

A Appendix

In this appendix, we offer additional details regarding our FastParticle and Panoptic datasets, which provide the necessary context for our experiments. We delve into our method for articulated objects segmentation, presenting full qualitative results that demonstrate its effectiveness across various scenarios. Additionally, we clarify our rationale for maintaining the same number of iterations in our comparisons and present a comparison under equal wall-clock time, showing that our method still outperforms Dynamic3DGS [Luiten et al. \(2023\)](#). We also include visualizations illustrating the multi-layer clustering structure we employ, as well as the manually annotated tracking labels used for evaluating 2D tracking results. Furthermore, we discuss our approach to learning the deformation, emphasizing the two-phase training strategy. Finally, we reflect on limitations, identifying potential areas for future improvement.

A.1 FastParticle and Panoptic Datasets

In this section, we introduce the FastParticle and Panoptic datasets used in our experiments in details. The real-world Panoptic dataset includes six scenes: Basketball, Boxes, Football, Juggle, Softball, and Tennis. Each frame in these scenes comes with segmentation provided by the original authors. Following [Luiten et al. \(2023\)](#), we distinguish between foreground and background in these scenes and utilize background loss and floor loss accordingly. Each scene in this dataset contains 150 frames captured by a total of 31 cameras, with 27 cameras used for training and 4 for testing.

The synthetic FastParticle dataset, which we have accelerated, contains six dynamic scenes: Robot, Spring, Wheel, Pendulums, Robot-Task, and Cloth. After acceleration, these scenes respectively have 35, 18, 38, 24, 35, and 35 frames. As illustrated in Fig. 9, we show the dynamic evolution of some scenes, highlighting the significant changes between frames. This dataset includes 40 cameras in total, from which we randomly select 4 as testing cameras and the remaining 36 as training cameras.

For all experiments, we provide the same static checkpoints to all baselines. For the 12 scenes across the two datasets, we train for 20,000 iterations to obtain the checkpoints. Due to the varying complexity of the static scenes, 3,000 iterations are sufficient for most FastParticle scenes.

A.2 Articulated Objects Segmentation

As mentioned in Sec. 5.1. The intuition behind the KMeans design is that, (1) Gaussians belong to the same part of the object should be close to each other at all time, and (2) the rotations of Gaussians within the same rigid part should be the same. The first one can be trivial, here we provide more explanations about the second point. As shown in Fig. 10, suppose we have a rigid body with its centroid denoted as C_0 . This rigid body can be considered as a combination of two smaller rigid bodies, with their centroids denoted as C_1 and C_2 , respectively. After rotation, C_1 and C_2 move to C'_1 and C'_2 . Taking C_0 as the origin of the coordinate system, the movement of the rigid body can only be a rotation R around C_0 , and the two smaller rigid bodies move accordingly. When considering the left smaller rigid body alone, its motion should consist of a translation of its centroid C_1 and a rotation R_1 around C_1 . We aim to prove that $R_1 = R$. Therefore, consider a point P on the left rigid body, which moves to point P' after the movement. From the perspective of C_0 , we have

$$\overrightarrow{C_0 P'} = R \overrightarrow{C_0 P}. \quad (9)$$

Also, from the perspective of C_1 , we can have

$$\begin{aligned} \overrightarrow{C_0 P'} &= R_1 \overrightarrow{C_1 P} + \overrightarrow{C_0 C'_1} \\ &= R_1 \overrightarrow{C_1 P} + R \overrightarrow{C_0 C_1}. \end{aligned} \quad (10)$$

