Motion Decoupled 3D Gaussian Splatting for Dynamic Object Representation

Xiao Hu, Libo Long, Jochen Lang✉

University of Ottawa, Canada,
{xhu008, llong014, jlang}@uottawa.ca

Abstract

Dynamic object modeling is a critical challenge in 3D scene reconstruction. Previous methods typically maintain a canonical space to represent the object model, and a deformation field to express the object motion. However, this approach fails when the object undergoes large motions. The position variation caused by significant motion not only complicates the establishment of a canonical space, but also misleads the interpretation of the deformation field. To overcome the above weaknesses, we propose Motion Decoupled Dynamic 3D Gaussian Splatting (M5D-GS), the first 3D-GS model that separates motion and deformation modeling for dynamic object representation with large motion from a monocular camera. M5D-GS increases the practicality of 3D-GS, as it is common for objects to move, rotate, and deform simultaneously. Current datasets only contain object deformations with slight motions. We introduce a pipeline to reuse current datasets by adding large motions into the scene. We also introduce a new benchmark featuring several new synthetic scenes with complex motions, some scenes augmented from previous datasets, and some real world recorded scenes. Our M5D-GS significantly increases the accuracy under large motion while maintaining high rendering speed, which makes it suitable for dynamic object representation tasks including 4D novel view synthesis and real-time rendering.

Code — https://github.com/haliphinx/M5D-GS

1 Introduction

3D scene representation is the foundation of many 3D computer vision tasks, including but not limited to novel view synthesis (Debevec, Yu, and Borshukov 1998), 3D modeling (Schonberger and Frahm 2016), and motion/animation rendering. These tasks are commonly involved in augmented reality (AR), virtual reality (VR), and potentially autonomous driving, where real-time rendering and dynamic objects are two critical challenges. Traditional methods usually use point clouds (Lin, Kong, and Lucey 2018), meshes (Gao et al. 2020), occupancy grids (Mescheder et al. 2019), or polygons (Nan and Wonka 2017) to represent the 3D scene. Although these representations may provide additional physical attributes, they are either expensive to acquire or lacking precision, especially in dynamic scenes.

Neural Radiance Fields (NeRF) (Mildenhall et al. 2021) introduced the volume rendering technique to bridge 3D scenes with projected 2D images. This differentiable rendering strategy allows NeRF to implicitly represent a 3D scene with a function based on the view pose, which can be fitted by a deep learning model.

NeRF represents the scene continuously, enabling high-precision 3D scene generation. However, the main bottleneck of NeRF is the training and rendering costs. The original NeRF requires hours or even days to be trained for a single scene. Multiple variants of NeRF (Chen et al. 2022; Wang et al. 2022, 2023; Yu et al. 2021) have been published to increase training speed, but it is still challenging to find the balance between speed and accuracy. 3D Gaussian Splatting (3D-GS) (Kerbl et al. 2023) represents a 3D scene with a set of 3D Gaussian points and uses a differentiable splatting technique to project the 3D scene into a 2D image. This approach represents the scene explicitly and saves memory by describing only the occupied points, making it easier for visualization and editing. 3D-GS achieves comparable accuracy with NeRF but reduces time complexity significantly, allowing for real-time rendering.

Various follow-up works to NeRF (Li et al. 2023, 2022a) explore using NeRF for dynamic scenes. Different 3D graphic structures, including voxel-grids (Fang et al. 2022; Shao et al. 2023) and planes (Cao and Johnson 2023; Fridovich-Keil et al. 2023), have been used to boost the training speed and provide extra spatial constraints. However, they still suffer from slow processing speeds, and low accuracy. Dynamic 3D-GS (Wu et al. 2024; Yang et al. 2024b; Sun et al. 2024; Yang et al. 2024a; Huang et al. 2024) incorporate deformation fields into static scenes. These approaches mostly share the fundamental idea of maintaining a static canonical space to represent the 3D structure and a time-related deformation field to represent motion. Both components are optimized together to achieve state-of-the-art performance, which leads to potential problems in large motion scenarios. First, large motion increases the difficulty of establishing a canonical space. Second, this joint optimization strategy can lead to ambiguities in understanding large motions and it can easily converge to local minima.

To overcome the limitations of 3D-GS in large motion scenes, we propose Motion Decoupled Dynamic 3D Gaussian Splatting (M5D-GS), which separates the learning of

