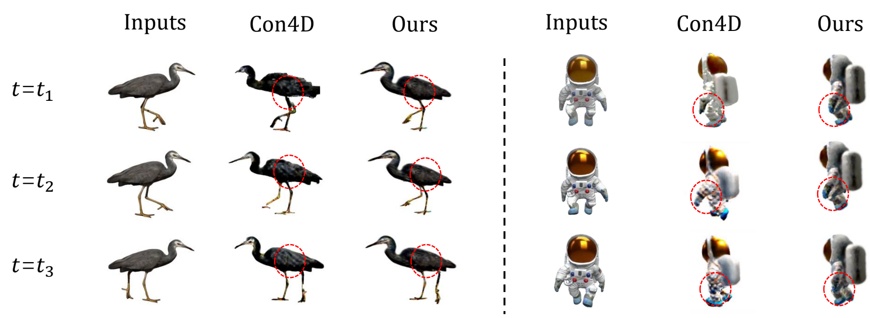


Figure 5: Comparisons between Consistent4D (Jiang et al., 2023) (a dynamic NeRF-based method) and ours. The flow supervision help us avoid the "bubble like" texture and non-consistent geometry on novel views.

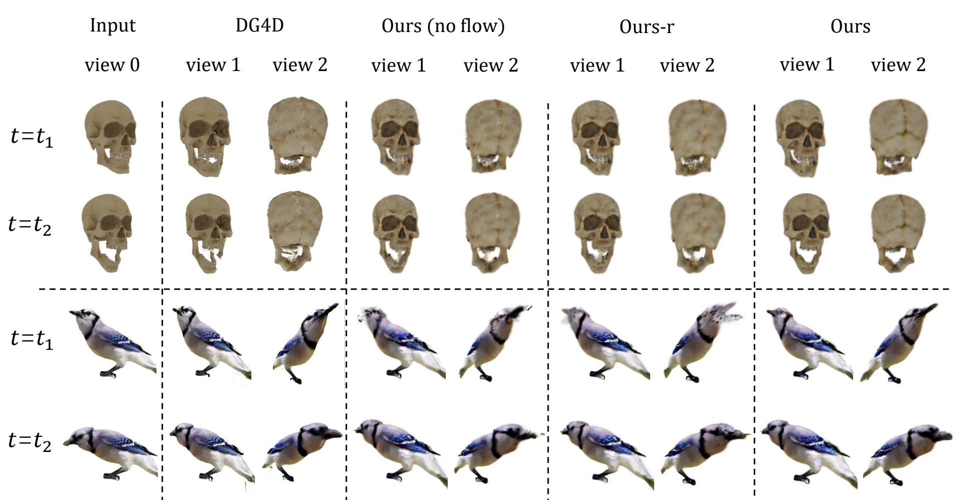


Figure 6: Qualitative comparisons among DreamGaussian4D (Ren et al., 2023), our method without flow loss, our method without flow loss but with Local Rigidity Loss (Ours-r) and ours.

5 Ablation Study

We validate our flow supervision through qualitative comparisons shown in Fig. 6. Compared with Ours (no flow) and Ours, the proposed flow supervision shows its effectiveness on moving parts. For the skull, 3D Gaussians on the teeth region initialized at $t = t_1$ are very close to each other and are hard to split apart completely when $t = t_2$. Because the Gaussians can move freely as long as they look photometrically correct from view 0, while SDS supervision applied from novel views works on latent domains and cannot provide pixel-wised supervision. This problem becomes more severe when involving Local Rigidity Loss (comparing Ours-r and Ours) because the motions of 3D Gaussians initialized at $t = t_1$ are constrained by their neighbors and the Gaussians are harder to split apart at $t = t_1$. Similarly, for bird, regions consisting of thin structures such as the bird's beak cannot be perfectly maintained across frames without our flow supervision. While originally utilized in 4D Gaussian fields (Luiten et al., 2023) to maintain the structure consistency during motion, Local Rigidity Loss as a motion constraint can incorrectly group Gaussians and is less effective than our flow supervision.

