SpeeDe3DGS:

Speedy Deformable 3D Gaussian Splatting with Temporal Pruning and Motion Grouping

CVPR 2026

Allen Tu* Haiyang Ying* Alex Hanson Yonghan Lee

Tom Goldstein Matthias Zwicker

speede3dgs.github.io

Dynamic Gaussian Splatting extends 3DGS with a deformation network that predicts time-varying offsets:

$$\mathcal{G} = \{\mathcal{G}_i = \{\mu_i, s_i, r_i, h_i, \sigma_i\}\}_{i=1}^{N}$$

$$(\mu + \Delta\mu_t,\ r + \Delta r_t,\ s + \Delta s_t) = \mathcal{D}(\mu, r, s, t)$$

Motion models fall in two categories:

- **Neural motion** is slow but produces high visual fidelity
- **Non-neural motion** is fast but produces lower visual fidelity

Speedy Deformable 3D Gaussian Splatting (SpeeDe3DGS) bridges this efficiency-fidelity gap through three modules.

1. **Temporal Sensitivity Pruning (TSP)** removes redundant Gaussians by aggregating their gradient sensitivity across both space and time.
2. **Temporal Sensitivity Sampling (TSS)**
3. **GroupFlow**

For each Gaussian $\mathcal{G}_i$, compute its
Temporal Sensitivity Pruning (TSP) score:

$$\tilde{U}_{\mathcal{G}_i} \approx \nabla^2_{g_i} L_2 \approx \sum_{\phi, t \in \mathcal{P}_{gt}} \left(\nabla_{g_i} I_{\mathcal{G}_t}(\phi)\right)^2$$

as the second-order sensitivity of the
L_2 reconstruction loss to its projected value g_i,
where $\mathcal{P}_{gt}$ denotes the set of training poses and timesteps
and $I_{\mathcal{G}_t}(\phi)$ is the rendered image at pose ϕ and timestep t.

$\tilde{U}_{\mathcal{G}_i}$ reflects each Gaussian's **cumulative contribution** to the dynamic scene reconstruction over time and space.

TSP	TSS	GF	PSNR ↑	SSIM ↑	LPIPS ↓	FPS ↑	Size (MB) ↓	# Gaussians ↓	Train Time (s) ↓
DeformableGS [53]			23.80	0.8503	0.1781	54.37 (1.00×)	33.21 (1.00×)	132.22K (1.00×)	1523.83 (1.00×)
✓			23.78	0.8507	0.1863	346.96 (6.38x)	4.52 (7.35×)	10.90K (12.13×)	741.66 (2.05×)

We periodically **prune low-contributing Gaussians** during training using **TSP**, reducing neural inference load.

1. Temporal Sensitivity Pruning (TSP)
2. **Temporal Sensitivity Sampling (TSS)** enhances pruning stability by perturbing timestamps during sensitivity estimation, suppressing floaters and improving temporal coherence.
3. GroupFlow

Floaters: Unstable Gaussians that appear stable under observed motion but drift at unseen timestamps.

TSP is calculated on training poses and timesteps.

Temporal Sensitivity Sampling (TSS) introduces temporal perturbation during sensitivity estimation by jittering the timestamp input to the deformation network:

$$(\mu + \Delta\mu, r + \Delta r, s + \Delta s) = \mathcal{D}(\mu, r, s, t + \mathcal{N}(0, 1)\beta\Delta t(1 - i/\tau)).$$

Temporal Sensitivity Sampling (TSS) introduces temporal perturbation during sensitivity estimation by jittering the timestamp input to the deformation network:

$$(\mu + \Delta\mu, r + \Delta r, s + \Delta s) = \mathcal{D}(\mu, r, s, t + \mathcal{N}(0, 1)\beta\Delta t(1 - i/\tau)).$$

The noise anneals during training, exposing unstable Gaussians early in training and ensuring precise reconstruction in later stages.