Therefore, we have

$$R \overrightarrow{C_1 P} = R_1 \overrightarrow{C_1 P}. \quad (11)$$

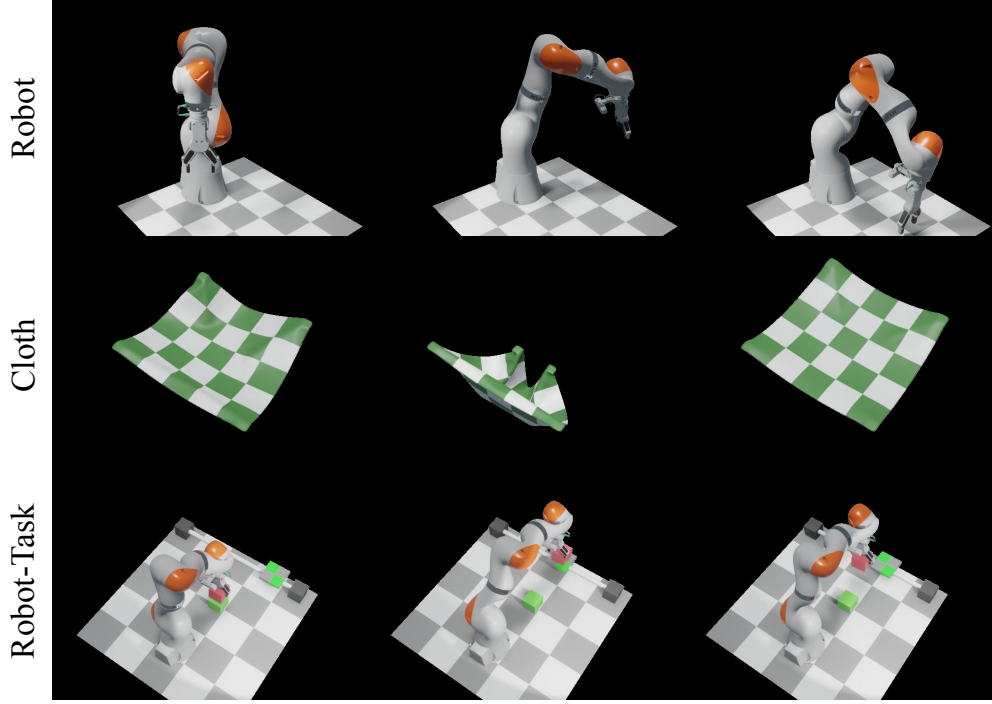


Figure 9: This figure shows the evolution of three scenes from the FastParticle dataset, demonstrating the high dynamic characteristics of the accelerated dataset.

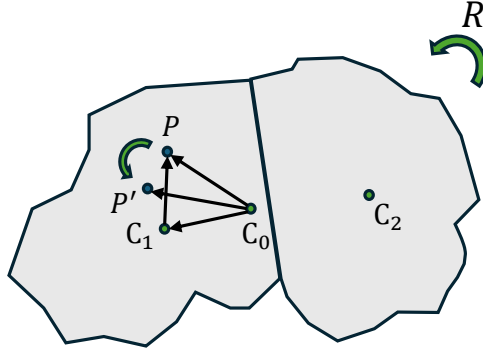


Figure 10: Illustration of a rigid body rotating R around its centroid. When considering the rigid body as composed of two smaller rigid bodies, it can be shown that the rotation of each smaller rigid body around its own centroid is the same with R .

Since the choice of P is arbitrary, we can conclude that $R_1 = R$. Similarly, we can prove that the rotation of the smaller rigid body on the right is also R . The above demonstrates the case where the rigid body is divided into two parts. This conclusion can be generalized to any case of multiple divisions, meaning that all parts of the same rigid body have the same rotation. Returning to our problem, since the rotation of Gaussians is around their centroids, the Gaussians belonging to the same rigid body should have the same rotation.

