VELOCITY 4D GAUSSIAN SPLATTING: VELOCITY IS ALL YOU NEED

Anonymous authors
Paper under double-blind review

ABSTRACT

Novel view synthesis for dynamic scenes remains a significant challenge in computer vision, primarily due to the complexity introduced by the temporal dimension and diverse motion patterns. The additional dimensionality in 4D Gaussians complicates their constraint, resulting in motion inconsistencies and artifacts. Methods that model time independently struggle to decouple temporal and spatial dimensions effectively, while time-slicing techniques fall short in capturing temporal consistency. To address motion inconsistencies, we introduce the Velocity 4D Gaussian Splatting (V4DGS), a dynamic reconstruction approach that incorporates velocity consistency regularization into 4D Gaussians, leveraging temporal constraints alongside spatial priors. By using velocity to quantify the time dimension, we learn transferable temporal features that enable accurate motion prediction at new timestamps. Our velocity consistency regularization framework includes both temporal and spatial components, effectively addressing motion inconsistency by constraining velocity variations within an infinitesimal time interval. V4DGS excels in dynamic scenes with multiple objects and even reconstructing reflective regions. Through extensive evaluations on both monocular and multi-view motion datasets, V4DGS achieves real-time rendering at high resolutions, surpassing existing methods in both quantitative and qualitative metrics. V4DGS substantially improves quality (PSNR gain of 3.31 on the Plenoptic dataset), maintaining comparable rendering speeds and training time to previous methods.

INTRODUCTION

Reconstructing 3D scenes from 2D images and synthesizing novel viewpoints has long been a central task in computer vision and graphics. Novel View Synthesis (NVS) is critical for a wide range of industries, including film, gaming, and VR/AR. With the rapid advancements in generative AI, the demand for fast, realistic, real-time rendering has become more pressing than ever. NVS tasks can be divided into two categories: static scenes, where objects remain stationary across images Mildenhall et al. (2020); Knapitsch et al. (2017); Hedman et al. (2018); Barron et al. (2022), and dynamic scenes, where the content evolves over time Pumarola et al. (2021); Park et al. (2021b); Li et al. (2022b); Wu et al. (2020). While substantial progress has been made in static scene synthesis, efficiently and accurately generating novel views of dynamic scenes remains challenging due to the complexities introduced by the time dimension and diverse motion patterns.

To address the challenges of dynamic novel view synthesis, various approaches have been proposed to jointly model 3D scenes and their dynamics Du et al. (2021); Gao et al. (2021). NeRF has achieved notable success by representing scenes as implicit functions to synthesize novel views Mildenhall et al. (2020); Park et al. (2021a;b); Tretschk et al. (2021); Pumarola et al. (2021); Fang et al. (2022); Li et al. (2022b); Song et al. (2023). Its introduction of volumetric rendering techniques has enabled the effective connection between 2D images and 3D scenes. However, the original NeRF method is computationally expensive for both training and rendering, requiring dense sampling along millions of rays. Although some NeRF variants have reduced training time from days to minutes, rendering still suffers from significant delays, preventing real-time performance, even for static scenes.

3D Gaussian Splatting (3DGS) Kerbl et al. (2023) improves rendering speed by representing scenes as 3D Gaussian distributions. By replacing the complex volumetric rendering in NeRF with fast

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

rasterization techniques, 3DGS achieves real-time performance. Furthermore, 3DGS is extended to 4DGS to model motion in dynamic scenes. RealTime4DGS Yang et al. (2024) models time using independent marginal distributions. However, it represents 4D rotation with two entangled isoclinic quaternions. This tight coupling of spatial and temporal rotations complicates the process of decoupling time and space. 4DRotorGS Duan et al. (2024) models dynamics at each time point using time-sliced 4D Gaussians, but it does not fully leverage spatial geometric priors and lacks constraints on object motion in the temporal dimension. Deformation-based Gaussian splatting Yang et al. (2023); Huang et al. (2024); Wu et al. (2023) addresses this by jointly optimizing Gaussians in canonical space and deformation fields to capture both the geometric and dynamic features of a scene.

However, the added dimensionality in 4D Gaussians makes them harder to constrain Duan et al. (2024), leading to motion inconsistencies and artifacts. These inconsistencies are particularly problematic as they disrupt temporal coherence and degrade perceptual quality. The primary challenge in extending 3DGS to 4D representations for dynamic scenes is ensuring temporal consistency in the 4D space. If the Gaussians representing the same object or feature are inconsistent across time steps, it causes the illusion of object parts floating or moving independently. When the Gaussians' variation over time lacks smoothness, the motion of the represented features becomes jittery and irregular. Although current 4DGS methods Duan et al. (2024); Wu et al. (2023) separate time and space, they fail to capture consistent temporal features in the learned 4D Gaussians.