Pruning with TSP + TSS improves PSNR and SSIM on NeRF-DS

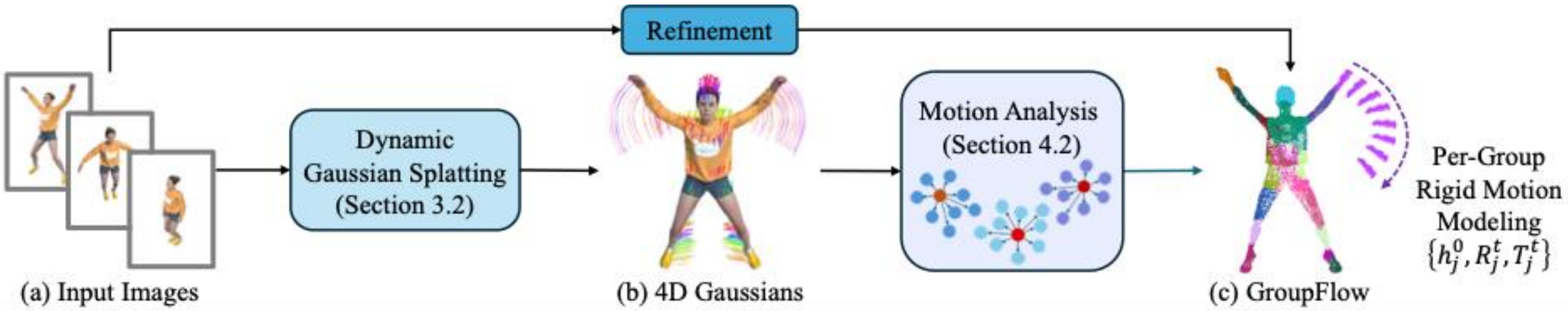
TSP	TSS	GF	PSNR ↑	SSIM ↑	LPIPS ↓	FPS ↑	Size (MB) ↓	# Gaussians ↓	Train Time (s) ↓
DeformableGS [53]			23.80	0.8503	0.1781	54.37 (1.00×)	33.21 (1.00×)	132.22K (1.00×)	1523.83 (1.00×)
✓			23.78	0.8507	0.1863	346.96 (6.38x)	4.52 (7.35×)	10.90K (12.13×)	741.66 (2.05×)
✓	✓		23.81	0.8515	0.1853	345.24 (6.35x)	4.55 (7.29×)	11.06K (11.95×)	750.69 (2.03x)

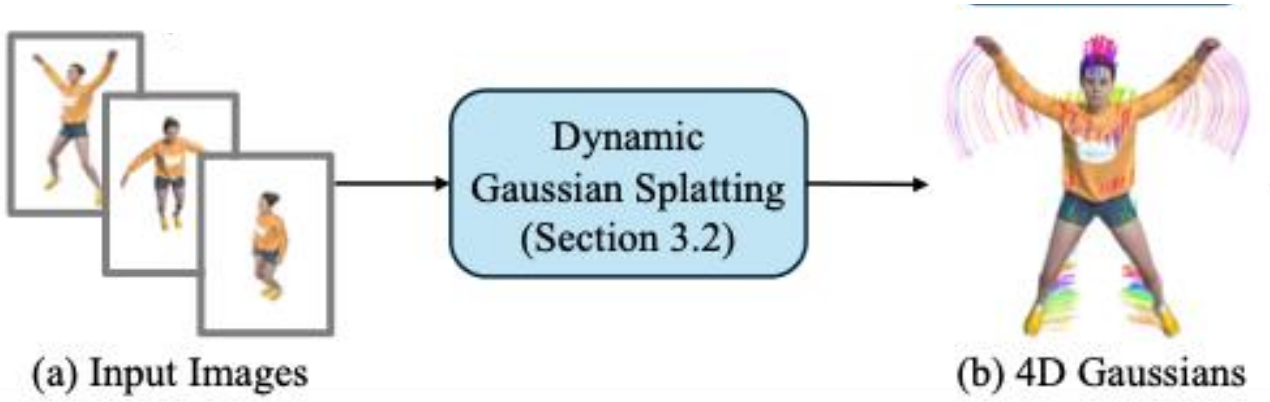
while **rendering 6.35× faster** and using **11.95× fewer Gaussians**.