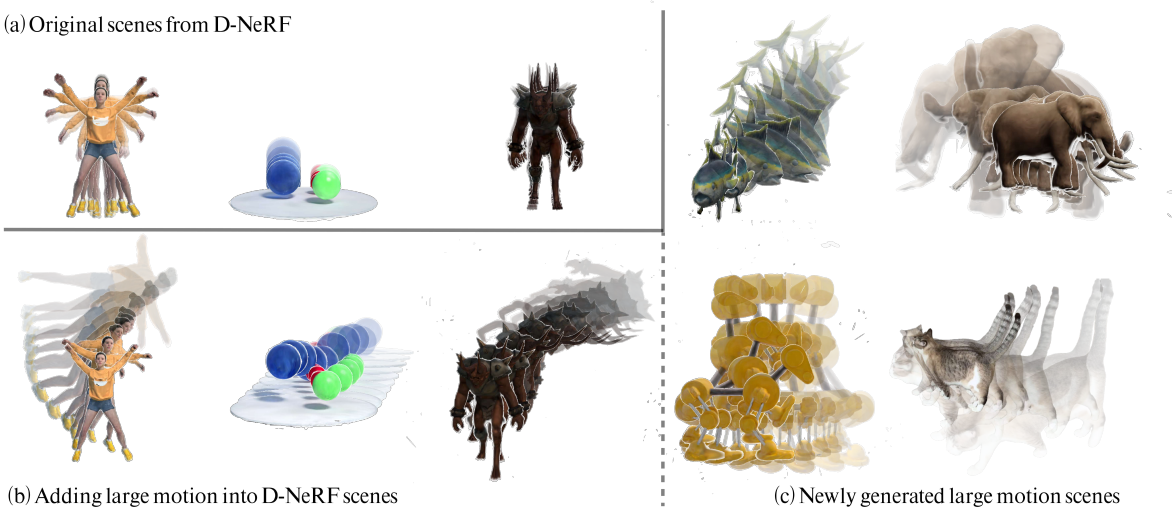

Figure 1: Visualization of scene motion. Each scene is represented by a stack of 10 frames uniformly sampled over time. Greater overlap indicates smaller motion within the scene. (a) Scenes from the D-NeRF dataset, where the frames are mostly overlapping, indicating minimal motion. (b) Scenes with large motions added to the original scenes in (a). (c) Newly generated and real world scenes.

object motion and deformation in dynamic environments. We investigate the motion range in a widely used benchmark D-NeRF (Pumarola et al. 2021). As shown in Figure 1 (a), the motion in each scene is minimal. To address this, we introduce a pipeline that adds object-level motions to existing benchmarks, allowing us to reuse previous datasets without introducing additional bias (Figure 1 (b)). We also generate new scenes containing large motions and real world recorded scenes (Figure 1 (c)) to effectively evaluate the performance in a total of ten scenarios involving complicated motion. Our proposed M5D-GS (Figure 2) decouples the representation of 3D structure, object level motion, and per-Gaussian local deformation, which significantly improves performance over previous state-of-the-art methods in large motion scenes. In summary, our main contributions are:

- We propose M5D-GS, a 3D-GS-based framework designed to handle complex motions for dynamic object representation with high-fidelity and real-time rendering.
- We design a framework to explicitly estimate the object motion and the object deformation separately, and a coarse-to-fine matching strategy to handle large motion initialization.
- We introduce an easy-to-use pipeline that converts existing benchmarks into large motion scenarios, facilitating the study of complex motion representation.
- We generate a new dataset that includes both novel scenes with complex motions and scenes converted from existing benchmarks. Several real world recorded testcases are included as well to fully evaluate the practicality in real world applications. Both the dataset with a total of ten scenes and the source files used for its creation are available open-source, allowing the community to further investigate severe motion understanding.

2 Related Work

2.1 Dynamic Neural Rendering

Neural Radiance Fields (NeRF) (Mildenhall et al. 2021) have demonstrated promising performance in 3D scene representation and high-quality novel view synthesis. However, traditional NeRF suffers from high training and inference times and struggles with dynamic scenes. One strategy is to extend the 3D scene with a temporal dimension to create a 4D scene (Gao et al. 2021; Li et al. 2022b), but this straightforward extension increases computational costs. Another direction (Guo et al. 2023; Li et al. 2021; Park et al. 2021; Pumarola et al. 2021; Tretschk et al. 2021; Xian et al. 2021) maintains a static canonical space with a deformation field to represent motion. This split structure reduces the difficulty constraining dynamic scenes and results in improved accuracy. Other methods improve overall quality by incorporating additional modalities. Some researches (Song et al. 2023; Tretschk et al. 2021) use object segmentation to separate static background from moving objects. TöRF (Attal et al. 2021) employs depth maps to provide extra constraints. Additionally, 2D CNNs (Lin et al. 2022; Peng et al. 2023) are utilized to enrich spatial understanding.

Some efforts aim at reducing the computational expenses in dynamic NeRF. Neural 3D (Li et al. 2022b) utilizes keyframes to compress visual features, capitalizing on redundant information in videos. Combining implicit neural rendering with explicit 3D structures significantly reduces the time complexity. For instance, Zip-NeRF (Barron et al. 2023) utilize voxel grids to represent the 3D scene, while K-Planes (Fridovich-Keil et al. 2023) use planes to provide additional spatial constraints. Despite these advancements, current NeRF-based methods still struggle with either accuracy or high computational demands in dynamic scene.

2.2 Dynamic 3D Gaussian Splatting

3D Gaussian Splatting (3D-GS) represents a 3D scene with a set of 3D Gaussian points and uses a differentiable neural rendering procedure for image rendering. By representing the scene with sparse points, 3D-GS significantly reduces computation time and memory while achieving comparable accuracy to NeRF while providing the benefits of an explicit representation. Extending 3D-GS to dynamic scenes follows a pathway similar to that of NeRF. 4DGS (Yang et al. 2023) extends the 3D Gaussian distribution to 4D Gaussian by incorporating a temporal dimension. Some methods (Wu et al. 2024; Yang et al. 2024a; Huang et al. 2024; Sun et al. 2024) use separate sub-models to represent the static canonical space and the deformation field. Although this approach works well for scenes with slight motion and deformation, it struggles with large motions. It may be ambiguous whether the re-projection error is coming from the deformation or the canonical space structure change. Spacetime Gaussian (Li et al. 2024) represent complex and long motions by multiple shorter segments with simpler motion. However, this leads to multiple independent 3DGS, which increases memory usage. It also does not address large motions during object initialization. To address this, we propose our M5D-GS, which fully decouples the static 3D structure, object level motion, and local deformation. Our method significantly improves accuracy in large motion scenarios while maintaining comparable time efficiency.