Our key observation is that to effectively enforce temporal consistency in 4D Gaussians, a new attribute should be introduced that quantifies temporal consistency and learns time-representative features. To better capture the characteristics of different time steps, we introduce velocity as an attribute of the 4D Gaussians. As illustrated in Fig. 2, we augment each Gaussian at every time step with an additional velocity attribute. An added benefit is that velocity links temporal and spatial consistency, as velocity inherently exhibits spatial locality. Our V4DGS allows us to quantify temporal features, leading to more precise and data-efficient representations.

To enforce temporal consistency in 4D Gaussian Splatting, we propose that the velocity within a unit time interval should remain constant in 4D space. Since velocity captures the motion characteristics at a given moment, we introduce temporal velocity regularization to ensure consistency across time intervals. As illustrated in Fig. 2, for continuously moving Gaussians, we capture velocity variations over a unit time interval as our temporal regularization. By penalizing irregular velocity changes, we constrain non-smooth transitions in the motion of Gaussians over time. Our temporal velocity consistency regularization enables 4D Gaussians to learn more transferable temporal features, allowing 4DGS to accurately predict Gaussian motion even with sparse input data.

Theorem 1. In dynamic motion modeling, assuming constant velocity within a fixed unit time interval simplifies complex trajectories by treating motion as linear and uniform. As the interval decreases, this approximation increasingly aligns with the object's true, continuous motion trajectory, converging to a more accurate representation. See the proof in Sec. .

Temporal velocity consistency regularization ensures consistency among Gaussians representing different objects within a given time interval. For each Gaussian, those in its neighboring region are treated as approximations of the same object or feature. To maintain spatial consistency, we further apply velocity constraints across neighboring regions. However, these spatial velocity constraints should extend over a unit time interval, as single-moment regularization is insufficient to enforce temporal consistency. To address this, as shown in Fig. 2, we begin by capturing and calculating the velocity variations within the Gaussian point's neighborhood Ω at a single time step. We then extend this spatial constraint from a single moment to a higher-dimensional unit time interval,

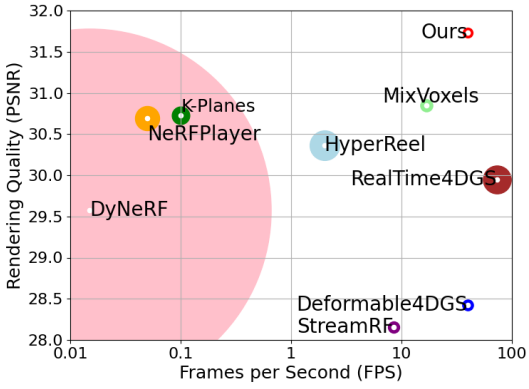


Figure 1: **PSNR vs. Training Time vs. FPS.** The size of each dot represents the corresponding training time.

yielding the final velocity spatial regularization. This approach within V4DGS effectively improves the rendering quality of moving objects while maintaining fast inference speeds for dynamic scenes.

To accurately predict 4D Gaussian motion, V4DGS leverages a deformation field to learn Gaussian attributes at subsequent time steps. The deformation field is especially effective at integrating topological spatial priors, making it well-suited for this purpose. Spatial consistency forms the basis for temporal consistency and plays a crucial role in ensuring the motion consistency of 4D Gaussians. After obtaining consistent spatial variations, we calculate the velocity of the 4D Gaussians by differentiating their static displacement. Furthermore, our velocity consistency regularization constrains velocity variations within a unit time interval, integrating both temporal and spatial components. Unlike methods that rely solely on local constraints, we extend the spatial regularization of velocity to incorporate the temporal dimension. This elevates the spatial regularization from being applied at a single moment to covering a continuous unit time interval.

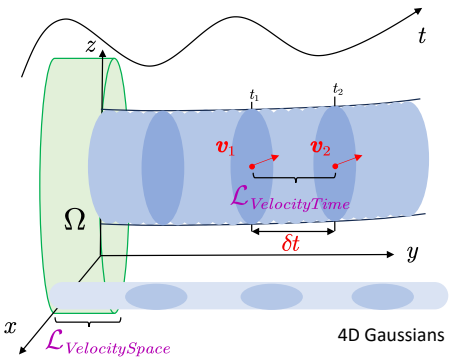


Figure 2: **A Simplified Illustration of the Velocity Consistency Regularization.** First, we compute the current velocity of the 4D Gaussians by differentiating their static displacement. Next, we apply temporal regularization by capturing the velocity variation over a unit time interval. Last, we enhance spatial consistency by extending the velocity variation within neighboring regions and scaling it to the same unit time interval as part of our spatial regularization process.