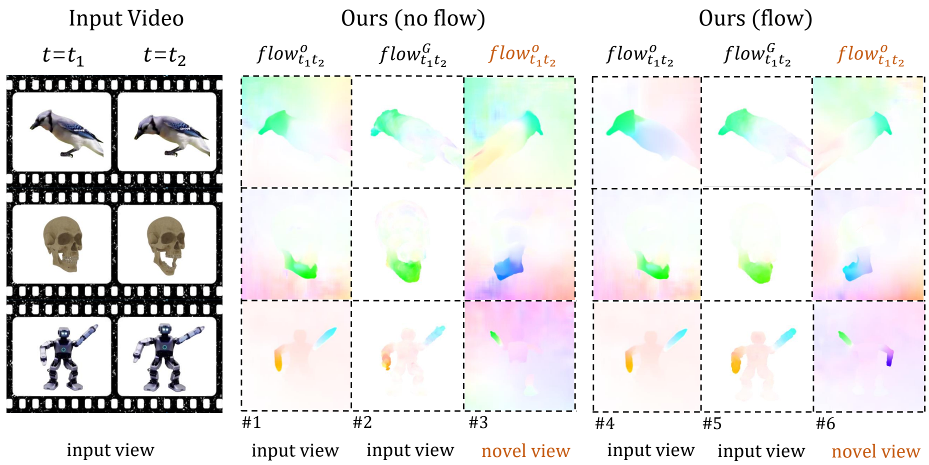


Figure 7: Visualization of optical and Gaussian flows on the input view and a novel view. "Ours (no flow)" denotes our model without flow supervision while "Ours" is our full model. The optical flow values of the background should be ignored because dense optical flow algorithms calculate correspondences among background pixels. We calculate optical flow $flow^o_{t_1t_2}$ on rendered sequences by autoflow (Sun et al., 2021). From column #1 and #4, we can see that both rendered sequences from input view have high-quality optical flow, indicating correct motions and appearance. Comparing Gaussian flow $flow^G_{t_1,t_2}$ at column #2 and #5 , we can see that the underlining Gaussians move inconsistently without flow supervision. It is due to the ambiguity of appearance and motions while only being supervised by photometric loss on a single input view. Aligning Gaussian flow to optical flow can drastically improve irregular motions(column #3) and create high-quality dynamic motions (column #6) on novel views.

We also visualize optical flow $flow^o_{t_1t_2}$ and Gaussian flow $flow^G_{t_1t_2}$ with and without our flow supervision in Fig. 7. In both cases, the optical flow $flow^o_{t_1t_2}$ between rendered images on the input view are very similar to each other (shown in #1 and # 4 column) and align with ground-truth motion because of direct photometric supervision on input view. However, comparing optical flows on novel view as shown in #3 and #6, without photometric supervision on novel views, inconsistent Gaussian motions are witnessed without our flow supervision. Gaussian flow $flow^G_{t_1t_2}$ in #2 column also reveals the inconsistent Gaussian motions. Incorrect Gaussian motion can still hallucinate correct image frames on input view. However, this motion-appearance ambiguity can lead to unrealistic motions from novel views (the non-smooth flow color on moving parts in #3). While #5 shows consistent Gaussian flow, indicating the consistent Gaussian motions with flow supervision.

6 Limitation

By aligning with the optical flow, our Gaussian flow effectively optimizes Gaussian splats' motion. However, if the optical flow cannot be reliably estimated, our method cannot provide beneficial signal for optimization. For similar reason, this supervision is less helpful for modeling dynamic objects with constantly changing textures, which remains a challenge for current 4D generation methods.

7 Conclusion and Future Work

We present GaussianFlow, an analytical solution to supervise 3D Gaussian dynamics including scaling, rotation, and translation with 2D optical flow. Extensive qualitative and quantitative comparisons demonstrate that our method is general and beneficial to Gaussian-based representations for both 4D generation and 4D novel view synthesis with motions. In this paper, we only consider the short-term flow supervision between every two neighbor frames in our all experiments. Long-term flow supervision across multiple frames is expected to be better and smoother, which we leave as future work. Another promising future direction is to explore view-conditioned flow SDS to supervise Gaussian flow on novel view in the 4D generation task.

REFERENCES

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*, pp. 16610–16620, 2023.

Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. *arXiv preprint arXiv:2311.17984*, 2023.

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*, pp. 130–141, 2023.

Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16123–16133, 2022.

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

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13142–13153, 2023.

Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2553–2560. IEEE, 2022.

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*, pp. 1–9, 2022.

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*, pp. 12479–12488, 2023.

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*, pp. 5712–5721, 2021.

Quankai Gao, Qiangeng Xu, Hao Su, Ulrich Neumann, and Zexiang Xu. Strivec: Sparse tri-vector radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17569–17579, 2023.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023.

Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *European Conference on Computer Vision*, pp. 624–642. Springer, 2022.

Yanqin Jiang, Li Zhang, Jin Gao, Weimin Hu, and Yao Yao. Consistent4d: Consistent 360 {\deg} dynamic object generation from monocular video. *arXiv preprint arXiv:2311.02848*, 2023.

Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. *arXiv preprint arXiv:2312.02126*, 2023.

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.

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*, pp. 5521–5531, 2022.

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*, pp. 6498–6508, 2021.

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*, pp. 4273–4284, 2023.

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 300–309, 2023.

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*, pp. 21136–21145, 2024.

Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. *arXiv preprint arXiv:2312.13763*, 2023.

Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. *arXiv preprint arXiv:2311.07885*, 2023a.

Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 36, 2024.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9298–9309, 2023b.

Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023c.

Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. Nero: Neural geometry and brdf reconstruction of reflective objects from multi-view images. *arXiv preprint arXiv:2305.17398*, 2023d.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. *arXiv preprint arXiv:2310.15008*, 2023.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023.

Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. *arXiv preprint arXiv:2312.06741*, 2023.

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.

Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pp. 127–136. Ieee, 2011.

Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 343–352, 2015.

Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022.

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*, pp. 5865–5874, 2021a.

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*, 2021b.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

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*, pp. 10318–10327, 2021.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Amit Raj, Srinivas Kaza, Ben Poole, Michael Niemeyer, Nataniel Ruiz, Ben Mildenhall, Shiran Zada, Kfir Aberman, Michael Rubinstein, Jonathan Barron, et al. Dreambooth3d: Subject-driven text-to-3d generation. *arXiv preprint arXiv:2303.13508*, 2023.

Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023.

Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, et al. L4gm: Large 4d gaussian reconstruction model. *arXiv preprint arXiv:2406.10324*, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

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

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*, pp. 16632–16642, 2023.

Xiaoyu Shi, Zhaoyang Huang, Weikang Bian, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Videoflow: Exploiting temporal cues for multi-frame optical flow estimation. *arXiv preprint arXiv:2303.08340*, 2023a.

Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv preprint arXiv:2308.16512*, 2023b.

Uriel Singer, Shelly Sheynin, Adam Polyak, Oron Ashual, Iurii Makarov, Filippos Kokkinos, Naman Goyal, Andrea Vedaldi, Devi Parikh, Justin Johnson, et al. Text-to-4d dynamic scene generation. *arXiv preprint arXiv:2301.11280*, 2023.

Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pp. 835–846. 2006.

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.

Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10093–10102, 2021.

Jingxiang Sun, Bo Zhang, Ruizhi Shao, Lizhen Wang, Wen Liu, Zhenda Xie, and Yebin Liu. Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior. *arXiv preprint arXiv:2310.16818*, 2023.

Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023.

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*, pp. 12959–12970, 2021.

Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. in 2022 ieee. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5481–5490, 2022.

Chaoyang Wang, Ben Eckart, Simon Lucey, and Orazio Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021.

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*, pp. 21128–21137, 2023a.

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*, pp. 19706–19716, 2023b.

Peng Wang, Hao Tan, Sai Bi, Yinghao Xu, Fujun Luan, Kalyan Sunkavalli, Wenping Wang, Zexiang Xu, and Kai Zhang. Pf-lrm: Pose-free large reconstruction model for joint pose and shape prediction. *arXiv preprint arXiv:2311.12024*, 2023c.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36, 2024.

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.

Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *arXiv preprint arXiv:2311.09217*, 2023.

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*, pp. 8285–8295, 2023.

Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T Freeman, and Ce Liu. Lasr: Learning articulated shape reconstruction from a monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15980–15989, 2021a.

Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Ce Liu, and Deva Ramanan. Viser: Video-specific surface embeddings for articulated 3d shape reconstruction. *Advances in Neural Information Processing Systems*, 34:19326–19338, 2021b.

Gengshan Yang, Chaoyang Wang, N Dinesh Reddy, and Deva Ramanan. Reconstructing animatable categories from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16995–17005, 2023a.

Gengshan Yang, Shuo Yang, John Z Zhang, Zachary Manchester, and Deva Ramanan. Ppr: Physically plausible reconstruction from monocular videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3914–3924, 2023b.

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

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20331–20341, 2024.

Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9150–9161, 2023.

Vladimir Yugay, Yue Li, Theo Gevers, and Martin R Oswald. Gaussian-slam: Photo-realistic dense slam with gaussian splatting. *arXiv preprint arXiv:2312.10070*, 2023.

Yuyang Zhao, Zhiwen Yan, Enze Xie, Lanqing Hong, Zhenguo Li, and Gim Hee Lee. Animate124: Animating one image to 4d dynamic scene. *arXiv preprint arXiv:2311.14603*, 2023.

Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (ToG)*, 33 (4):1–12, 2014.