3 Method

3.1 Preliminaries

3D Gaussian Splatting (3D-GS) is an explicit 3D representation. The 3D scene is represented as a set of 3D Gaussians $\mathcal{G}$. Each 3D Gaussian $\mathcal{G}_i$ contains a mean vector $\mathcal{X}_i \in \mathbb{R}^3$ to represent the center point, and a covariance matrix $\Sigma_i \in \mathbb{R}^{3\times3}$. The distribution of $\mathcal{G}_i$ can be described as $\mathcal{G}_i = e^{-\frac{1}{2}\mathcal{X}_i^T\Sigma_i^{-1}\mathcal{X}_i}$.

In order to optimize the covariance matrix in a differentiable back-propagation pass, the matrix is decomposed into a rotation matrix R and a scaling matrix S as $\Sigma = RSS^TR^T$. The 3D covariance matrix and the center point can be projected into a 2D camera plane by using the camera extrinsic matrix P and the Jacobian matrix of the affine approximation of the projective transformation J as:

$$\mathcal{X}^{2D} = JP\mathcal{X}, \Sigma^{2D} = JP\Sigma P^TJ^T \tag{1}$$

The rotation matrix R is represented by a quaternion $q \in \mathbf{SO}(3)$, and the scale matrix S is expressed as a vector $s \in \mathbb{R}^3$. The projected 2D Gaussian $\mathcal{G}_i^{2D}(\mathcal{X}^{2D}, \Sigma^{2D})$, the opacity α_i, and the view-dependent color defined by spherical harmonic (SH) coefficients $sh_i \in \mathbb{R}^k$ (k is a hyperparameter for the number of SH functions) determine how a Gaussian point affects the color of a pixel $\mathcal{C}$ by:

$$c_i = \mathcal{G}_i^{2D}sh_i\alpha_i, \mathcal{C} = \sum_{i\in N} c_i\alpha_i\prod_{j=1}^{i-1}(1-\alpha_j) \tag{2}$$

N is the set of the ordered 3D Gaussians overlapping the pixel C projected into the camera. We now have the full attributes of a 3D Gaussian as $\mathcal{G}_i : (\mathcal{X}_i, q_i, s_i, sh_i, \alpha_i)$ which will be optimized during the training and stored for inference.

3.2 M5D-GS Framework

To handle large motions in the dynamic scene representation task, we introduce M5D-GS, which decouples the object level motion and per-Gaussian deformation δ_t. D3D-GS (Yang et al. 2024a) is selected as our baseline, since it achieves both good accuracy and robustness in slight motion scenes. The overall framework is shown in Figure 2. The inputs of the framework contain a timestamp t and the camera pose P_t. The output is the rendered 2D image. The whole framework contains three trainable parts (emphasized with yellow background in Figure 2): a set of 3D Gaussians as canonical space $\mathcal{G}$, an MLP for object level motion prediction M_t, and an MLP for per-Gaussian deformation prediction δ_t. The canonical 3D Gaussians are first transformed to the destined pose at time t by the predicted object level M_t to handle the large motion. Then, each Gaussian is fed into another MLP, along with t, to predict the local deformation. The canonical space is dynamically controlled by various thresholds, including gradient, Gaussian scale, and opacity, to be either pruned or densified.

Motion & Deformation Representation. The entire motion of the scene is decoupled into an object level motion for large translation and rotation, and a per-Gaussian deformation to represent local deformation. The object level motion is predicted by an MLP, Θ_{motion} (shown in red in Figure 2). The input to this block is the timestamp t, and the output is the overall motion M_t, which includes a rotation quaternion and a translation vector. The 3D Gaussians in the canonical space are first transformed by M_t to align with the 3D scene at time t in a rigid manner. The per-Gaussian deformation is predicted by another MLP, Θ_{deform} (shown in green in Figure 2). The output, δ_t, contains a set of transformations for each Gaussian, including rotation, translation, and scaling. This deformation allows the 3D Gaussian point cloud to capture local deformations that cannot be addressed by the object level motion alone. Since the global motion has already been excluded by the object level motion M_t, δ_t are typically very small (close to 0). To avoid vanishing gradients, δ_t is added as an offset to the 3D Gaussians rather than being multiplied. The final 3D Gaussians $\mathcal{G}_t$ at time t can be calculated as:

$$\begin{aligned} M_t = \Theta_{motion}(t), \delta t = \Theta_{deform}(\mathcal{G}, t) \\ \mathcal{G}_t = M_t\mathcal{G} + \delta_t \end{aligned} \tag{3}$$

Coarse-to-Fine Matching. Training monocular dynamic 3D scene representation often lacks constraints because there is only one frame available at each time. Different motions can result in the same observation. For instance, moving toward the camera and enlarging the scale can produce similar images. We assume that the object level motion involves only translation and rotation, without signifi-