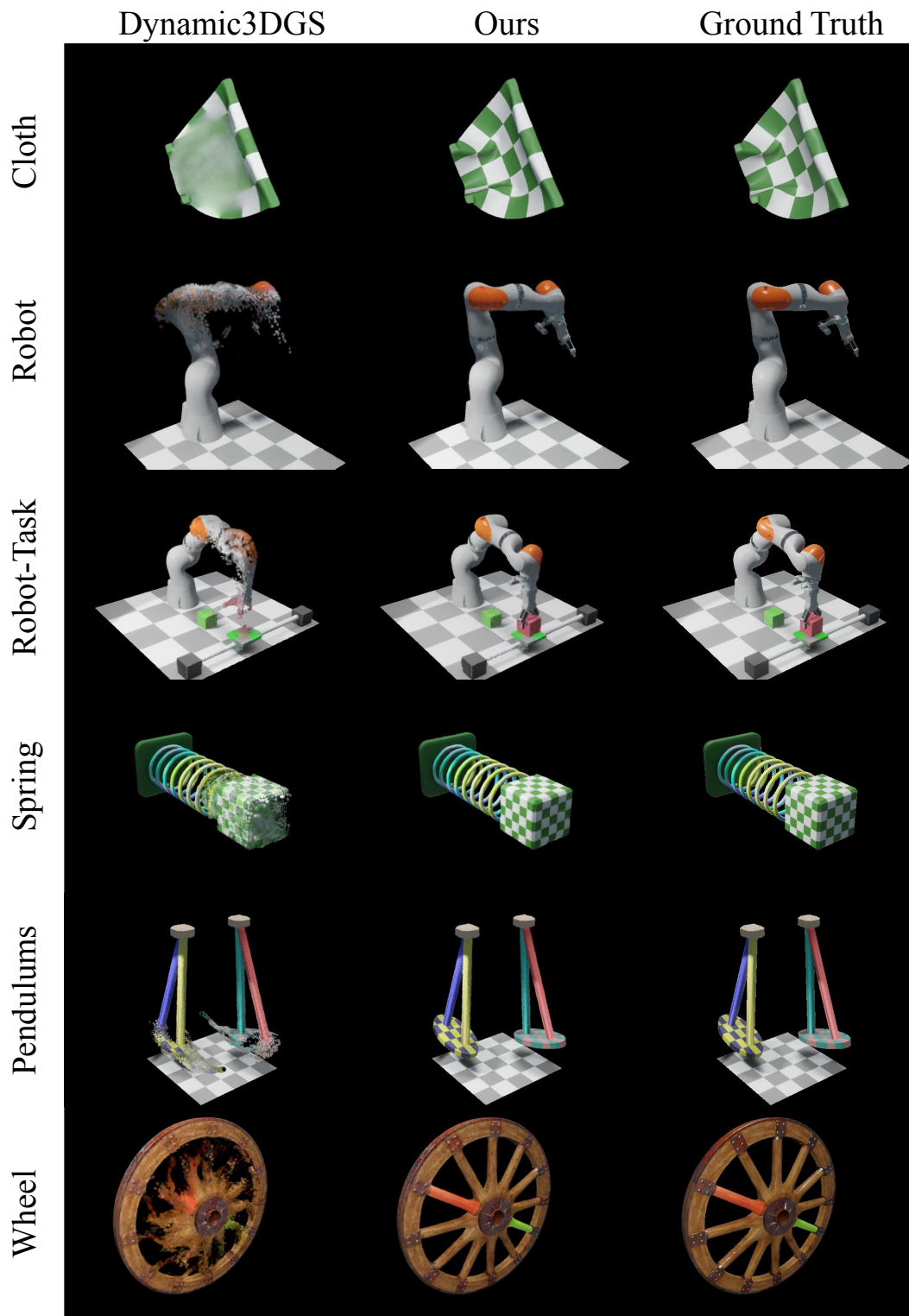


Figure 11: Qualitative results on FastParticle

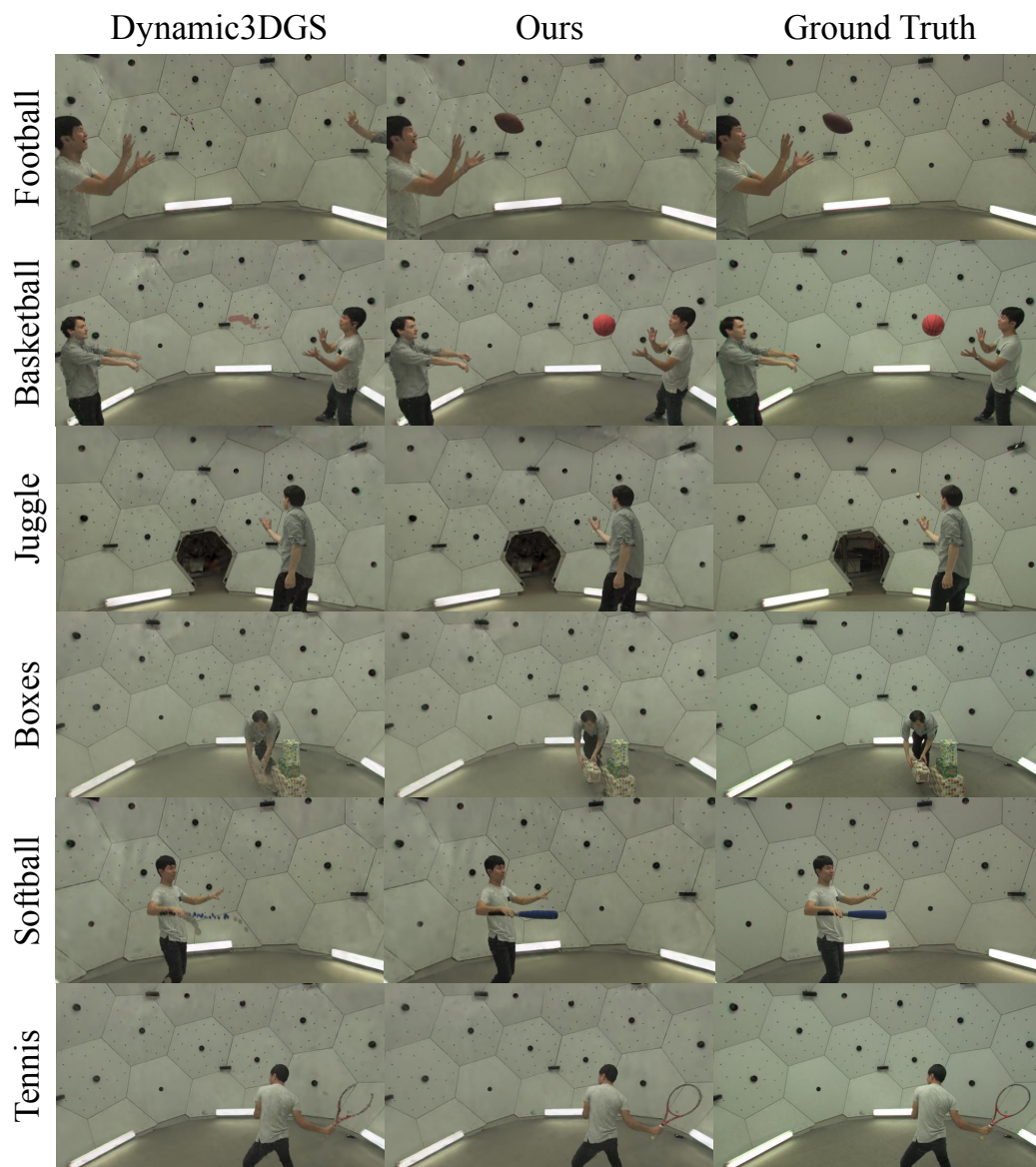


Figure 12: Qualitative results on Panoptic

Metrics	Method	FastParticle					
		Robot	Spring	Wheel	Pendulums	Robot-Task	Cloth
PSNR \uparrow	Ours ₁₀₀	29.46	30.28	27.95	30.60	27.67	31.68
	Dynamic3DGS ₃₀₀ Luiten et al. (2023)	27.66	27.16	26.67	29.57	26.79	30.41
SSIM \uparrow	Ours ₁₀₀	0.96	0.97	0.94	0.97	0.95	0.97
	Dynamic3DGS ₃₀₀ Luiten et al. (2023)	0.95	0.95	0.93	0.96	0.95	0.97
LPIPS \downarrow	Ours ₁₀₀	0.09	0.04	0.07	0.06	0.10	0.06
	Dynamic3DGS ₃₀₀ Luiten et al. (2023)	0.10	0.06	0.08	0.06	0.10	0.07

Table 5: Comparison of our method trained with 100 iterations per time frame against Dynamic3DGS.