DeformableGS
Baseline
33.07 FPS
186K Gaussians
0.8460 SSIM
1984sec Train
SpeeDe3DGS (Ours)
Pruning (TSP + TSS)
277.22 FPS
16K Gaussians
0.8498 SSIM
961sec Train

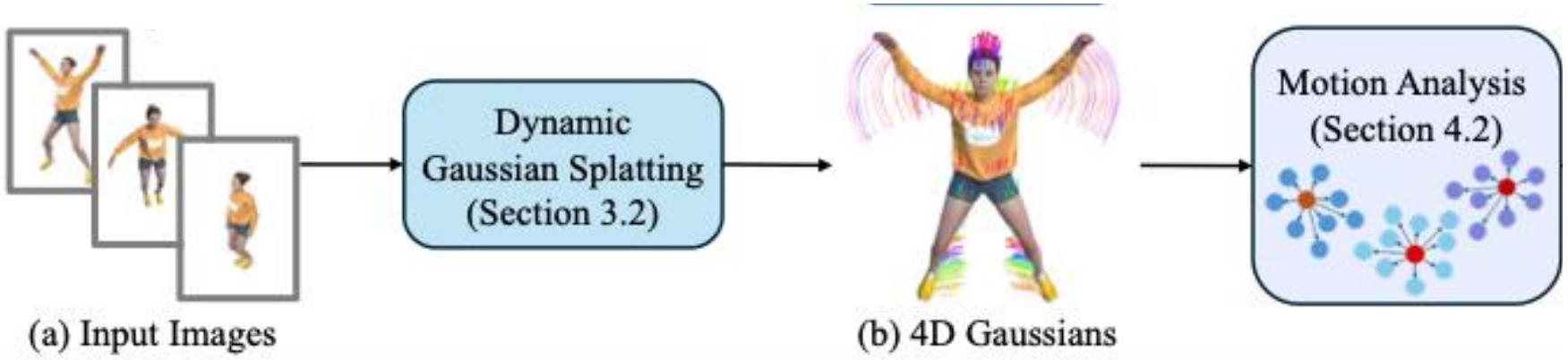

DeformableGS
Baseline
33.07 FPS
186K Gaussians
0.8460 SSIM
1984sec Train
SpeeDe3DGS (Ours)
Pruning (TSP + TSS)
277.22 FPS
16K Gaussians
0.8498 SSIM
961sec Train

1. **Temporal Sensitivity Pruning (TSP)**
2. **Temporal Sensitivity Sampling (TSS)**
3. **GroupFlow** clusters Gaussians with similar motion trajectories, distilling per-Gaussian neural deformations into shared groupwise SE(3) transformations.


Refinement
Dynamic
Gaussian Splatting
(Section 3.2)
Motion Analysis
(Section 4.2)
Per-Group
Rigid Motion
Modeling
$\{h_j^0, R_j^t, T_j^t\}$
(a) Input Images
(b) 4D Gaussians
(c) GroupFlow

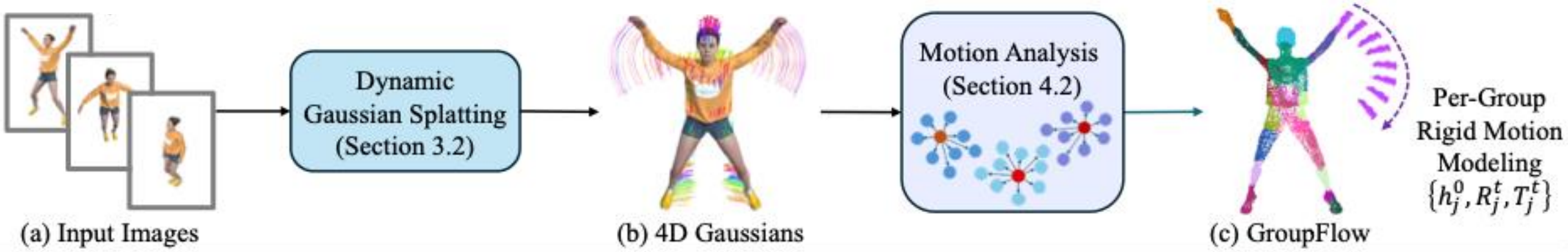


1. Start with a dense dynamic Gaussian Splatting model, where each Gaussian $\mathcal{G}_i$ is represented as a sequence of mean positions $\mathcal{M}_i = \{\mu_i^t\}_{t=0}^{F-1}$.