Our velocity consistency regularization substantially enhances rendering quality, capturing finer details and delivering more realistic reconstructions of reflective surfaces (see Fig. 3). On the Plenoptic Dataset, V4DGS achieves a PSNR improvement of 3.31 over the deformation-based baseline. Similarly, on the D-NeRF Dataset, V4DGS raises the best PSNR from 32.99 to 34.26. Moreover, our velocity optimization strategy simplifies the model and accelerates convergence by approximating velocity through static displacement, a method validated both theoretically and experimentally. Importantly, our velocity consistency regularization significantly enhances rendering quality without increasing training time, while preserving efficient inference speeds (see Fig. 1). In summary, our contributions are as follows:

- The introduction of **velocity consistency regularization** combined with spatial priors, effectively addressing **the motion inconsistencies in 4DGS**.
- The first 4DGS algorithm enforces **temporal consistency** by constraining the velocity of 4D Gaussians to constant within **a unit time interval**.
- An optimization method of 4D Gaussian properties, dynamically enhancing spatial velocity constraint both at **the start and end** of each time interval for **motion consistency**.
- A fast, differentiable GPU rendering approach for **dynamic scenes**, unlike 3DGS, that substantially improves **rendering quality** while maintaining competitive FPS and training time.

Related Work

Dynamic Scene Representation with NeRFs. Generating novel views of dynamic scenes from 2D images is a challenging task, largely due to the need to capture temporal relationships across different moments. Unlike traditional methods that rely on geometric structures to handle dynamics Li et al. (2012); Collet et al. (2015); Kanade et al. (1997); Zitnick et al. (2004), recent work often represents dynamic scenes using Neural Radiance Fields (NeRFs) Mildenhall et al. (2021). One approach implicitly captures dynamics by incorporating temporal inputs or latent codes Du et al. (2021); Gao et al. (2021), while another explicitly models deformation fields, mapping 3D points from arbitrary timestamps to a canonical space Park et al. (2021a;b); Tretschk et al. (2021); Pumarola et al. (2021); Fang et al. (2022). Additionally, methods like scene decompositions Song et al. (2023); Fridovich-Keil et al. (2023); Cao & Johnson (2023); Shao et al. (2023), keyframe extraction Attal et al. (2023),



Figure 3: **We introduce Velocity 4D Gaussian Splatting, a novel method designed for real-time rendering of dynamic scenes at high image resolutions.** Our approach surpasses existing methods in rendering quality while offering shorter training times and faster inference speeds, as demonstrated on the Plenoptic Dataset Li et al. (2022b).

and flow field estimation Li et al. (2021); Guo et al. (2023); Tian et al. (2023) have been explored for dynamic scene representation. However, NeRF-based dynamic scene modeling often incurs additional storage and time costs.

Dynamic Gaussian Splatting. 3D Gaussian Splatting (3DGS) Kerbl et al. (2023) for novel view synthesis effectively harnesses the mathematical properties of Gaussians to represent complex geometry and lighting in a more natural way Zhang et al. (2022); Xu et al. (2022); Abou-Chakra et al. (2022); Luiten et al. (2024). However, storing the data for every 3D Gaussian primitive at each timestamp results in a linear increase in memory usage, which poses challenges for real-time dynamic reconstruction. Spacetime Li et al. (2023) improves upon 3D Gaussians by adding temporal opacity and parameterized motion/rotation, but it still requires tracking each individual Gaussian. A series of deformable-based Gaussian splatting methods Yang et al. (2023); Huang et al. (2024); Wu et al. (2023) incorporate topological invariance priors. Different from these methods, we address the challenge of inconsistent motion in dynamic scenes by introducing velocity consistency regularization. Our V4DGS framework effectively resolves inconsistencies in the Gaussian representations of the same object or feature across consecutive time steps.

METHOD

DEFORMATION FIELD

To construct a 4D Gaussian Splatting (4DGS) framework that maintains spatiotemporal consistency, we opt to incorporate an additional deformation field to capture variations in position and shape, learning 4D Gaussian distributions in a canonical space. Our approach allows 4D Gaussians to effectively model dynamic scenes without sacrificing physical coherence. Extending 3D Gaussian Splatting (3DGS) Kerbl et al. (2023) to 4D is a complex challenge. The original motivation behind 3DGS was to provide an efficient solution for rendering and representing 3D scenes, particularly in static environments. By leveraging the spatial representation of 3D Gaussians alongside rasterization techniques, 3DGS enables continuous and efficient rendering of both geometry and color.

Introducing time-conditioned learnable parameters to each 3D Gaussian distribution would cause these parameters to evolve over time, making the attributes of each Gaussian dynamic rather than fixed. As a result, the 3D Gaussians representing the same object or feature exhibit discontinuities in shape or position across consecutive time points within the same scene. The original goal of differentiable Gaussian rasterization is to deliver stable and predictable rendering by maintaining fixed

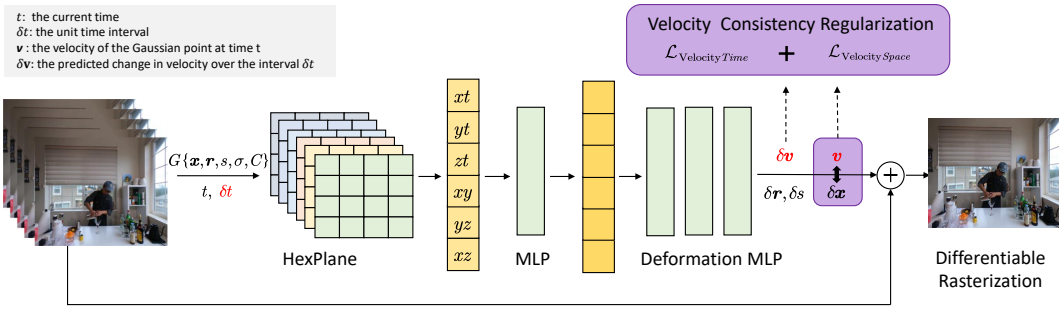


Figure 4: **Framework Overview.** The 3D Gaussians G are processed through six HexPlanes to compute voxel features, which are subsequently transformed into a full 4D Gaussian representation using a lightweight MLP. The deformation network then adapts the 4D Gaussians into 3D Gaussians at a given timestamp t. These deformed Gaussians are finally projected onto the rendered image.