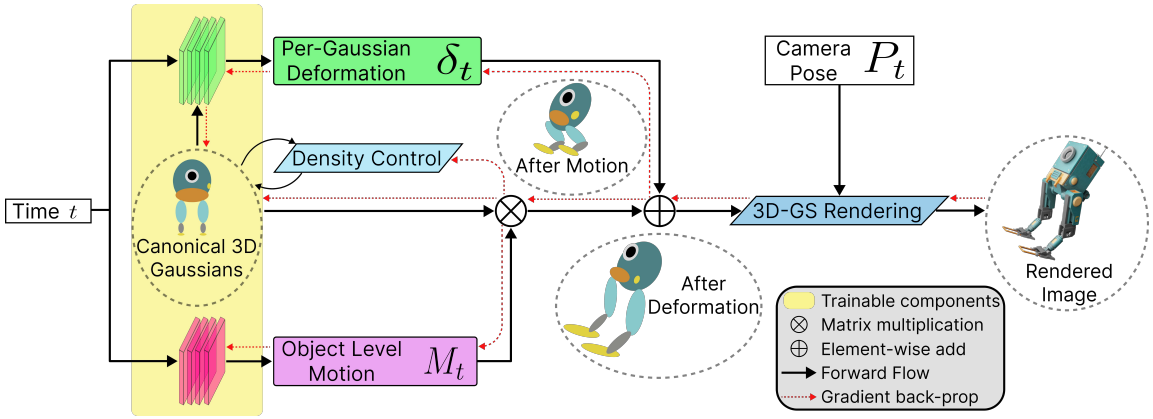

Figure 2: The overall workflow of M5D-GS. The inputs are the timestamp t, and the camera pose P_t. The output is the rendered image at the time t viewed from P_t. Three trainable components are included (highlighted with a yellow background): an MLP for predicting object level motion (highlighted in pink), an MLP for predicting local deformation (highlighted in green), and a set of 3D Gaussians representing the canonical space. The 3D Gaussian points are first transformed by the overall motion M_t, then deformed by the local deformation δ_t, and finally projected into the camera frame.

cant scale changes, while minor scaling can be handled by per-Gaussian deformation.

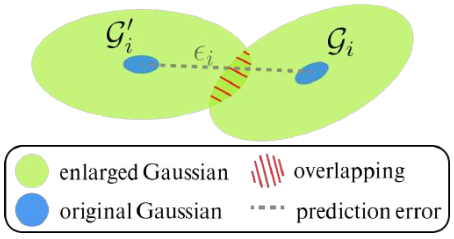

Figure 3: A simplified 2D example of the coarse-to-fine matching. Two blue ellipses represent the predicted distribution $\mathcal{G}'_i$ and the actual distribution $\mathcal{G}_i$ of a same Gaussian point at time t_i at real scale (fine matching). The gray dashed line represents the predicted translation error ϵ_i. The two green ellipses are the enlarged Gaussian points for coarse matching, with the red shadow represents the overlap.

The initialization of the canonical space is a core step in the framework. A low-quality canonical space will also negatively affect the subsequent per-Gaussian deformation. Thus, it is a common strategy to first warm up the training by only optimizing the canonical space for an initial scene. This strategy assumes that, even if the scene is dynamic, there is still some overlap between the predicted and actual 3D Gaussians. This assumption allows the multi-view geometry to estimate the 3D Gaussian points during the warm up stage. However, in scenes with large motion, this assumption is usually broken, resulting in the warm up initialization generating an empty scene. Figure 3 illustrates a simplified 2D example. $\mathcal{G}'_i$ and $\mathcal{G}_i$ represent the predicted and actual distributions of the same Gaussian point at time t_i. ϵ_i denotes the predicted translation error. The blue ellipses depict the distributions at real scale for fine matching, while the green ellipses represent the enlarged Gaussian points for coarse matching. To statistically analyze this task, the Bhattacharyya distance (D_B) (Bhattacharyya 1946) is used to measure the Overlap Coefficient (OC) between $\mathcal{G}'_i$ and $\mathcal{G}_i$ as $OC = exp(-D_B)$. The definition of D_B is as follows:

$$D_B(\mathcal{G}_i, \mathcal{G}'_i) = \frac{1}{8}(\mathcal{X}'_t - \mathcal{X}_t)^T \left(\frac{\Sigma'_t + \Sigma_t}{2}\right)^{-1} (\mathcal{X}'_t - \mathcal{X}_t) + \frac{1}{2} ln \left(\frac{det(\frac{\Sigma'_t + \Sigma_t}{2})}{\sqrt{det(\Sigma'_t) det(\Sigma_t)}}\right) \tag{4}$$

$0 \leq OC \leq 1$, and $OC = 1$ means fully overlapping. The correlations between OC and each variable are:

$$OC \propto D_B^{-1} \propto \epsilon_i^{-1} S_i \tag{5}$$

In Equation 5, S_i represents the scale of the Gaussian point, and $\propto$ indicates a positive correlation. A step-by-step proof is provided in the supplementary material. In large motion scenes, ϵ_i is typically very large at the beginning. This leads to a low OC, which causes the model to treat the Gaussian point as an outlier. To address this issue, we also train the object level motion block during the warm up stage. During this stage, each Gaussian point is uniformly scaled up in its local coordinates to increase OC. After the warm up stage, ϵ_i is reduced to an acceptable range, and the scale factor is gradually reduced to 1 by the time when the warm up stage is complete. Our proposed coarse-to-fine matching improves the initialization robustness and quality in large motion scenes.

Optimization Loss. The main constraints for the proposed M5D-GS still follow the original 3D-GS without additional loss for motion estimation. The overall constraints include a per-pixel $\mathcal{L}_1$ loss and a D-SSIM loss (Kerbl et al. 2023) $\mathcal{L}_{D-SSIM}$. The loss function is $\mathcal{L}_{img} = \mathcal{L}_1 + \lambda \mathcal{L}_{D-SSIM}$ with λ as the loss coefficient.