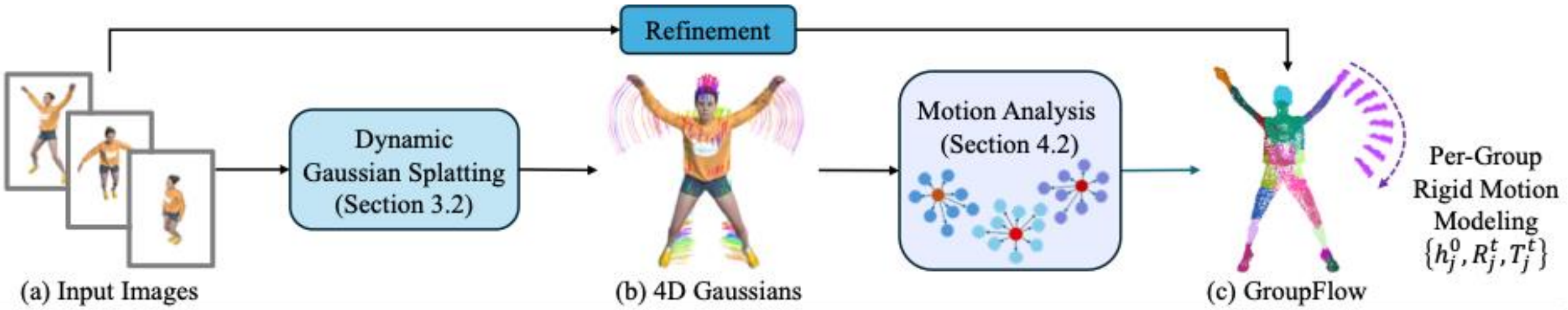
2. Initialize J control trajectories $\{h_j^t\}$ and assign each mean μ_i to the most similar control point h_j using trajectory similarity score:

$$S_{i,j} = \lambda_r \, \mathrm{std}_t(\|\mu_i^t - h_j^t\|) + (1 - \lambda_r) \, \mathrm{mean}_t(\|\mu_i^t - h_j^t\|).$$



2. We fit a group-wise SE(3) flow to each group $\mathcal{M}^j$ using Umeyama alignment. The rotation R_j^t and translation T_j^t applied to $\mu_i \in \mathcal{M}^j$ at timestep t are:

$$\mu_i^t = R_j^t(\mu_i^0 - h_j^0) + h_j^0 + T_j^t.$$



4. The shared flow $\{h_j^0, R_j^t, T_j^t\}$ is optimized jointly with the scene, **reducing the number of transforms per frame from N (per-Gaussian) to J (per-group).**

TSP	TSS	GF	PSNR ↑	SSIM ↑	LPIPS ↓	FPS ↑	Size (MB) ↓	# Gaussians ↓	Train Time (s) ↓
DeformableGS [53]			23.80	0.8503	0.1781	54.37 (1.00×)	33.21 (1.00×)	132.22K (1.00×)	1523.83 (1.00×)
✓			23.78	0.8507	0.1863	346.96 (6.38x)	4.52 (7.35×)	10.90K (12.13×)	741.66 (2.05×)
✓	✓		23.81	0.8515	0.1853	345.24 (6.35x)	4.55 (7.29×)	11.06K (11.95×)	750.69 (2.03x)
		✓	23.54	0.8433	0.1892	406.21 (8.58×)	51.00 (0.65×)	132.32K (1.00×)	826.75 (1.84×)