3D Gaussians. Applying time-based parameters undermines this stability, making rendering outcomes dependent on temporal changes. This contradicts the core principle of Gaussian rasterization, which emphasizes consistency and simplicity, and disrupts the geometric continuity of the scene. In summary, while time-conditioned parameters introduce flexibility Yang et al. (2024), they cause 3D Gaussians to vary across time, breaking spatial consistency and predictability—key elements in the design of 3D Gaussian Splatting.

Similarly, the rotor-based 4DGS Duan et al. (2024) does not fully capitalize on the spatial prior information inherent in the time series. While the rotor-based 4DGS effectively decouples time and space, it fails to adequately incorporate the spatial priors from previous time steps into the current moment. Given that spatial consistency is a crucial component of motion consistency, this limitation leads to motion inconsistency when the rotor-based 4DGS is applied to video generation tasks.

To preserve the physical meaning of 3D Gaussian distributions when modeling dynamic scenes, we employ the deformation field Yang et al. (2023) that learns four-dimensional spatiotemporal features. Deformation field Yang et al. (2023) takes as input a set of monocular dynamic scene images, along with camera poses and timestamps calibrated using structure from motion (SfM) Schonberger & Frahm (2016), which also generates a sparse point cloud. Based on these inputs, we can obtain a set of 3D Gaussian distributions, represented as $G(\mathbf{x}, \mathbf{r}, s, \sigma, C)$, where $\mathbf{x}$ denotes the center position, $\mathbf{r}$ is the rotation (modeled by a quaternion), s is the scaling factor, and σ represents opacity. The color C of each 3D Gaussian is represented using spherical harmonics (SH).

We then leverage the deformation field model time-varying 3D Gaussians by decoupling their properties from a learned deformation field. The deformation network $\mathcal{D}$ consists of an MLP that takes the position of the 3D Gaussian and the current time t as inputs, and outputs adjustments to position ($\delta\mathbf{x}$), rotation ($\delta\mathbf{r}$), and scaling (δs). The deformed 3D Gaussian $G(\mathbf{x}+\delta\mathbf{x}, \mathbf{r}+\delta\mathbf{r}, s+\delta s, \sigma, C)$ is efficiently rendered using a differentiable Gaussian rasterization pipeline. To more effectively capture scene geometry while ensuring computational efficiency, we employ the adaptive optimization method Kerbl et al. (2023) that dynamically adjusts the Gaussian distributions, refining the continuous spatial priors within the deformation field.

4D Gaussian Deformation Network

As shown in Fig. 4, we begin by using HexPlane Fridovich-Keil et al. (2023) to decouple temporal and spatial features, effectively decomposing the space and time dimensions into six distinct 2D planes. Space and time features from the six planes are then combined into a unified 4D Gaussian spatiotemporal representation using a lightweight multi-layer perceptron (MLP). This representation is subsequently fed into the Gaussian deformation network $\mathcal{D}$ to predict the deformation of the 4D Gaussians at the next time step. In contrast to previous methods, our 4DGS incorporates the unit time interval δt at the input stage. Correspondingly, new Gaussian attributes, *i.e.* $\mathbf{v}$ and the change in velocity $\delta\mathbf{v}$ over the unit time interval, are introduced at the output of the deformation network. The changes in the attributes of 4D Gaussians can be represented as:

$$\Delta G = \mathcal{D}\{G, t, \delta t\} = \{\delta\mathbf{x}, \delta\mathbf{r}, \delta s, \sigma, C, \mathbf{v}, \delta\mathbf{v}\}. \tag{1}$$

Proof for Theorem 1. For a unit time interval δt, the velocity is approximated as:

$$\mathbf{v}_{\delta t}(t) = \frac{\mathbf{x}(t+\delta t) - \mathbf{x}(t)}{\delta t} \tag{2}$$

Within each fixed unit time interval δt, the object's velocity is assumed to remain constant. This implies:

$$\mathbf{x}(t+\delta t) \approx \mathbf{x}(t) + \mathbf{v}(t) \cdot \delta t = \mathbf{x}(t) + \mathbf{v}(t) \tag{3}$$