A Implementation Details

We take t_2 as the next timestep of t_1 and calculate optical flow between every two neighbor frames in all experiments. In our CUDA implementation of Gaussian dynamics splatting, though the number of Gaussians K along each pixel ray is usually different, we use $K = 20$ to balance speed and effectiveness. A larger K means more number of Gaussians and their gradient will be counted through backpropagation. For video frames with size $H \times W \times 3$, we track the motions of Gaussians between every two neighbor timesteps t_1 and t_2 by maintaining two $H \times W \times K$ tensors to record the indices of top-K Gaussians sorted in depth order, top-K Gaussians' rendered weights w_i for each pixel and an another tensor with size $H \times W \times K \times 2$ denotes the distances between pixel coordinate and 2D Gaussian means $\mathbf{x}_{t_1} - \boldsymbol{\mu}_{i,t_1}$, respectively. Besides, 2D mean $\boldsymbol{\mu}_{i,t_1}$ and 2D covariance matrices $\boldsymbol{\Sigma}_{i,t_1}$ and $\boldsymbol{\Sigma}_{i,t_2}$ of each Gaussian at different two timesteps are accessible via camera projection (Kerbl et al., 2023).

Algorithm 1: Detailed pseudo code for GaussianFlow

Input:
$flow^o_{t_k,t_{k+1}}$: Pseudo ground-truth optical flow from off-the-shelf optical flow algorithm;
$I^{gt}_{t_k}$: ground-truth images , where $k = 0, 1, ..., T$;
$renderer$: A Gaussian renderer;
$Gaussians_{t_k}, Gaussians_{t_{k+1}}$: n Gaussians with learnable parameters at t_k and t_{k+1};
cam_{t_k} and $cam_{t_{k+1}}$: Camera parameters at t_k and t_{k+1};
Loss init
$\mathcal{L} = 0$
for timestep $k \leq T - 1$ **do**
  // renderer outputs at t_k
  $renderer_{t_k} = renderer(Gaussians_{t_k}, cam_{t_k})$;
  $I^{render}_{t_k} = renderer_{t_k}[\text{"}image\text{"}]$; # $H \times W \times 3$
  $idx_{t_k} = renderer_{t_k}[\text{"}index\text{"}]$; # $H \times W \times K$, Gaussian indices that cover each pixels
  $w_{t_k} = renderer_{t_k}[\text{"}weights\text{"}]$; # $H \times W \times K$
  $w_{t_k} = w_{t_k}/sum(w_{t_k}, dim = -1)$; # $H \times W \times K$, weight normalization
  $x\_\mu_{t_k} = renderer_{t_k}[\text{"}x\_mu\text{"}]$; # $H \times W \times K \times 2$, denotes $x_{t_k} - \mu_{t_k}$
  $\mu_{t_k} = renderer_{t_k}[\text{"}2D\_mean\text{"}]$; # $n \times 2$
  $\Sigma_{t_k} = renderer_{t_k}[\text{"}2D\_cov\text{"}]$; # $n \times 2 \times 2$
  $B_{t_k} = \Sigma^{\frac{1}{2}}_{t_k}$;
  # renderer outputs at t_{k+1}
  $renderer_{t_{k+1}} = renderer(Gaussians_{t_{k+1}}, cam_{t_{k+1}})$;
  $\mu_{t_{k+1}} = renderer_{t_{k+1}}[\text{"}2D\_mean\text{"}]$; # $n \times 2$
  $\Sigma_{t_{k+1}} = renderer_{t_{k+1}}[\text{"}2D\_cov\text{"}]$; # $n \times 2 \times 2$
  $B_{t_{k+1}} = \Sigma^{\frac{1}{2}}_{t_k}$;
  # Eq.8 while ignoring resize operations for simplicity
  $flow^G_{t_k,t_{k+1}} =$
   $w_{t_k} * \big(B_{t_{k+1}}[idx_{t_k}] * inv(B_{t_k})[idx_{t_k}] * x\_\mu_{t_k} + (\mu_{t_{k+1}}[idx_{t_k}] - \mu_{t_k}[idx_{t_k}] - x\_\mu_{t_k})\big)$
  # Eq.10
  $\mathcal{L}_{flow} = norm(flow^o_{t_k,t_{k+1}}, sum(flow^G_{t_k,t_{k+1}}, dim = 0))$
  # (1) Loss for 4D novel view synthesis
  $\mathcal{L} = \mathcal{L} + \mathcal{L}_{photometric}(I^{render}_{t_k}, I^{gt}_{t_k}) + \lambda_1 \mathcal{L}_{flow} + \lambda_3 \mathcal{L}_{other}$
  # (2) Loss for 4D generation
  $\mathcal{L} = \mathcal{L} + \mathcal{L}_{photometric}(I^{render}_{t_k}, I^{gt}_{t_k}) + \lambda_1 \mathcal{L}_{flow} + \lambda_2 \mathcal{L}_{sds} + \lambda_3 \mathcal{L}_{other}$
end