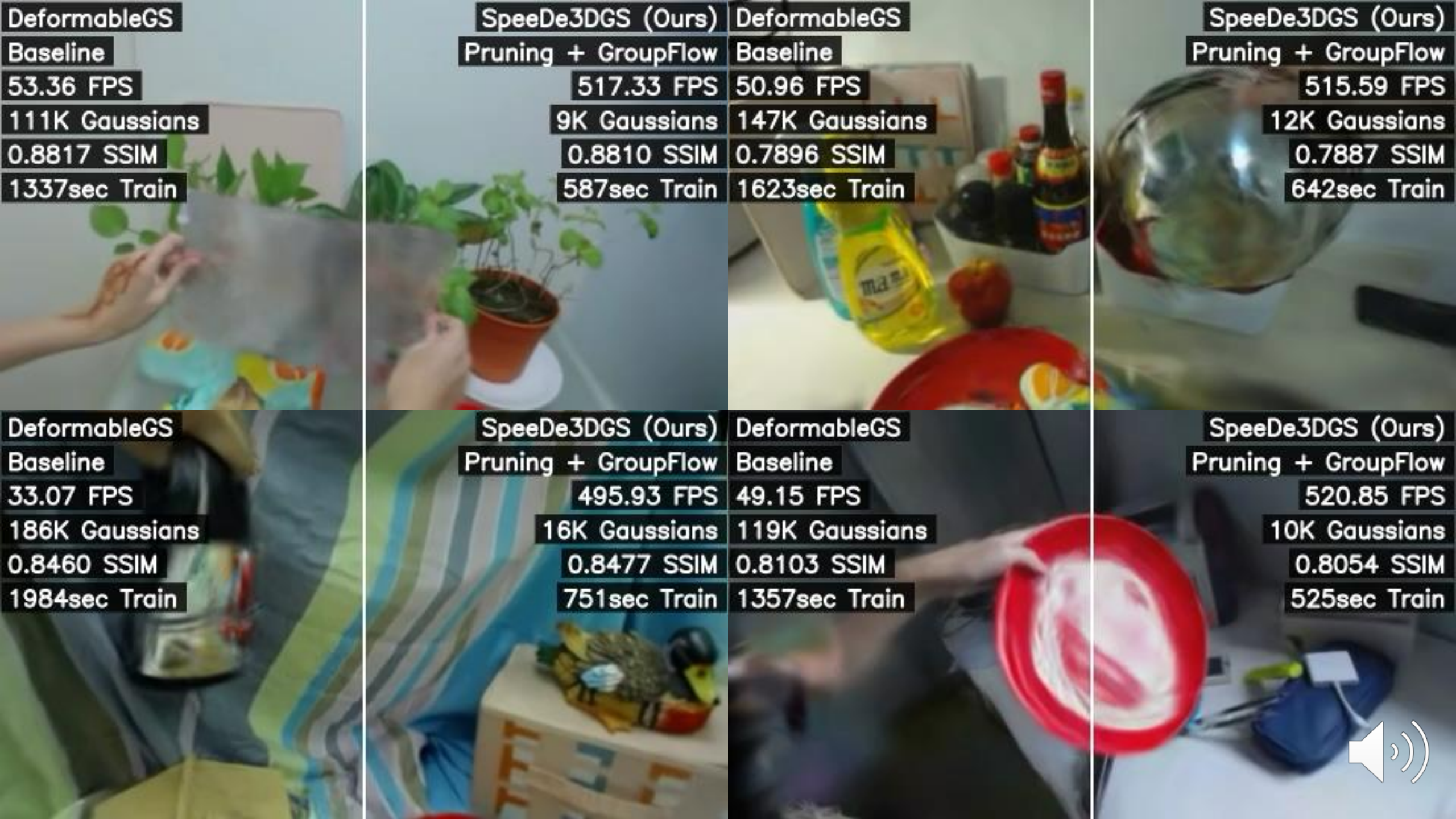
GroupFlow accelerates rendering speed by 8.58×
on the NeRF-DS dataset.