Method	Particle					
	Robot	Spring	Wheel	Pendulums	Robot-Task	Cloth
K=2	29.48	30.40	27.86	30.54	27.35	31.51
K=4	29.39	30.09	27.87	30.38	27.55	31.48
K=5	29.17	30.00	27.79	30.36	27.60	31.43

Table 6: More ablation study results for the number of cluster layers. The reported metric is PSNR.

A.3 Full Qualitative Results

In this section, we provide qualitative results on all 12 scenes from the two datasets. As shown in Fig. 11 and Fig. 12, both our method and Luiten et al. (2023) are trained 100 iterations between two consecutive frames.

A.4 Same Wall-clock Time Comparisons

In our experiments, we use the same number of iterations across different methods for consistency. While wall-clock time may vary depending on the specific implementation (e.g., whether CUDA acceleration is employed), the number of iterations reflects the convergence speed of the algorithms. A lower number of iterations indicates faster convergence, showing that the optimization problem is easier to solve. This practice is commonly used in the evaluation of online methods, as demonstrated in the Dynamic3DGS Luiten et al. (2023) comparison (see Table 1 in their paper), where different methods are also compared using the same number of iterations.

Even when comparing with equivalent wall time, our method remains superior. To further illustrate this, we provide a comparison of our method trained for 100 iterations per frame versus Dynamic3DGS Luiten et al. (2023) trained for 300 iterations per frame on the FastParticle dataset. The results show that our method has an average training speed per iteration approximately twice as fast as Dynamic3DGS Luiten et al. (2023). As seen in Table 5, despite the difference in iteration count, our method still outperforms Dynamic3DGS Luiten et al. (2023) in terms of both efficiency and final performance.

A.5 Ablation Study on Number of Cluster Layers

We test layer numbers K from 1 to 5 on the FastParticle dataset. As shown in Table 6, $K = 3$ performs best. We find that increasing the number of cluster layers from 3 does not bring additional performance gains. On the contrary, it introduces more parameters to optimize, which may increase the required number of iterations and lead to diminishing returns. Therefore, we conclude that $K = 3$ offers the best trade-off between efficiency and performance in our framework.

A.6 Illustration of the Multi-Layer Structure

In Fig. 13, we show the coarse-to-fine multi-layer clustering structures for two objects in the FastParticle dataset. Different colors in the figure represent different clusters, and for clarification, the same color in different layers does not indicate any correlation between the clusters.