A detailed pseudo code for our flow supervision can be found at Algorithm 1. We extract the projected Gaussian dynamics and obtain the final Gaussian flow by rendering these dynamics. Variables including the weights and top-K indices of Gaussians per pixel (as mentioned in implementation details of our main paper) are calculated in CUDA by modifying the original CUDA kernel codes of 3D Gaussian Splatting (Kerbl et al., 2023). And Gaussian flow $flow^G$ is calculated by Eq.8 with PyTorch.

In our 4D generation experiment, we run 500 iterations static optimization to initialize 3D Gaussian fields with a batch size of 16. The Tmax in SDS is linearly decayed from 0.98 to 0.02. For dynamic representation, we run 600 iterations with batch size of 4 for both DG4D (Ren et al., 2023) and ours. The flow loss weight λ_1 in Eq. 11 of our main paper is 1.0.

Our method slightly decreases speed and increases memory only on training stage but not for inference stage because our flow supervision is only for training a better/robust deformation field or other 4DGS designs and then will be no needed in inference stage. The training speed for DG4D is around 1.4it/s while it then becomes around 2.2it/s with our flow supervision. And the difference between training speeds with (around 2.5s/it) and without (around 2.2s/it) our flow supervision for RT-4DGS is marginal. Even with more memory footprint by tracking per-pixel gradients for Gaussians, a single 30GB GPU is adequate for reproducing all our results. In our 4D novel view synthesis experiment, we follow RT-4DGS(Yang et al., 2023c) except that we add our proposed flow supervision for all cameras. The flow loss weight λ_1 in Eq. 11 of our main paper is 0.5.

B More Results

B.1 More Visualization and Comparison in 4D Generation.

More comparisons between Gaussian flow $flow^G$ and optical flow $flow^o$ on rendered images are shown in Fig. 9. The first row of each example is the rgb frames rendered from a optimized 4D Gaussian field. We rotate our cameras for each time steps so that the object can move as optimized and the camera is moving at the same time to show the scene from different angles. The second row of each example shows the visualized Gaussian flows. These Gaussian flows are calculated by the rendered images of consecutive time steps at each camera view, therefore, containing no camera motion in the flow values. The third row is the estimated optical flows between the rendered images of consecutive time steps at each camera view. We use off-the-shelf AutoFlow (Sun et al., 2021) for the estimation. We can see that enhanced by the flow supervision from the single input view, our 4D generation pipeline can model fast motion such as the explosive motion of the gun hammer (see the last example in Fig. 9).

Figure 8: Qualitative results on Consistent4D dataset.