TSP	TSS	GF	PSNR ↑	SSIM ↑	LPIPS ↓	FPS ↑	Size (MB) ↓	# Gaussians ↓	Train Time (s) ↓
DeformableGS [53]			23.80	0.8503	0.1781	54.37 (1.00×)	33.21 (1.00×)	132.22K (1.00×)	1523.83 (1.00×)
✓			23.78	0.8507	0.1863	346.96 (6.38x)	4.52 (7.35×)	10.90K (12.13×)	741.66 (2.05×)
✓	✓		23.81	0.8515	0.1853	345.24 (6.35x)	4.55 (7.29×)	11.06K (11.95×)	750.69 (2.03x)
		✓	23.54	0.8433	0.1892	406.21 (8.58×)	51.00 (0.65×)	132.32K (1.00×)	826.75 (1.84×)
✓	✓	✓	23.66	0.8487	0.1901	505.60 (10.68×)	21.40 (1.55×)	11.10K (11.91×)	625.48 (2.44×)

When compared to DeformableGS, **SpeeDe3DGS** achieves

10.68× faster FPS and 59% faster training

while maintaining image quality on the NeRF-DS dataset.


DeformableGS
Baseline
53.36 FPS
111K Gaussians
0.8817 SSIM
1337sec Train
SpeeDe3DGS (Ours)
Pruning + GroupFlow
517.33 FPS
9K Gaussians
0.8810 SSIM
587sec Train
DeformableGS
Baseline
50.96 FPS
147K Gaussians
0.7896 SSIM
1623sec Train
SpeeDe3DGS (Ours)
Pruning + GroupFlow
515.59 FPS
12K Gaussians
0.7887 SSIM
642sec Train
DeformableGS
Baseline
33.07 FPS
186K Gaussians
0.8460 SSIM
1984sec Train
SpeeDe3DGS (Ours)
Pruning + GroupFlow
495.93 FPS
16K Gaussians
0.8477 SSIM
751sec Train
DeformableGS
Baseline
49.15 FPS
119K Gaussians
0.8103 SSIM
1357sec Train
SpeeDe3DGS (Ours)
Pruning + GroupFlow
520.85 FPS
10K Gaussians
0.8054 SSIM
525sec Train

TSP	TSS	GF	FPS ↑	Size (MB) ↓	# Gaussians ↓	Train Time (s) ↓
DeformableGS [53]			14.48 (1.00×)	159.30 (1.00×)	665.35K (1.00×)	5248.91 (1.00×)
✓			141.11 (9.75×)	14.98 (10.64×)	55.12K (12.07×)	1992.39 (2.63×)
✓	✓		135.70 (9.37×)	15.18 (10.49×)	56.00K (11.88×)	1968.72 (2.67×)
		✓	226.77 (15.66×)	179.79 (0.89×)	666.31K (1.00×)	2505.44 (2.10×)
✓	✓	✓	422.88 (29.21×)	31.62 (5.04×)	54.63K (12.18×)	1401.85 (3.74×)

When compared to DeformableGS, **SpeeDe3DGS** achieves

29.21× faster FPS and 73% faster training

while maintaining image quality on the HyperNeRF dataset.


DeformableGS
Baseline
13.24 FPS 557K Gaussians
4139sec Train 133.61MB
SpeeDe3DGS (Ours)
Pruning (TSP + TSS)
130.83 FPS 49K Gaussians
1441sec Train 13.60MB
SpeeDe3DGS (Ours)
Pruning + GroupFlow
438.86 FPS 48K Gaussians
977sec Train 28.88MB

Method	PSNR ↑	SSIM ↑	MS-SSIM ↑	LPIPS ↓	FPS ↑	Train Time (s) ↓
EffGS [20]	21.84	0.672	0.725	0.347	177.21	3757.81
STG-decoder [24]	21.81	0.678	0.742	0.352	109.42	5980.64
STG [24]	19.51	0.583	0.643	0.475	181.70	5359.56
RTGS [52]	21.61	0.663	0.720	0.350	143.37	7352.52
4DGS [47]	23.55	0.708	0.765	0.277	62.99 (1.00×)	8628.89 (1.00×)
+ Pruning (Ours)	22.44	0.689	0.737	0.334	**179.64 (2.85×)**	**4358.17* (1.47×)**
+ GroupFlow (Ours)	21.00	0.667	0.705	0.380	**290.21 (4.61×)**	**4176.49* (2.07×)**
DeformableGS [51]	24.07	0.694	0.755	0.283	20.20 (1.00×)	6227.43 (1.00×)
+ Pruning (Ours)	23.86	0.694	0.749	0.295	**137.01 (6.78×)**	**2850.60* (2.18×)**
+ GroupFlow (Ours)	23.52	**0.709**	**0.771**	0.313	**276.91 (13.71×)**	**2461.14* (2.53×)**