3.3 New Benchmarks with Complex Motion

Current benchmarks (Pumarola et al. 2021; Yan, Li, and Lee 2023; Li et al. 2022b) for dynamic scene representation usually contain only slight motion and deformation. Figure 1 qualitatively shows the range of motion for different scenes. Each scene is represented as a stack of 10 frames uniformly sampled from the temporal space with the camera pose fixed. The more overlap between different frames, the less motion exists in the scene. Figure 1 (a) shows the scenes from the original D-NeRF dataset, most of the frame contents overlap, especially in the latter two scenes. To fairly evaluate previous methods under large motion scenarios, we propose a pipeline to add large rigid motions to existing scenes while using the same dataset images. Assume I_t is the frame image at time t, J is the camera intrinsic matrix, $P_t^{w\to c}$ and $P_t^{c\to w}$ represent the world-to-camera and camera-to-world transformation matrices, and O_t represents the object pose in the world coordinates. The goal is to add a designed motion T_t to O_t, while keeping I_t the same. The object pose is an implicit variable that cannot be modified directly without regenerating the whole scene. Therefore, we design the following equations to add the object motion indirectly by modifying the camera pose:

$$\begin{aligned} I_t &= JP_t^{w\to c}O_t = JP_t^{w\to c}(T_t^{-1}T_t)O_t \\ &= J(P_t^{w\to c}T_t^{-1})(T_tO_t) = J(T_tP_t^{c\to w})^{-1}(T_tO_t) \end{aligned} \tag{6}$$

Equation 6 proves that transforming the camera extrinsic matrix by the inverse of the designed motion and using the same image is equivalent to transforming the object by the same motion matrix. This allows us to reuse previous dataset images by only manipulating the camera extrinsic matrix. Figure 1 (b) shows the scenes after adding complex motions. We also generate 5 new synthetic scenes involving complex motion and 2 novel real world recorded scenes. Several examples are shown in Figure 1 (c).

4 Experimental Evaluation

4.1 Experiment Setup

We select five state-of-the-art 3D representation methods as comparators. D3D-GS (Yang et al. 2024a), SC-GS (Huang et al. 2024), and 4D-GS (Wu et al. 2024) are three state-of-the-art Gaussian Splatting based methods. 4D-GS shares a similar structure with D3D-GS but is different in the description of the deformation field. SC-GS is developed on top of D3D-GS by adding sparse control points and additional local deformation constraints. Two state-of-the-art NeRF-based methods, K-Planes (Fridovich-Keil et al. 2023) and TiNeuVox (Fang et al. 2022) are also incorporated. NeRF-based methods represent the scene densely and are usually more stable but perform worse than the 3D-GS based methods.

The evaluation metrics follow the previous public benchmarks (Pumarola et al. 2021; Li et al. 2021). Specifically, Peak Signal-to-Noise Ratio (PSNR), Structural Similarity (SSIM), and VGG-based Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al. 2018) are used. More details are available in the supplementary material.

4.2 Dataset

Current public datasets (Pumarola et al. 2021; Li et al. 2021; Yan, Li, and Lee 2023) only contain scenes with slight motions. To fully evaluate performance in scenes with large motion, we propose a novel dataset that contains both synthetic and real world scenes with objects under complex motions. The proposed dataset contains ten different scenes, three of which are augmented from the previous dataset by adding large motions, five brand new scenes generated from scratch, and two real world recorded scenes.

Our novel dataset contains more frames with object motions and deformations, including both rigid and non-rigid deformations. We also release the source files of the object models used for the synthetic dataset generation, allowing for potential 3D-level evaluations in the future. Several examples of our dataset are shown in Figure 1 (b) and (c). More details of our dataset and the evaluations on previous slight motion dataset (Pumarola et al. 2021) are in the supplementary material.

4.3 Results and Comparisons

Comparison on Large Motion Dataset. Table 1 shows the results of the comparison between our method and competitors on the proposed large motion dataset. The best results are shown in bold, and the second best results are underlined. Two real world testcases are listed in the last column. In each sub-table, methods above the dashed line are NeRF-based, while the GS-based methods are below the dashed line. "Fails" in the table indicates that the method fails to reconstruct the 3D scene in the corresponding testcase or generates meaningless noise. The experiment shows that our method significantly improves the accuracy and robustness in large motion scenes. A visual comparison is shown in Figure 4, and more visualizations can be found in the supplementary material.

Visualizing the Predicted Motion. The ground truth of the motion is not accessible because object deformation may be indistinguishable from slight motions. For example, in Figure 5, the fish's body movement is reflected as minor fluctuations in the trajectory. To illustrate this, we provide an example in Figure 5 that qualitatively shows the estimated motion trajectory of the scene. The black line represents the estimated motion trajectory. Five frames are uniformly sampled from the entire timeline. The estimated positions are indicated by red dots and the orientations are represented by RGB coordinates. This visualization demonstrates that our method successfully estimates motions that meet our objective.

Comparison of Rendering Speed. Table 2 shows the rendering speed for all 3D-GS based competitors. NeRF-based methods are significantly slower than 3D-GS based methods, so only four 3D-GS based methods are compared. The average processing Frame Per Seconds (FPS) of four testcases where all the 3D-GS based methods succeeded are evaluated. The best result is bold, and the second best is underlined. The total rendering time depends not only on the method complexity but also on the total number of Gaussian points in the scene. D3D-GS achieves the highest FPS, and