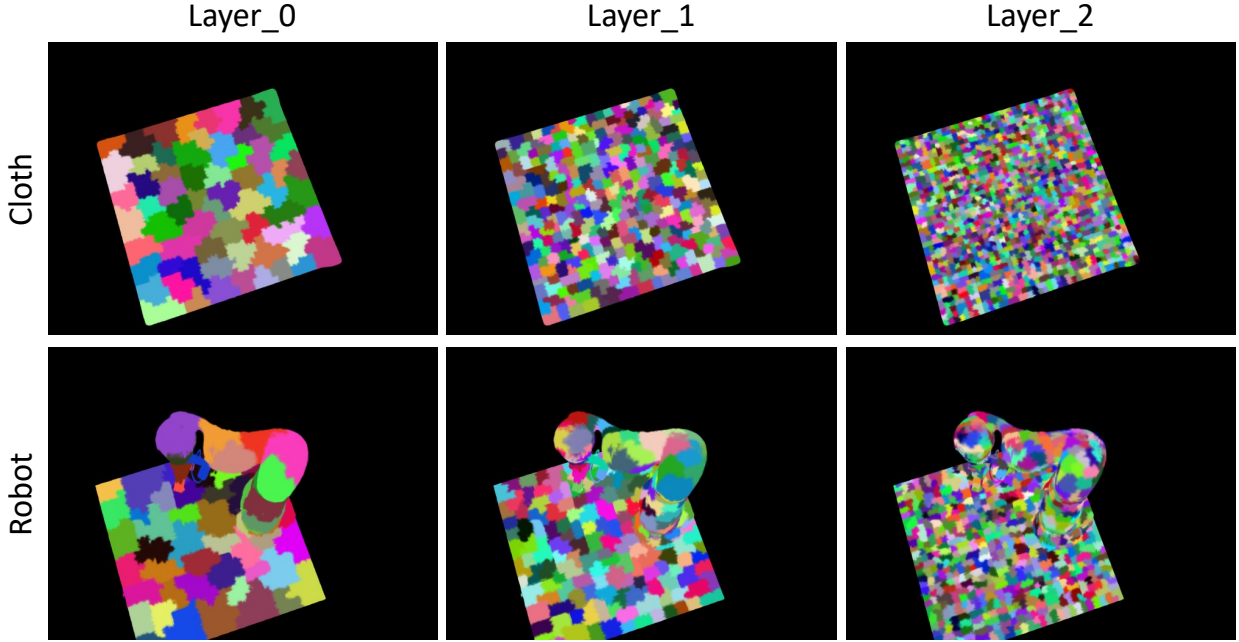


Figure 13: Coarse-to-fine multi-layer clustering structures for two objects in the FastParticle dataset.

A.7 Tracking Labels

Here, as shown in Fig. 14, we present all manually annotated 2D tracking ground truths. Since the human eye can only track points with distinct features across multiple frames, we only selected such points for annotation.

A.8 Learning the Deformation

Algorithm 1 summarizes our training process. Initially, we train our Gaussians on the static scene using observations from the first frame. Subsequently, we perform multilevel coarse-to-fine clustering for the centroids of the Gaussians. For each input in every time frame, we use an optimization approach to backpropagate loss and subsequently update our deformation functions.

Algorithm 1: Deformation-based Dynamic Scene Reconstruction Algorithm

Input: Images from all frames

$\Theta_{\text{prev}} \leftarrow$ Initialization stage (Static Gaussian Splatting);

Do Clustering;

for t **in** time_frames **do**

 Initialize the Deformation D ;

for iter **in** max_iters **do**

$\Theta_{\text{curr}} \leftarrow D(\Theta_{\text{prev}})$;

 Images \leftarrow Render(Θ_{curr});

 loss \leftarrow Loss(gt_Images, Images);

 Backpropagate(loss);