On the 50 scenes in MonoDyGauBench, **SpeeDe3DGS** achieves

100+ FPS higher speed and better visual fidelity

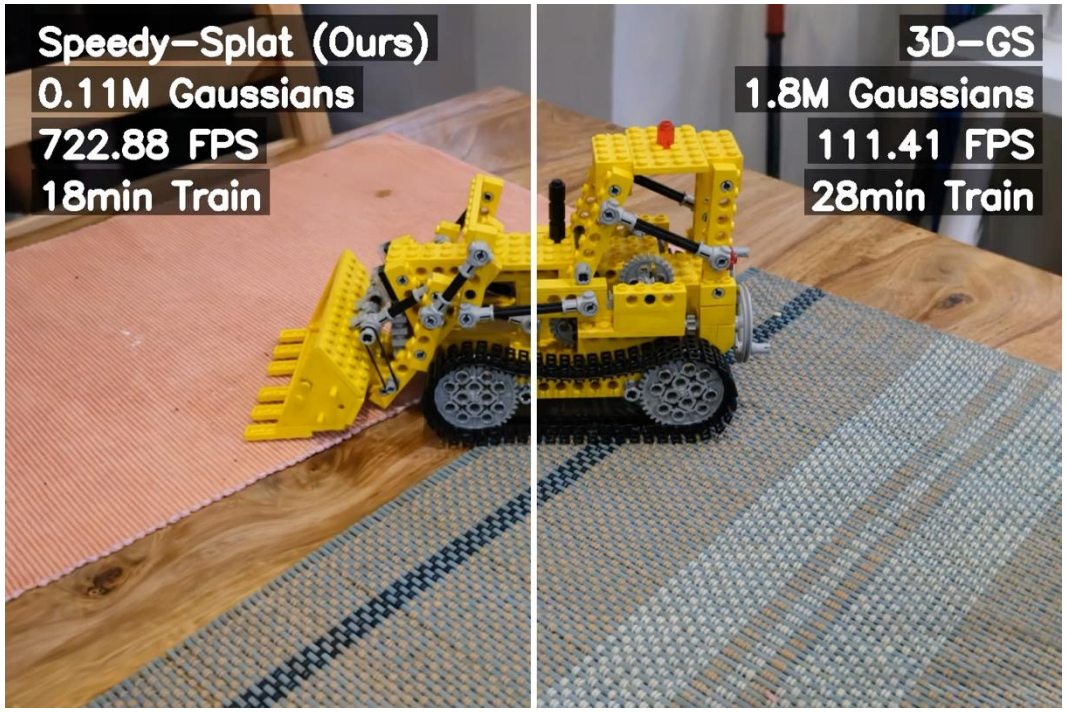
than all non-neural motion baselines.

PUP 3D-GS: Principled Uncertainty Pruning for 3D Gaussian Splatting

CVPR 2025

pup3dgs.github.io

Prune 90% of primitives from any pretrained 3D Gaussian Splatting model using a mathematically principled sensitivity score, more than tripling rendering speed while retaining more salient foreground information and higher visual fidelity than previous techniques at a substantially higher compression ratio.



Speedy-Splat: Fast 3D Gaussian Splatting with Sparse Pixels and Sparse Primitives

CVPR 2025

speedysplat.github.io

Accelerate 3D Gaussian Splatting rendering speed by 2× for free by accurately localizing primitives during rasterization and over 6× in total by pruning the scene by more than 90% during training, providing a significantly higher speedup than existing techniques while maintaining competitive image quality.

SpeeDe3DGS:

Speedy Deformable 3D Gaussian Splatting with Temporal Pruning and Motion Grouping

CVPR 2026

Allen Tu* Haiyang Ying* Alex Hanson Yonghan Lee

Tom Goldstein Matthias Zwicker

speede3dgs.github.io