Method	Jumping Jacks + Motion			Hell Warrior + Motion			Bouncing Balls + Motion			Elephant			Grey Cat		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
TiNeuVox	29.22	0.95	0.07	31.18	0.92	0.13	28.71	0.97	0.11	16.37	0.77	0.20	33.28	0.97	0.04
K-Planes	27.05	0.94	0.07	18.69	0.88	0.14	32.60	0.97	0.07	Fails			—		
4D-GS	Fails			12.01	0.84	0.17	16.65	0.88	0.27	15.05	0.87	0.17	17.59	0.89	0.10
SC-GS	Fails			Fails			9.42	0.53	0.37	18.26	0.82	0.15	Fails		
D3D-GS	Fails			32.16	0.93	0.10	30.37	0.97	0.05	29.00	0.96	0.07	Fails		
M5D-GS	**35.86**	**0.99**	**0.02**	**37.37**	**0.97**	**0.06**	**36.47**	**0.99**	**0.02**	**37.44**	**0.99**	**0.03**	**39.69**	**0.99**	**0.01**

Method	Robot			Pokemon			Fish			Robot Dog			Pillow		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
TiNeuVox	24.52	0.89	0.13	14.89	0.75	0.24	27.33	0.93	0.10	12.99	0.71	0.31	27.38	0.93	0.08
K-Planes	17.56	0.84	0.22	Fails			22.32	0.91	0.13	Fails			—		
4D-GS	22.45	0.91	0.11	17.15	0.89	0.20	16.02	0.90	0.13	16.65	0.88	0.17	17.69	0.88	0.13
SC-GS	Fails			21.44	0.83	0.15	Fails			15.08	0.79	0.20	Fails		
D3D-GS	24.32	0.92	0.11	29.84	0.95	0.07	Fails			19.25	0.90	0.12	29.89	0.96	0.06
M5D-GS	**29.78**	**0.96**	**0.05**	**35.60**	**0.98**	**0.03**	**36.15**	**0.98**	**0.02**	**33.46**	**0.98**	**0.02**	**32.37**	**0.98**	**0.04**

Table 1: Quantitative results on our large motion dataset. PSNR, SSIM, and LPIPS(VGG) are evaluated for each competitor. The best result is in **bold**, and the second best is underlined. The last column contains two real world testcases.

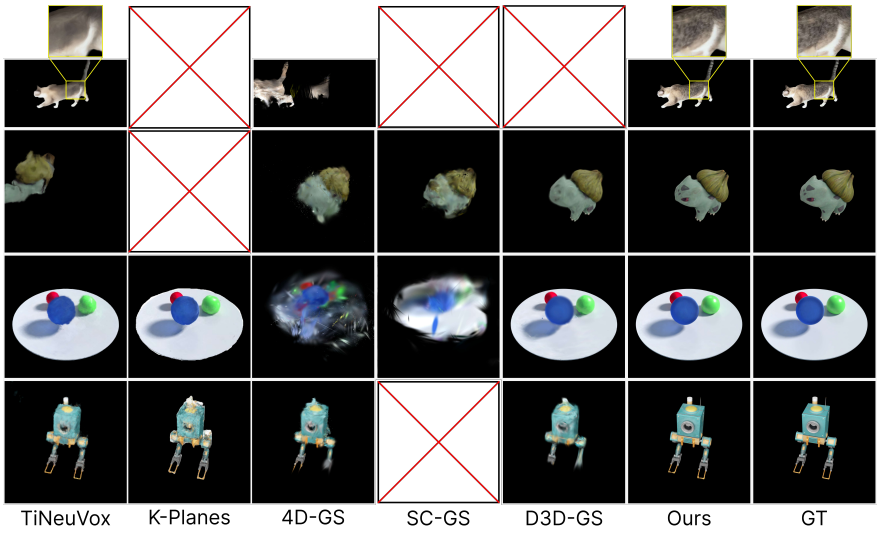

Figure 4: Qualitative examples from our large motion dataset for each competitor. The blank box with a red cross indicates that the method cannot handle a particular scene. Zoom in for more detailed views.

our M5D-GS achieves the second best. However, we can observe that the 3D scenes generated by D3D-GS are blurrier (shown in Figure 4) with lower PSNR accuracy.

4.4 Ablation Study

Motion & deformation prediction. M5D-GS represents the change of the entire scene with an overall motion model to predict the object level translation and rotation, and a local deformation model for local deformation prediction. It is not meaningful to evaluate the rendering accuracy for each part independently since both parts are working collaboratively. Therefore, we provide a set of qualitative results, as shown in Figure 6. The three rows represent three different types of deformations. In the first row, the elephant undergoes a non-rigid deformation; the robot in the middle row undergoes a rigid deformation; and, multiple objects with different motions are included in the last row. Object rotations and translations are involved in all three scenes as well. The first three columns show the rendered image from the 3D Gaussian points at different stages (corresponding to the three hand-drawn robot pictures in Figure 2), and the last column shows the ground truth image. In each row, the four images are generated from the same camera viewpoint and angle. The framework selects the optimal pose for the scene to reconstruct the canonical space during training, so the 3D Gaussians in the canonical space (Figure 6 first column) are under different orientations and locations. Once the object level motion is predicted, the 3D Gaussian points are transformed to match the desired object structure (Figure 6 second column). Finally, the time-variant local deformation is predicted and added to the transformed 3D Gaussian points to generate the final 3D Gaussian points for image rendering (Figure 6 third column). This visualization indicates that our framework effectively handles motion and deformation