end

end

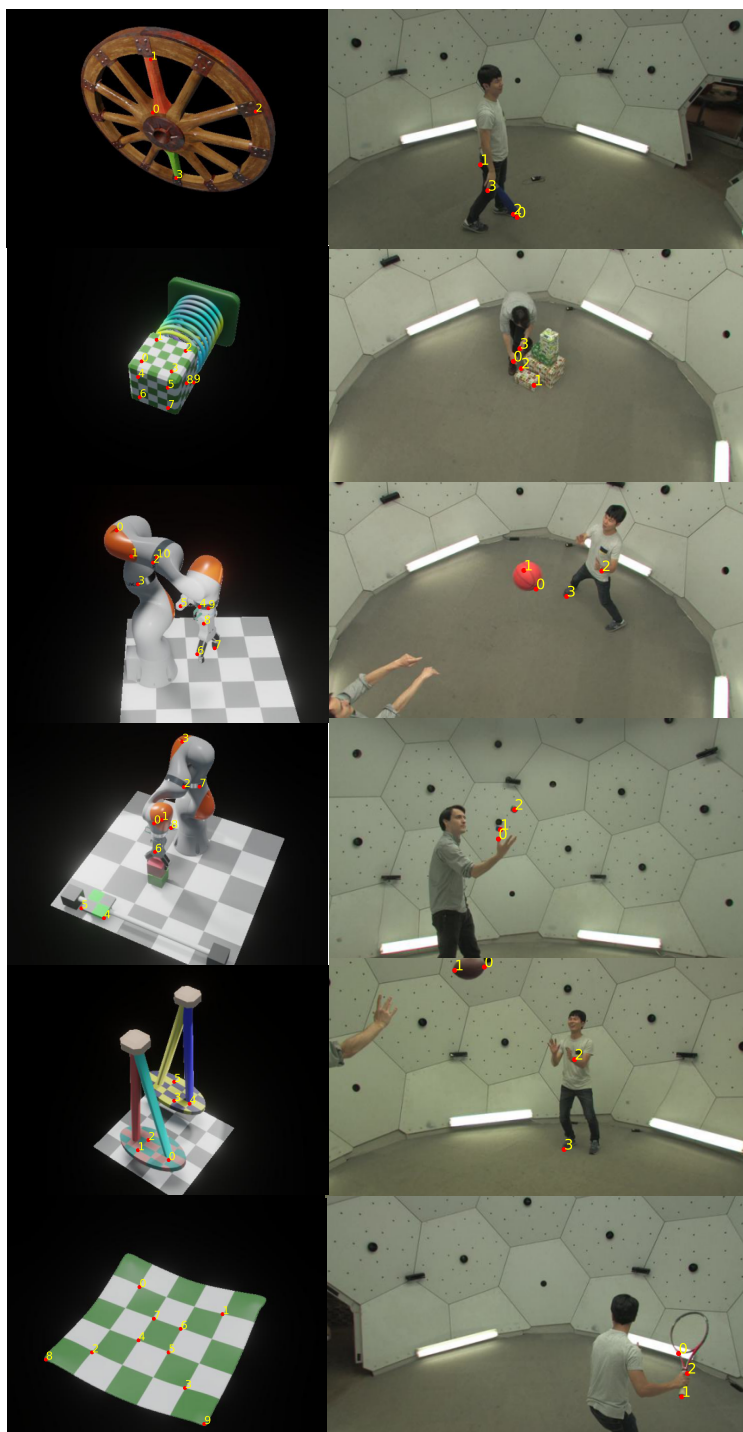


Figure 14: Illustration of our manually annotated tracking ground truths.

Method	Particle					
	Robot	Spring	Wheel	Pendulums	Robot-Task	Cloth
N=16	29.76	30.43	27.97	30.84	27.69	31.56
N=32	29.63	30.43	27.91	30.77	27.87	31.62
N=48	29.49	30.36	28.00	30.70	27.62	31.61

Table 7: PSNR results when varying the number of clusters in the first layer.

For potential negative impacts, since LayeredGS can learn deformation information and be used for creating new motions or inserting objects, such applications can be used for fake news to convince people by multi-view renderings. More censorship needs to be established in such cases.

A.9 Robustness to the number of clusters in the first layer

To further examine the robustness of our method to the initial clustering configuration, we conducted an additional experiment by varying the number of clusters in the first layer. The default number of clusters used in the main experiments is $N = 64$. Reducing the number of clusters leads to coarser groupings, potentially introducing more clustering errors at the coarse level. Despite this, our model achieves comparable performance across different settings, as summarized in Table 7. These results indicate that our structural cascaded optimization remains robust even when the initial clustering is coarser, thanks to the refinement provided by the multi-layer hierarchy and per-Gaussian updates at the finest level.

A.10 Effect of the Max Scale Constraint

To further examine the role of the max scale constraint in our method, we conducted an experiment by varying the value of `max_scale`. The scale loss is designed to prevent Gaussians from becoming excessively large during deformation, which may otherwise introduce rendering artifacts. We found that setting a reasonable threshold (in our main experiments, `max_scale` = 0.02) is sufficient to maintain rendering quality.