To analyze the accuracy of the finite difference approximation, consider the Taylor series expansion of the position function around time t:

$$\mathbf{x}(t+\delta t) = \mathbf{x}(t) + \mathbf{v}(t) \cdot \delta t + \frac{1}{2} a(t) \cdot (\delta t)^2 + \mathcal{O}((\delta t)^3) \tag{4}$$

where:

$$a(t) = \frac{d\mathbf{v}(t)}{dt} \tag{5}$$

is the acceleration at time t. Dividing both sides of the Taylor series by δt:

$$\frac{\mathbf{x}(t+\delta t) - \mathbf{x}(t)}{\delta t} = \mathbf{v}(t) + \frac{1}{2} a(t) \cdot \delta t + \mathcal{O}((\delta t)^2) \tag{6}$$

This gives:

$$\mathbf{v}_{\delta t}(t) = \mathbf{v}(t) + \frac{1}{2} a(t) \cdot \delta t + \mathcal{O}((\delta t)^2) \tag{7}$$

Taking the limit as δt approaches zero:

$$\lim_{\delta t \to 0} \mathbf{v}_{\delta t}(t) = \lim_{\delta t \to 0} \left(\mathbf{v}(t) + \frac{1}{2} a(t) \cdot \delta t + \mathcal{O}((\delta t)^2) \right) = \mathbf{v}(t) \tag{8}$$

This demonstrates that:

$$\lim_{\delta t \to 0} \mathbf{v}_{\delta t}(t) = \mathbf{v}(t) \tag{9}$$

Hence, the finite difference approximation $\mathbf{v}_{\delta t}(t)$ converges to the true instantaneous velocity $\mathbf{v}(t)$ as the time interval δt approaches zero.

During experiments, we observe that the displacement $\delta \mathbf{x}$ predicted by the deformation field closely approximates the velocity $\mathbf{v}$. To accelerate the rendering process and improve suitability for large-scale real-time applications, we propose using $\delta \mathbf{x}$ directly instead of relying on an independent MLP to predict $\mathbf{v}$.

Proposition 1. For a Gaussian point, the velocity $\mathbf{v}$ over a unit time interval δt is defined as:

$$\mathbf{v} = \frac{\mathbf{x}(t_2) - \mathbf{x}(t_1)}{t_2 - t_1} = \frac{\mathbf{x}(t_2) - \mathbf{x}(t_1)}{\delta t} = \delta \mathbf{x}. \tag{10}$$

where $\delta t = t_2 - t_1$ simplifies the velocity to the displacement $\delta \mathbf{x}$.

This allows us to efficiently obtain the fully deformed 4D Gaussian:

$$G^* = \{\mathbf{x} + \delta \mathbf{x}, \mathbf{r} + \delta \mathbf{r}, s + \delta s, \sigma, C, \mathbf{v}, \delta \mathbf{v}\}. \tag{11}$$

Optimization Schema

We now present our velocity consistency regularization constraint in detail. For 4D Gaussians, we first apply velocity consistency regularization along the temporal dimension. After calculating the velocity of the 4D Gaussians at the current time step, we then capture the velocity variations within a unit time interval. These variations are more significant than the velocity at a single moment because, with the added dimensionality, 4D Gaussians are harder to constrain compared to 3D Gaussians. Previous 4DGS methods Wu et al. (2023); Yang et al. (2024); Duan et al. (2024) struggle to synchronize the Gaussians for the same object or feature across different time steps, resulting in motion inconsistencies and artifacts. By constructing temporal velocity variations, our approach tightly integrates temporal and spatial information. The implementation of temporal velocity consistency regularization effectively addresses the challenge of maintaining temporal consistency. At a given

unit time interval δt, the velocity consistency regularization along the temporal dimension can be expressed as:

$$\mathcal{L}_{\text{VelocityTime}} = \frac{1}{N}\sum_{i=1}^{N}\delta\mathbf{v}_i. \tag{12}$$

After resolving the issue of temporal inconsistency, we further enhance spatial consistency within each time step. Our 4DGS framework applies velocity consistency regularization across both temporal and spatial dimensions. To extend spatial velocity constraints into the temporal dimension, we design a regularization approach that enforces motion consistency between the start and end of a unit time interval. For each Gaussian point, we approximate the velocity at both the start and end of the interval by doubling the current velocity of neighboring Gaussians within its local region. This enhanced velocity space regularization effectively addresses inconsistent motion of the same object or feature across time.

Proposition 2. To ensure motion consistency across a unit time interval, the velocity $\mathbf{v}$ of each Gaussian i should align with the average velocity of its K nearest neighbors Ω_i: $\mathbf{v}_i - \frac{1}{K}\sum_{j\in\Omega_i}\mathbf{v}_j$. By scaling both sides by a factor of 2, we extend this consistency across the entire unit time interval: $2\mathbf{v}_i - \frac{2}{K}\sum_{j\in\Omega_i}\mathbf{v}_j$.