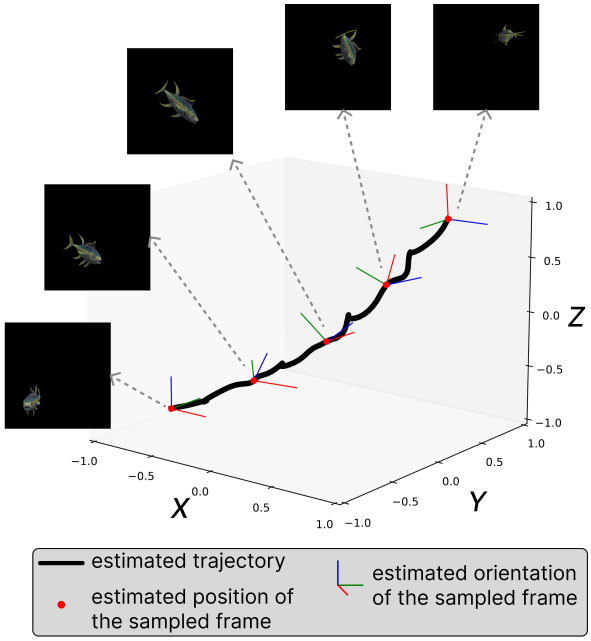

Figure 5: An example of the estimated object motion. The black line represents the estimated trajectory over the entire timeline. Five samples are highlighted where the red dot indicates the estimated position, and the RGB coordinate frame represents the estimated orientation.

in two separate steps as designed.

Influence of the Large Motion. Our proposed benchmark contains three scenes that are upgraded from the original D-NeRF dataset with large motion added. These three scenes can be used to evaluate how large motion affects the performance of different methods. The PSNR decrease is evaluated for different scenes. SC-GS is not compared since it either fails during training, or generates a low-accuracy result in three scenes (the first three scenes in Table 1). Table 3 shows how large motion affects different methods. The numbers shown in the table indicate the decrease of PSNR after adding large motion to the original scenes. The smaller the number, the less influence the method experiences by the large motion. The dash means the method fails when processing the large motion scenes. Our method significantly improves robustness under large motion scenarios with the smallest decrease. The full experiment on the original D-NeRF dataset can be found in the supplementary material.

Method	SC-GS	4D-GS	D3D-GS	Ours
FPS	37.36	93.90	**267.39**	<u>135.15</u>
PSNR	16.05	16.38	<u>27.12</u>	**35.74**

Table 2: Comparison of rendering frame per second (FPS) and rendering quality. The average value across three scenes where all 3D-GS based methods succeeded are compared. The best result is in **bold**, and the second best is <u>underlined</u>.

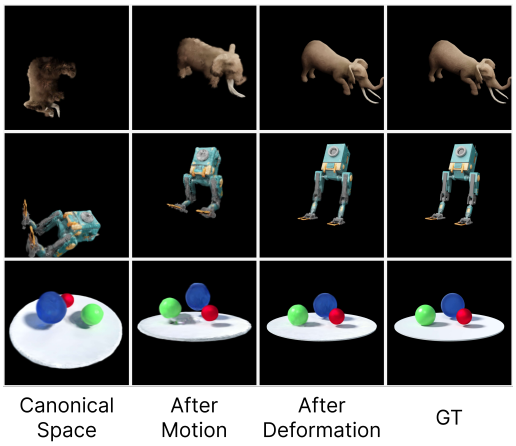

Figure 6: Visualization of the reconstructed scenes at three stages: the canonical space, after motion, and after deformation. These three steps correspond to the stages in Figure 2.

	J Jacks	H Warrior	B Balls	mean ↓
TiNeuVox	4.27	7.09	11.52	7.63
K-Planes	4.06	5.89	7.45	5.80
4D-GS	-	16.70	23.97	20.34*
D3D-GS	-	9.38	10.64	10.01*
Ours	**2.19**	**3.76**	**3.22**	**3.06**

Table 3: Influence of large motion. The average decrease after adding large motion is compared. "-" indicates the method fails, and "*" represents the value is average of only two successful scenes.

5 Conclusion

We propose M5D-GS, a framework that decouples motion and deformation prediction, for 3D model representation and acquisition in large motion scenarios. By processing the object level motion and per-Gaussian local deformation separately, our method not only improves the robustness but also outperforms all the competitors in large motion scenes. To fully evaluate the capacity of different methods under large motion, we introduce a novel large motion dataset combining three scenes from previous dataset with additional large motion, five newly generated synthetic challenging scenes, and two real world recorded scenes. Our method significantly improves performance in the large motion dataset without losing generalization in previous benchmarks with slight motion. However, some limitations remain as potentially future work directions. Our method only supports single motion prediction. It is a promising direction to combine semantic labels in the scene and predict separate motions for different parts independently,or separate dynamic objects and static background. Monocular camera based dynamic object representation is a lack of constraint problem. Exploring the spatial-temporal connection can reduce the overfitting. Also, the proposed method can potentially be extended into a 6-DoF object pose estimation method with slight modifications.

References

Attal, B.; Laidlaw, E.; Gokaslan, A.; Kim, C.; Richardt, C.; Tompkin, J.; and O'Toole, M. 2021. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. *Advances in neural information processing systems*, 34: 26289–26301.

Barron, J. T.; Mildenhall, B.; Verbin, D.; Srinivasan, P. P.; and Hedman, P. 2023. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 19697–19705.

Bhattacharyya, A. 1946. On a measure of divergence between two multinomial populations. *Sankhyā: the indian journal of statistics*, 401–406.

Cao, A.; and Johnson, J. 2023. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 130–141.

Chen, A.; Xu, Z.; Geiger, A.; Yu, J.; and Su, H. 2022. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, 333–350. Springer.

Debevec, P.; Yu, Y.; and Borshukov, G. 1998. Efficient view-dependent image-based rendering with projective texture-mapping. In *Rendering Techniques' 98: Proceedings of the Eurographics Workshop in Vienna, Austria, June 29—July 1, 1998 9*, 105–116. Springer.