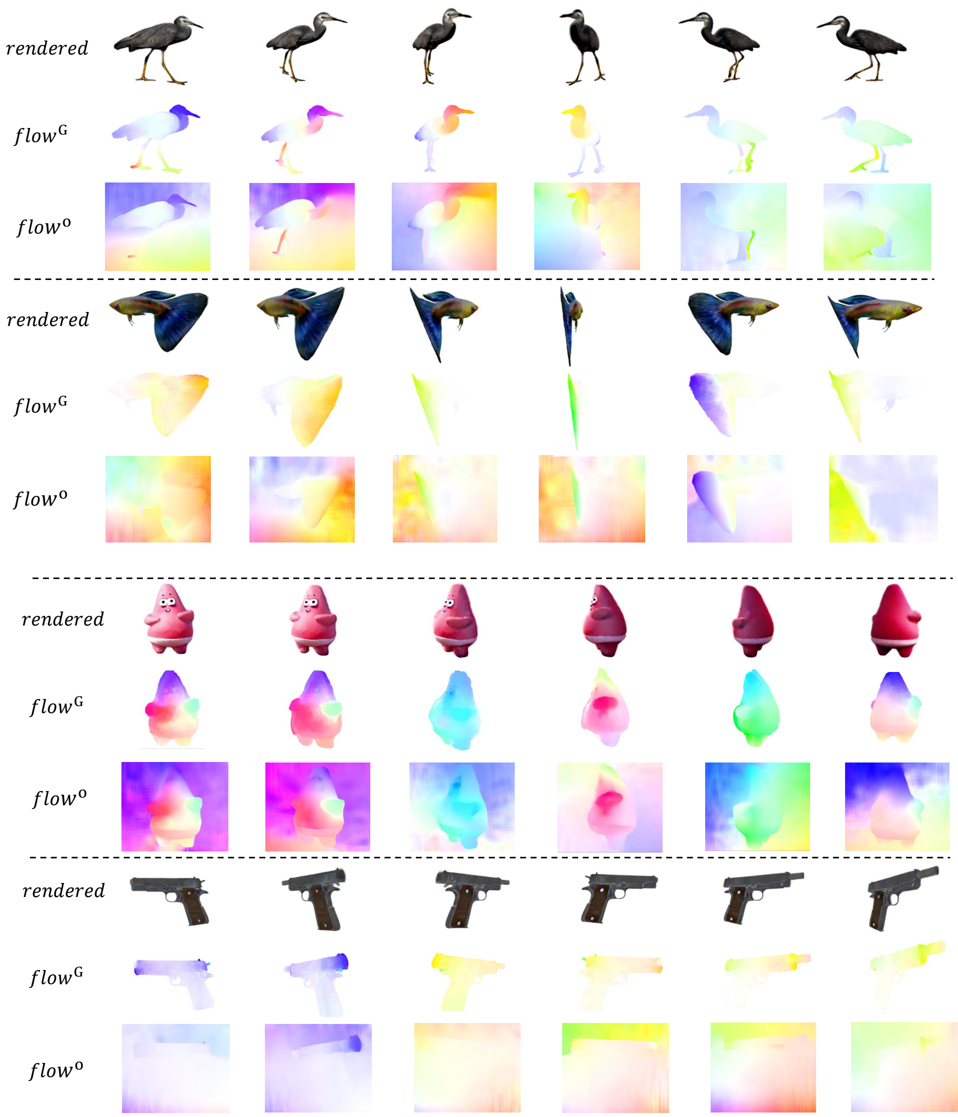


Figure 9: Visualization of Gaussian flow $flow^G$ and optical flow $flow^o$ on rendered sequences from different views.

B.2 More Quantitative Results on the DyNeRF Dataset.

We show SSIM, MSSIM, D-SSIM and LPIPS of our methods on the DyNeRF dataset (Li et al., 2022) breakdown by scenes in Tab. 4. We also show the comparisions of our methods and other methods on PSNR, D-SSIM, LPIPS averaged over all scenes of the DyNeRF dataset (Li et al., 2022) in Tab. 5.

Table 4: The SSIM, MSSIM, D-SSIM and LPIPS of our methods on the DyNeRF dataset breakdown by scenes.

	Coffee Martini	Spinach	Cut Beef	Flame Salmon	Flame Steak	Sear Steak	Mean
SSIM ↑	0.9185	0.9578	0.9598	0.9248	0.9643	0.9645	0.9483
MSSIM ↑	0.9544	0.9786	0.9808	0.9597	0.9816	0.9808	0.9726
D-SSIM ↓	0.0228	0.0107	0.0096	0.0202	0.0092	0.0096	0.0137
LPIPS ↓	0.0708	0.0389	0.0378	0.0639	0.0337	0.0354	0.0468

Table 5: Overall quantitative comparisions between ours and other methods on the DyNeRF dataset (Li et al., 2022). We report PSNR, D-SSIM, LPIPS averaged over all scenes. "Ours" denotes RT-4DGS with the proposed flow supervision.

	Mean PSNR ↑	Mean D-SSIM ↓	Mean LPIPS ↓
HexPlane Cao & Johnson (2023)	31.70	**0.014**	0.075
K-Planes Fridovich-Keil et al. (2023)	31.63	0.018	-
MixVoxels Wang et al. (2023b)	30.80	0.02	0.126
NeRFPlayer Song et al. (2023)	30.69	0.034	0.111
HyperReel Attal et al. (2023)	31.10	0.036	0.096
4DGS Wu et al. (2023)	31.15	0.016	0.150
RT-4DGS Yang et al. (2023c)	32.01	**0.014**	0.055
Ours	**32.32**	**0.014**	**0.047**

B.3 More Qualitative Results on the DyNeRF Dataset.

More qualitative results on DyNeRF dataset Li et al. (2022) can be found in Fig. 10, Fig. 11, Fig. 12 and our video.