Theorem 2. Given K nearest 4D points from the i-th Gaussian point neighboring space Ω_i, the velocity space regularization loss ensures spatiotemporal consistency by aligning each Gaussian's velocity with the average velocity of its neighbors, both at the start and the end of a unit time interval. The regularization loss is defined as:

$$\mathcal{L}_{\text{VelocitySpace}} = \frac{1}{N}\sum_{i=1}^{N}\left\|2\mathbf{v}_i - \frac{2}{K}\sum_{j\in\Omega_i}\mathbf{v}_j\right\|_1. \tag{13}$$

This formulation extends the spatial velocity constraint into the temporal dimension, enforcing smooth and coherent motion across dynamic scenes.

Proof. Refer to Proposition 1, we obtain the velocity in a unit time interval. To ensure motion consistency not just at a single moment but across a unit time interval, we must consider how velocities evolve from the start to the end of the interval. This involves enforcing that the motion (velocity) of each Gaussian is consistent with the motion of its neighbors throughout the interval. Refer to Proposition 2, we formulate the velocity consistency condition. This term quantifies the absolute difference between the scaled velocity of Gaussian i and the scaled average velocity of its neighbors. By minimizing this loss, we enforce that each Gaussian's velocity is closely aligned with the average velocity of its local neighborhood, both at the start and end of the unit time interval. To apply this consistency across the entire set of Gaussians, we sum the regularization terms for all N Gaussians and normalize by N. The inclusion of the scaling factor 2 in both terms of the regularization loss serves to extend the spatial velocity constraint into the temporal dimension. Specifically, it ensures that the velocity of each Gaussian is consistent with the average velocity of its neighbors at both the beginning and the end of the unit time interval. This dual enforcement is crucial for maintaining smooth and coherent motion across dynamic scenes, preventing abrupt changes that could lead to visual artifacts. Scaling by values greater than 2 (e.g., 3) overemphasizes individual velocity, causing overshooting and oscillations, which introduce artifacts and destabilize optimization. Scaling by values less than 2 (e.g., 1.5) underestimates alignment, leading to motion inconsistencies and reduced regularization effectiveness. The scaling factor of 2 ensures smooth transitions by aligning velocity with both the start and end of the unit time interval, while also synchronizing each Gaussian's motion with its neighbors to maintain overall motion coherence.

Finally, the complete velocity space regularization loss combines both temporal and spatial consistency, where λ is hyperparameters determined through experimentation:

$$\mathcal{L}_{\text{Velocity}} = \mathcal{L}_{\text{VelocityTime}} + \lambda\mathcal{L}_{\text{VelocitySpace}}. \tag{14}$$

Notably, our velocity consistency regularization approach is distinct from optical flow. While optical flow is computationally intensive and unsuitable for real-time rendering of dynamic scenes. 4DGS velocity consistency regularization is imposed on Gaussian primitives in 3D space, while optical flow is formulated on image pixels as a 2D image-plane constraint. No existing 4DGS method that incorporates optical flow has successfully resolved the issue of temporal inconsistencies.

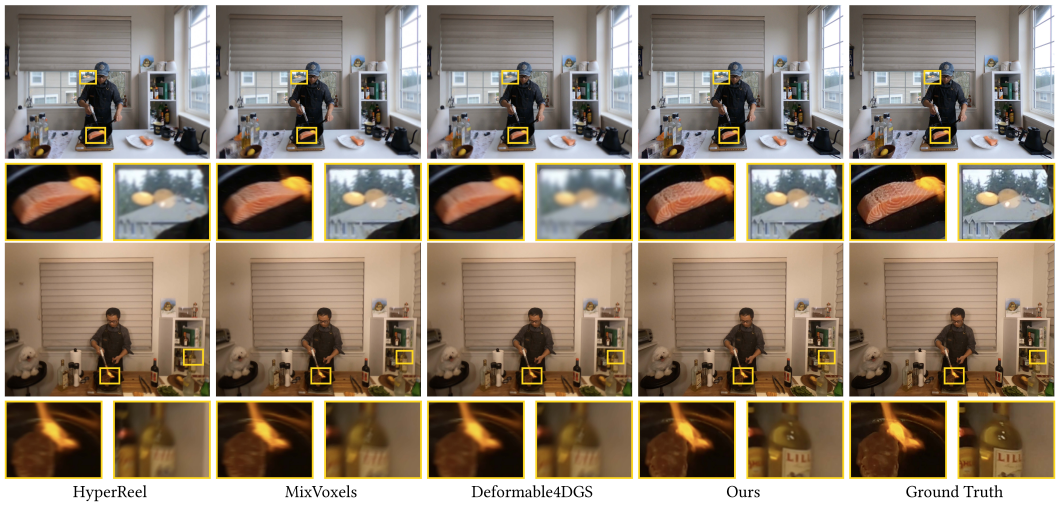


Figure 5: **Qualitative Comparison on Plenoptic Video Dataset**. Our approach outperforms prior work by recovering finer details in dynamic regions, *e.g.*, the color at the center of the flames. Additionally, it renders sharper static regions, *e.g.*, the light reflections on the glass bottle in row 2.