Our experiment shows that disabling this constraint (i.e., setting `max_scale` to a very large value) leads to significant PSNR degradation, as summarized in Table 8. These results demonstrate the importance of the max scale constraint in preserving stability during optimization.

Method	Particle					
	Robot	Spring	Wheel	Pendulums	Robot-Task	Cloth
<code>max_scale=0.02</code>	29.46	30.28	27.95	30.60	27.67	31.68
<code>max_scale=2.0</code>	15.86	19.86	22.42	18.86	18.15	21.53

Table 8: PSNR results under different values of the max scale constraint.

A.11 Additional Experiment on Per-Gaussian Deformation for Segmentation

We provide an additional experiment to examine the difference between our method and Dynamic3DGS [Luiten et al. (2023)] for segmentation purposes. Specifically, we trained Dynamic3DGS [Luiten et al. (2023)] for 2000 iterations and extracted per-Gaussian deformation (without any layered structure) to perform segmentation, similar to our approach. While this method produces visually reasonable motion, we found that the per-Gaussian rotations are significantly noisier, leading to worse segmentation quality.

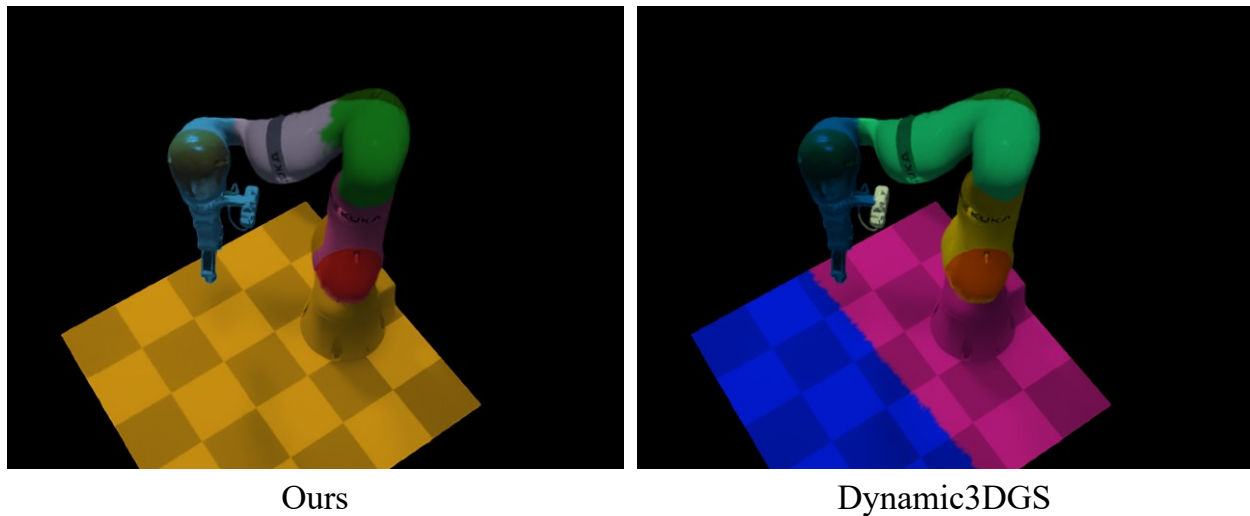


Figure 15: Visual comparison of segmentation results using per-Gaussian deformation (Dynamic3DGS [Luiten et al. \(2023\)](#)) and our multi-layer deformation structure.

In contrast, our multi-layer structure encourages Gaussians to move coherently with their parent clusters, resulting in more stable and interpretable rotations. This structural regularization improves segmentation accuracy by reducing noise in the estimated motion. Please refer to Figure [15](#) in the updated appendix for visual comparisons.

A.12 Limitations

While our method significantly reduces training iterations to 100 per frame, achieving real-time training and rendering remains a challenge. Additionally, the learned deformation information is not fully utilized, and the presented articulated object segmentation results are not well refined. Future work will focus on addressing these limitations by exploring real-time training approaches, refining deformation utilization techniques, and developing more sophisticated segmentation methods.