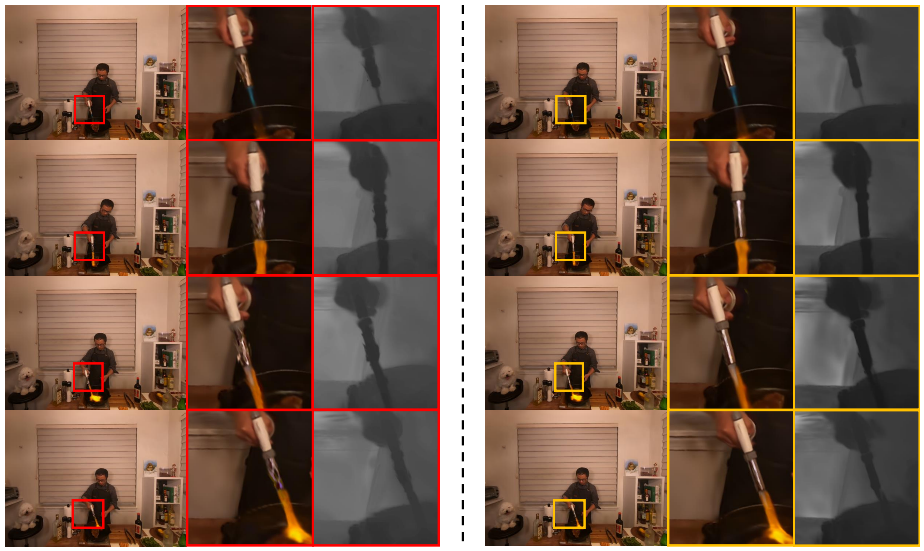
C Flow Visualization in Dynamic Gaussian Fields

Note that dynamic 3D Gaussian (Luiten et al., 2023) provided a way to visualize 3D scene motions between consecutive frames, however, by tracking one "most influential" 3D Gaussian per pixel. This is neither efficient nor effective to be used in flow supervision, because the "most influential" 3D Gaussian for each pixel is determined by searching the nearest 3D Gaussian's center from tens of thousands of 3D Gaussian candidates with a virtual 3D point along pixel ray lifted with corresponding rendered depth. Also, the "most influential" Gaussian of a pixel might not even cover the same pixel but still be considered just because this Gaussian's center is the nearest one to the virtual point among all 3D Gaussians. We have also applied flow supervision in this way, but we find it has no observable benefit for rendering quality while resulting in slower training speed due to the per-pixel nearest search. On the other hand, RT-4DGS showed "render flow" in their paper only for visualization purpose and the detail was not clarified and the function was not enabled, please refer to their code, issue 1 and issue 2.

D Camera Motion and Object Motion

When considering the cases with both camera motions and object motions, we have the relationship $flow^o = flow^G + flow^{cam}$, where $flow^{cam}$ is one portion of optical flow cased by camera motion and $flow^G$ is still the foreground object Gaussian dynamics. And the original flow supervision in our Eq.9 is rewritten as:

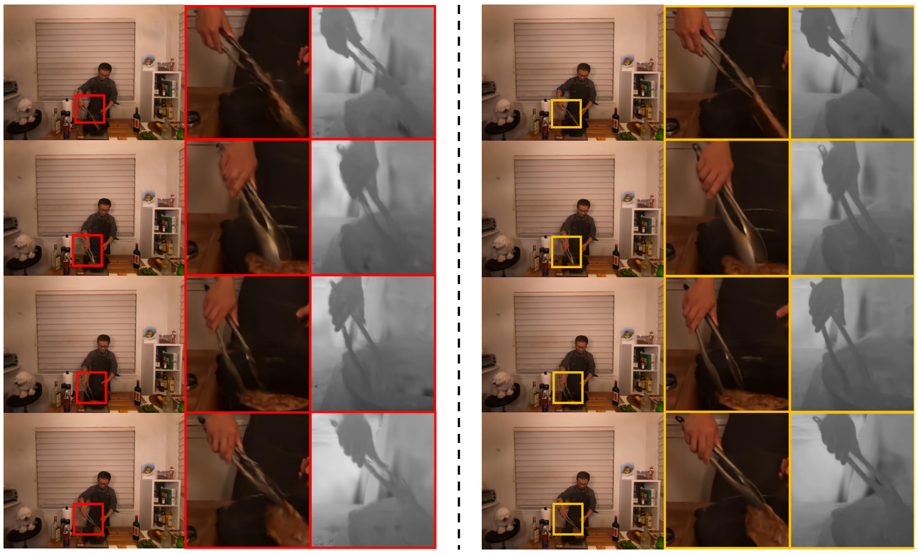
$$\mathcal{L}_{flow} = ||flow^o_{t_1 t_2}(\mathbf{x}_{t_1}) - flow^{cam}_{t_1 t_2} - flow^G_{t_1 t_2}||, \tag{12}$$