EXPERIMENTS

Datasets *Plenoptic Video Dataset Li et al. (2022b)* captures real-world scenes using a multi-view GoPro camera system. Six different scenes are used for baseline evaluations. Image resolutions are 1352×1014. The dataset features several challenging aspects, such as the unexpected appearance of flames, moving shadows, and a mix of translucent and reflective materials. *D-NeRF Dataset Pumarola et al. (2021)* is a synthetic dataset consisting of monocular videos, posing a substantial challenge because each timestamp is captured from a single camera viewpoint. All images have been resized to a standard 800×800 resolution.

Table 1: **Quantitative Comparison on Plenoptic Video Dataset**. We compare our approach with both NeRF-based and Gaussian-based methods. *: trained on 8 GPUs and tested only on the Flame Salmon scene. The best, second best, and third best results are highlighted.

ID	Method	PSNR↑	SSIM↑	LPIPS↓	Train↓	FPS↑
a	DyNeRF Li et al. (2022b)*	29.58	-	0.08	1344 h	0.015
b	StreamRF Li et al. (2022a)	28.16	0.85	0.31	79 min	8.50
c	HyperReel Attal et al. (2023)	30.36	0.92	0.17	9 h	2.00
d	NeRFPlayer Song et al. (2023)	30.69	-	0.11	6 h	0.05
e	K-Planes Fridovich-Keil et al. (2023)	30.73	0.93	0.07	190 min	0.10
f	MixVoxels Wang et al. (2023)	30.85	0.96	0.21	91 min	16.70
g	Deformable4DGS Wu et al. (2023)	28.42	0.92	0.17	72 min	39.93
h	RealTime4DGS Yang et al. (2024)	29.95	0.92	0.16	8 h	72.80
i	Ours	31.73	0.92	0.15	72 min	39.93

Results Evaluation on Plenoptic Video Dataset As shown in Tab. 1, previous methods struggle to balance rendering velocity and quality. The primary challenges include slow volumetric rendering (*a*-*f*), the high computational cost of querying neural network components (*c*, *d*, *g*), and ineffective modeling of dynamic motion (*a*, *c*, *g*). In contrast, our approach offers significant improvements in both velocity and quality. Our method outperforms existing approaches in rendering high-resolution videos (1352×1014), delivering the best rendering quality. Compared to the baseline Deformable4DGS, our 4DGS achieves a 3.16% improvement in PSNR. As shown in Fig. 5, our approach produces clearer and more detailed reconstructions of dynamic regions compared to the baseline models. Our velocity consistency regularization, especially the temporal consistency constraint, effectively enforces consistency in Gaussians for the same object or feature across different time steps, enabling to learn more robust and transferable temporal features. While maintaining rendering efficiency, V4DGS substantially enhances rendering quality, capturing richer details and producing highly realistic reflections.

Table 2: **Quantitative Comparison on D-NeRF Dataset**. Our approach surpasses both NeRF-based and Gaussian-based baselines in PSNR.

Method	PSNR↑	SSIM↑	LPIPS↓	Train↓	FPS↑
D-NeRF Pumarola et al. (2021)	29.17	0.95	0.07	24 h	0.13
TiNeuVox Fang et al. (2022)	32.87	0.97	0.04	28 min	1.60
K-Planes Fridovich-Keil et al. (2023)	31.07	0.97	0.02	54 min	1.20
Deformable4DGS Wu et al. (2023)	32.99	0.97	0.05	13 min	104.00
RealTime4DGS Yang et al. (2024)	32.71	0.97	0.03	10 min	289.07
Ours	34.26	0.97	0.03	13 min	104.00

Evaluation on D-NeRF Dataset As analyzed in Tab. 2, our method achieves the highest rendering quality among all approaches. V4DGS improves PSNR from 32.99 to 34.26, while maintaining a rendering velocity of 104 FPS, ensuring real-time performance without compromise. Fig. 6 demon-

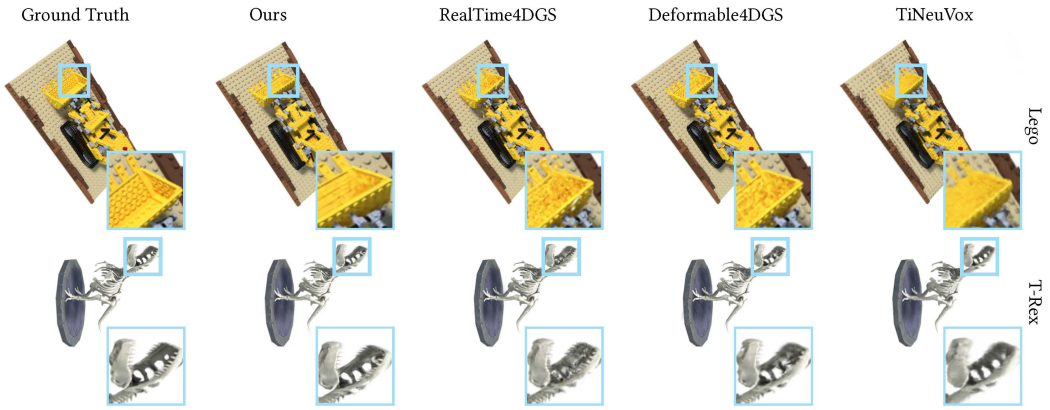


Figure 6: **Qualitative Comparison on D-NeRF Dataset.** As highlighted in specific regions, our method reconstructs scenes with less noise and greater detail, *e.g.*, the teeth of *T-Rex* (even rendering the shadow of the upper teeth).