Fang, J.; Yi, T.; Wang, X.; Xie, L.; Zhang, X.; Liu, W.; Nießner, M.; and Tian, Q. 2022. Fast dynamic radiance fields with time-aware neural voxels. In *SIGGRAPH Asia 2022 Conference Papers*, 1–9.

Fridovich-Keil, S.; Meanti, G.; Warburg, F. R.; Recht, B.; and Kanazawa, A. 2023. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12479–12488.

Gao, C.; Saraf, A.; Kopf, J.; and Huang, J.-B. 2021. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5712–5721.

Gao, J.; Chen, W.; Xiang, T.; Jacobson, A.; McGuire, M.; and Fidler, S. 2020. Learning deformable tetrahedral meshes for 3d reconstruction. *Advances in neural information processing systems*, 33: 9936–9947.

Guo, X.; Sun, J.; Dai, Y.; Chen, G.; Ye, X.; Tan, X.; Ding, E.; Zhang, Y.; and Wang, J. 2023. Forward flow for novel view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16022–16033.

Huang, Y.-H.; Sun, Y.-T.; Yang, Z.; Lyu, X.; Cao, Y.-P.; and Qi, X. 2024. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4220–4230.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.*, 42(4): 139–1.

Li, L.; Shen, Z.; Wang, Z.; Shen, L.; and Tan, P. 2022a. Streaming radiance fields for 3d video synthesis. *Advances in Neural Information Processing Systems*, 35: 13485–13498.

Li, T.; Slavcheva, M.; Zollhoefer, M.; Green, S.; Lassner, C.; Kim, C.; Schmidt, T.; Lovegrove, S.; Goesele, M.; Newcombe, R.; et al. 2022b. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5521–5531.

Li, Z.; Chen, Z.; Li, Z.; and Xu, Y. 2024. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8508–8520.

Li, Z.; Niklaus, S.; Snavely, N.; and Wang, O. 2021. 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*, 6498–6508.

Li, Z.; Wang, Q.; Cole, F.; Tucker, R.; and Snavely, N. 2023. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4273–4284.

Lin, C.-H.; Kong, C.; and Lucey, S. 2018. Learning efficient point cloud generation for dense 3d object reconstruction. In *proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 1.

Lin, H.; Peng, S.; Xu, Z.; Yan, Y.; Shuai, Q.; Bao, H.; and Zhou, X. 2022. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia 2022 Conference Papers*, 1–9.

Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; and Geiger, A. 2019. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4460–4470.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.

Nan, L.; and Wonka, P. 2017. Polyfit: Polygonal surface reconstruction from point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, 2353–2361.

Park, K.; Sinha, U.; Barron, J. T.; Bouaziz, S.; Goldman, D. B.; Seitz, S. M.; and Martin-Brualla, R. 2021. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5865–5874.

Peng, S.; Yan, Y.; Shuai, Q.; Bao, H.; and Zhou, X. 2023. Representing volumetric videos as dynamic mlp maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4252–4262.

Pumarola, A.; Corona, E.; Pons-Moll, G.; and Moreno-Noguer, F. 2021. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10318–10327.

Schonberger, J. L.; and Frahm, J.-M. 2016. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4104–4113.

Shao, R.; Zheng, Z.; Tu, H.; Liu, B.; Zhang, H.; and Liu, Y. 2023. 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*, 16632–16642.

Song, L.; Chen, A.; Li, Z.; Chen, Z.; Chen, L.; Yuan, J.; Xu, Y.; and Geiger, A. 2023. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5): 2732–2742.

Sun, J.; Jiao, H.; Li, G.; Zhang, Z.; Zhao, L.; and Xing, W. 2024. 3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20675–20685.

Tretschk, E.; Tewari, A.; Golyanik, V.; Zollhöfer, M.; Lassner, C.; and Theobalt, C. 2021. 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*, 12959–12970.

Wang, L.; Zhang, J.; Liu, X.; Zhao, F.; Zhang, Y.; Zhang, Y.; Wu, M.; Yu, J.; and Xu, L. 2022. Fourier plenoctrees for dynamic radiance field rendering in real-time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13524–13534.

Wang, P.; Liu, Y.; Chen, Z.; Liu, L.; Liu, Z.; Komura, T.; Theobalt, C.; and Wang, W. 2023. F2-nerf: Fast neural radiance field training with free camera trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4150–4159.

Wu, G.; Yi, T.; Fang, J.; Xie, L.; Zhang, X.; Wei, W.; Liu, W.; Tian, Q.; and Wang, X. 2024. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20310–20320.

Xian, W.; Huang, J.-B.; Kopf, J.; and Kim, C. 2021. Spacetime neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9421–9431.

Yan, Z.; Li, C.; and Lee, G. H. 2023. Nerf-ds: Neural radiance fields for dynamic specular objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8285–8295.

Yang, Z.; Gao, X.; Zhou, W.; Jiao, S.; Zhang, Y.; and Jin, X. 2024a. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20331–20341.

Yang, Z.; Yang, H.; Pan, Z.; and Zhang, L. 2024b. Real-time Photorealistic Dynamic Scene Representation and Rendering with 4D Gaussian Splatting. In *International Conference on Learning Representations (ICLR)*.

Yang, Z.; Yang, H.; Pan, Z.; Zhu, X.; and Zhang, L. 2023. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*.

Yu, A.; Li, R.; Tancik, M.; Li, H.; Ng, R.; and Kanazawa, A. 2021. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5752–5761.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 586–595.