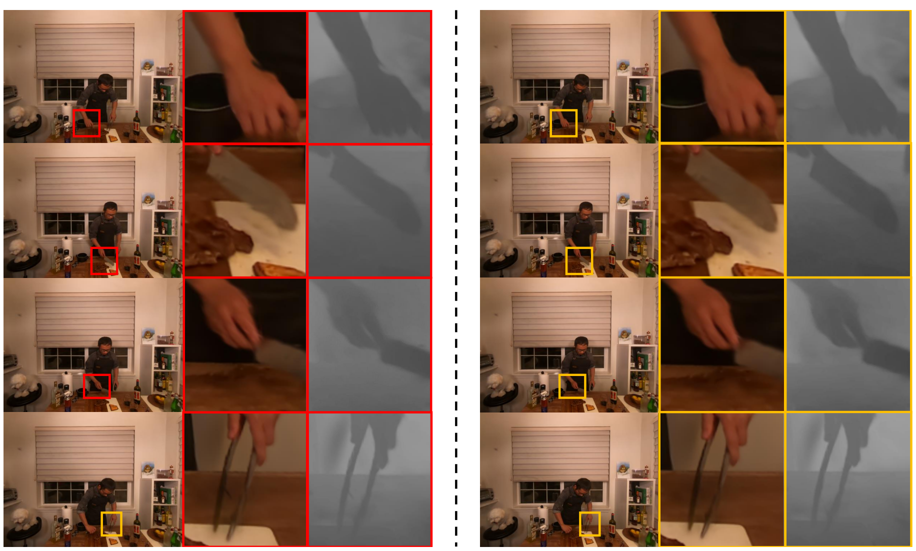
(a) *Flame Steak*

(b) *Cut Spinach*

Figure 10: Qualitative comparisons on DyNeRF dataset (Li et al., 2022). The left column shows the novel view rendered images and depth maps of a 4D Gaussian method Yang et al. (2023c), which suffers from artifacts in the dynamic regions. The right column shows the results of the same method while optimized with our flow supervision during training.



(a) *Sear Steak*



(b) *Cut Beef*

Figure 11: Qualitative comparisons on DyNeRF dataset Li et al. (2022). The left column shows the novel view rendered images and depth maps of a 4D Gaussian method (Yang et al., 2023c). While The right column shows the results of the same method while optimized with our flow supervision during training.

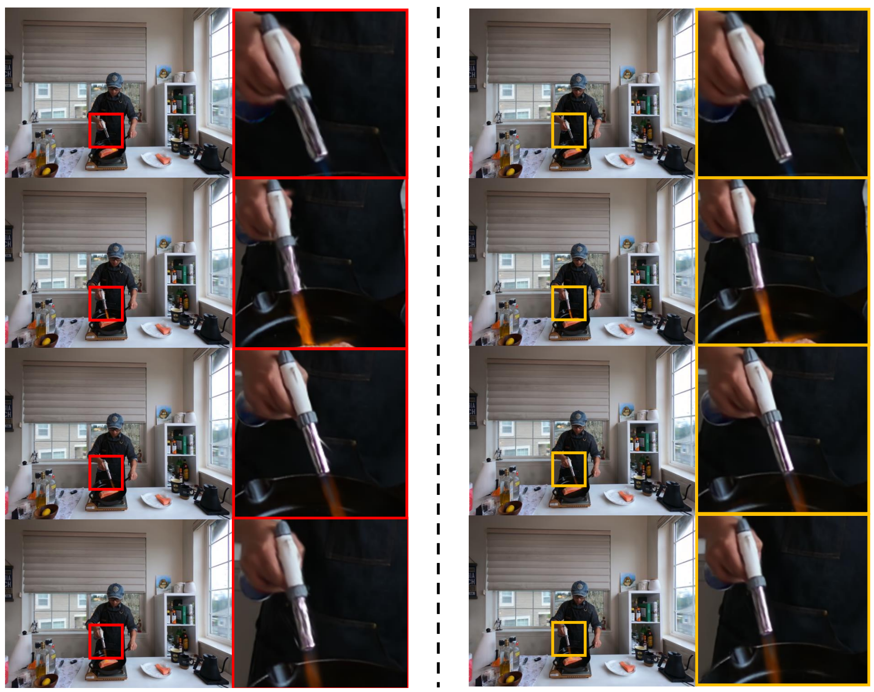


Figure 12: *Flame Salmon*

Figure 13: Qualitative comparisons on DyNeRF dataset (Li et al., 2022). Since the details of depth maps on *Flame Salmon* are hard to be recognized, we only compare the rendered images. The left column shows the novel view rendered images of a 4D Gaussian method (Yang et al., 2023c). While The right column shows the results of the same method while optimized with our flow supervision during training.