strates how our method outperforms the baseline in reducing temporal inconsistencies and improving reconstruction quality. For example, the patterns on the *Lego* bulldozer remain well-aligned, with no visible artifacts. Furthermore, in the *T-Rex* scene, the shape and arrangement of the upper teeth are accurately reconstructed. The color in the shaded areas of the upper jaw is also much closer to the ground truth.

Ablation Studies Temporal Consistency Regularization. On the D-NeRF dataset, as shown in Tab. 3 (*Full*), the addition of velocity temporal regularization significantly enhances rendering quality across different datasets. Specifically, on the scene *Trex*, it yields a PSNR improvement of over 0.5, while on the *Jumping Jacks*, it achieves a PSNR gain of nearly 0.4. Overall, velocity temporal regularization effectively mitigates temporal inconsistencies in the representation of 4D Gaussian points, enabling the model to learn more precise and transferable temporal features. Even with sparse input data, our V4DGS method accurately predicts 4D Gaussian motion, successfully addressing motion inconsistencies in dynamic rendering.

Table 3: **Ablation Study with Quantitative Comparison on D-NeRF Dataset**. We validate velocity consistency regularization on rendering quality PSNR.

ID	Ablation Items		D-NeRF				
	Space	Time	Jumping Jacks	Hell Warrior	Mutant	Stand Up	Trex
a			35.33	28.70	37.87	37.91	34.00
b	✓		35.36	29.00	37.71	37.98	33.85
Full	✓	✓	**35.74**	**29.21**	**37.87**	**38.15**	**34.37**

Space Consistency Regularization. The states of neighboring Gaussians in 4D space change freely, which increases the difficulty of optimization and the redundancy of the model. Our 4D consistency regularization enforces local consistency across both spatial and temporal dimensions. Velocity spatial regularization enhances local consistency by constraining the states of neighboring Gaussians within the 4D local space. This approach simplifies the model and accelerates convergence. This is demonstrated in Tab. 3. With the inclusion of velocity spatial regularization in Tab. 3 (*b*), 3 out of 5 datasets show improvement. Although two datasets experienced a slight decline, the overall gains outweigh the losses. For example, in the *Hell Warrior* scene, the PSNR increased from 28.70 to 29.00. Notably, unlike regularization that limits constraints to local spatial regions, we extend velocity spatial regularization to the temporal dimension. By expanding spatial constraints from a single moment to a unit time interval, our approach effectively addresses temporal inconsistencies that cannot be resolved by spatial constraints alone.

Conclusion

In this work, we introduce the Velocity 4D Gaussian Splatting, a framework designed for high-quality, real-time modeling of 4D dynamic scenes. V4DGS integrates velocity consistency regularization with deformation fields, leveraging learning transferable temporal features and spatial priors by constraining velocity variation within a unit time interval to accurately predict 4D Gaussian motion. Our approach includes both temporal and spatial regularization components, effectively resolving motion inconsistency issues. V4DGS not only significantly outperforms existing methods but also maintains comparable rendering speeds and training time.

REFERENCES

Jad Abou-Chakra, Feras Dayoub, and Niko Sünderhauf. Particlenerf: Particle based encoding for online neural radiance fields in dynamic scenes. *arXiv preprint arXiv:2211.04041*, 2022.

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.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5470–5479, 2022.

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.

Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (ToG)*, 34(4):1–13, 2015.

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)*, pp. 14304–14314. IEEE Computer Society, 2021.

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 *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

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.

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

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018.

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*, pp. 4220–4230, 2024.

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

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

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.

Hao Li, Linjie Luo, Daniel Vlasic, Pieter Peers, Jovan Popović, Mark Pauly, and Szymon Rusinkiewicz. Temporally coherent completion of dynamic shapes. *ACM Transactions on Graphics (TOG)*, 31(1):1–11, 2012.

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, 2022a.

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, 2022b.

Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. *arXiv preprint arXiv:2312.16812*, 2023.

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.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024.

B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, 2020.

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.

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. *ACM Transactions on Graphics (TOG)*, 40(6):1–12, 2021b.

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.

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.

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.

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*, pp. 17903–17913, 2023.

Edith 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.

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, 2023.

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.

Minye Wu, Yuehao Wang, Qiang Hu, and Jingyi Yu. Multi-view neural human rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1682–1691, 2020.

Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5438–5448, 2022.

Zeyu Yang, Hongye Yang, Zijie Pan, Xiatian Zhu, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *International Conference on Learning Representations (ICLR)*, 2024.

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.

Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. In *SIGGRAPH Asia 2022 Conference Papers*, pp. 1–12, 2022.

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.